Deep Learning with Limited Labelled Cardiac Data

Dani Kiyasseh
St. Cross College
University of Oxford

Supervised by
Prof. David Clifton
Dr. Tingting Zhu

Trinity Term 2021

إلى ماما، بابا، لاما، وفراس
إنّ هذا الإنجازَ إنجازُنا

ACKNOWLEDGEMENTS

I guess acknowledgements are a typical and expected preamble to many literary works. Although serving multiple functions, the acknowledgements section, more broadly, is a flawed idea that attempts to pigeon-hole and identify the perceived causal relationships between individual actors and the piece of work at hand. However, as individuals, we are inextricably connected to and impacted by our ecosystems, in their past and present form, and in ways that we cannot simply enumerate while simultaneously doing justice to the 'support system' that contributes to our lives. With that said, let us get started.

I would like to start off by thanking individuals in the Computational Health Informatics (CHI) laboratory. Professor David Clifton, thank you for providing me with the unparalleled freedom and flexibility to conduct research. Dr. Tingting Zhu, thank you for being willing to listen to the intricacies of my research and for your guidance along the way. Rasheed El-Bouri, thank you for constantly showing up with a supportive mindset and for inspiring my research paths. Farah Shamout, thank you for your words of encouragement and ongoing advice from half-way around the world. Thomas Taylor, thank you for the fruitful discussions and for introducing us to the pistachio cookies sold at the John Radcliffe hospital. Girmaw Abebe Tadesse, thank you for your support during the first year of the DPhil. To the entire crew at the Oxford University Clinical Research Unit (OUCRU) in Vietnam, Dr. Louise Thwaites, Dr. Le Van Tan, Dr. Le Nguyen Thanh Nhan, and Nhat, thank you for hosting me in the great Ho Chi Minh City, taking the time to explain autonomic nervous system dysfunction, and collecting data that contributed to my research.

Beyond the Oxford bubble, I have had the great pleasure to meet and work alongside researchers in various domains. Dr. Kenneth Fetterly and Dr. Zachi Attia at the Mayo Clinic, thank you for providing me with the opportunity to work on coronary angiograms, for your ongoing support beyond the internship, and for allowing me to catch a glimpse of what makes the Mayo Clinic what it is. Antong Chen at Merck & Co., thank you for taking me on at the Image Analytics Group, for your consistent feedback, and for your support beyond the internship.

Deep Learning with Limited Labelled Cardiac Data
Dani Kiyasseh
Thesis submitted for the degree of Doctor of Philosophy
St. Cross College
Trinity 2021

Abstract

Cardiac data, that which pertain to the heart, are a rich source of information that reflect a patient's cardiac status. Extracting clinically-useful insight from such data can be achieved via deep learning, a sub-field of artificial intelligence. Current clinical deep learning algorithms, however, are heavily dependent upon resources such as abundant data and high-quality annotations, both of which are scarce in many clinical settings. In low-resource settings, for example, the prohibitively high cost of medical infrastructure precludes the collection of data and the limited number of physicians hampers the provision of annotations. The reasons for data paucity in high-resource settings are multi-fold, ranging from stringent patient-privacy regulations to the low inter-operability of medical records. Physicians in this setting can also be disengaged due to the overwhelming number of annotation requests.

To address this challenge, in this thesis, we design deep learning algorithms that exploit cardiac data to achieve 'more with less'; less data, fewer labels, and less medical supervision during the learning process. While designing these algorithms, we focus predominantly on the clinical task of cardiac arrhythmia classification, which involves diagnosing abnormalities in the functioning of the heart.

We deploy our algorithms in three paradigms characterized by an incrementally increasing level of resource availability. In Part I of the thesis, we simulate a low-resource scenario with limited data and exploit conditional generative adversarial networks to generate cardiac time-series data for augmentation purposes. In Part II, we simulate access to abundant unlabelled data and limited labelled data. In this environment, we propose an active learning framework that dynamically determines whether an annotation should be requested from a physician or generated by an algorithm instead. We also present a family of patient-specific contrastive learning methods that improve resource-efficiency; the ability of a learner to solve a task with less data. In Part III, we deal with more extensive multi-modal data. We explore the degree to which algorithms suffer from catastrophic forgetting (the impaired ability to solve tasks from the past upon learning tasks in the present), and propose a continual learning framework to overcome this phenomenon. We also simultaneously exploit cardiac data and clinical textual reports to design a captioning system that, upon receiving cardiac signals, generates clinical reports in multiple languages.

By designing such resource-efficient frameworks, we hope to improve the accessibility of clinical deep learning algorithms, and, in turn, healthcare to vulnerable patients in low-resource settings.

PUBLICATIONS

1. **Kiyasseh, D.**, Tadesse, G. A., Nhan, L. N. T., Tan, L. V., Thwaites, L., Zhu, T., & Clifton, D. (2020). *PlethAugment: GAN-based PPG Augmentation for Medical Diagnosis in Low-resource Settings.* IEEE Journal of Biomedical and Health Informatics, 24(11), 3226-3235.

2. **Kiyasseh, D.**, Zhu, T., & Clifton, D. (2020). *The Promise of Clinical Decision Support Systems Targetting Low-Resource Settings.* IEEE Reviews in Biomedical Engineering.

3. **Kiyasseh, D.**, Zhu, T., & Clifton, D. (2021). *SoCal: Selective Oracle Questioning for Consistency-based Active Learning of Cardiac Signals.* Under review at IEEE Transactions of Pattern Analysis and Machine Intelligence.

4. **Kiyasseh, D.**, Zhu, T., & Clifton, D. (2021). *CLOCS: Contrastive Learning of Cardiac Signals Across Space, Time, and Patients.* International Conference on Machine Learning (ICML).

5. **Kiyasseh, D.**, Zhu, T., & Clifton, D. (2021). *A Clinical Deep Learning Framework to Learn Prototypes from Cardiac Signals for Interpretable Patient-Specific Diagnosis, Dataset Distillation, and Patient Retrieval.* Under Review at Nature Communications Medicine.

6. **Kiyasseh, D.**, Zhu, T., & Clifton, D. (2021). *CROCS: Clustering and Retrieval of Cardiac Signals Based on Patient Disease Class, Sex, and Age.* Advances in Neural Information Processing Systems (NeurIPS) 2021.

7. **Kiyasseh, D.**, Zhu, T., & Clifton, D. (2021). *A Clinical Deep Learning Framework for Continually Learning from Cardiac Signals Across Diseases, Time, Modalities, and Institutions.* Nature Communications.

8. **Kiyasseh, D.**, Zhu, T., & Clifton, D. (2021). *Automated Multilingual Captioning of Cardiac Signals via a Deep Neural Network.* Under Review at ICLR 2022.

CONTENTS

GLOSSARY

acquisition function (α) a user-defined metric which is used to quantify the informativeness of an instance 54, 56

cardiac arrhythmia an abnormality in the functioning of the heart 21
cardiac data data which are retrieved from, and pertain to, the heart 18
classification head (p_ω) a linear function approximator that maps a representation to probability values associated with a set of classes 56, 104, 106

electrocardiogram a clinical signal which reflects the electrical activity of the heart 18
embedding (**p** or **e**) a vector which is learned in an end-to-end manner using back-propagation and gradient descent 132

instance (**x**) a D-dimensional vector reflecting an individual data point in the input space 32, 56, 85, 105, 131, 149

label (y) a ground-truth category or annotation of an instance 32, 105, 149
lead a projection, at a certain angle, of the electrical signal generated by the heart 20
learner (f_θ or g_ϕ) a non-linear function approximator, such as a neural network, associated with a set of learnable parameters 56, 85, 105, 149
loss function ($\mathcal{L}$) an objective function which is minimized via optimization algorithms such as stochastic gradient descent 33

oracle an entity with expert domain knowledge and which is capable of annotating instances 53
output ($\hat{y}$) a C-dimensional vector reflecting the probability mass assigned to each class 56, 106, 149

photoplethysmogram a clinical signal which reflects blood volume changes in the peripheral vasculature of the human body 18

replay buffer a data structure which is used to store instances and from which instances can be retrieved 151
representation (**h** or **v**) an E-dimensional vector of features corresponding to an instance and which is either hand-crafted or generated by a neural network 56, 85, 105, 131

segment a vector reflecting the temporal information of an instance. This can also be referred to as a frame. 39

task a goal that is laid out for a neural network to achieve, e.g., classification, regression, etc. 83, 149, 170

1

INTRODUCTION

Deep learning, a subfield of artificial intelligence, involves learning from, and making predictions about, troves of data. Analogously, clinical deep learning involves extracting meaningful insight from clinical data. Such data can take on a multitude of forms, from vital signs collected via wearable sensors and biomarkers collected via routine blood-tests, to the genetic code that governs the functioning of cells. Existing algorithms are capable of, for example, identifying which hospitalized patients will develop acute kidney injury (Connell et al., 2019), whether individuals are experiencing abnormalities in the functioning of the heart (Attia et al., 2019b), and if a cancerous tumour will respond to a particular pharmaceutical agent (Adam et al., 2020). If successfully deployed, clinical deep learning algorithms have the potential to streamline medical diagnosis, prognosis, and treatment pathways, and ultimately improve patient outcomes.

At present, clinical deep learning algorithms continue to face several bottlenecks. First, they are heavily dependent upon the presence of abundant data (Thompson et al., 2020). However, in low-resource settings that lack sufficient medical infrastructure or in the event of rare medical conditions, data are scarce. Second, they are heavily dependent upon accurate, ground-truth annotations of data (Agency, 2019). Within the medical domain, such annotations require a high level of expertise and are thus typically provided by clinicians. In low-resource settings, however, clinicians can be unavailable or poorly-trained. In high-resource settings, clinicians who are increasingly experiencing 'burnout' (Shanafelt et al., 2019), are inundated

with annotation requests. Based on these observations, in this thesis, we are focused on addressing the following research question.

Research Question

How can we design clinical deep learning algorithms that are *less* dependent on **a)** abundant data, **b)** ground-truth annotations of data, and **c)** supervision by medical professionals?

1.1 RESOURCE SPECTRUM

To realize the full potential of clinical deep learning algorithms, we need to account for the entire spectrum of realistic settings encountered within healthcare. We define this spectrum based on resource availability where resources are considered to be the quantity and quality of medical infrastructure and devices, in addition to physicians and medical staff. On one end of the spectrum, we consider low-resource clinical environments. Such environments can range from hospitals in the developing world to primary care clinics in under-developed regions of developed nations. On the other end of the spectrum, we consider high-resource clinical environments. Here, medical infrastructure and high-quality physicians are available.

In this thesis, we anchor ourselves around the resource spectrum (Fig. 1.1) to allow us to better address the research question. Although such a spectrum would be continuous in nature, we discretize it into three paradigms, *doing some with less*, *doing more with less*, and *doing more with more*. In each paradigm, we pose more specific research questions and propose methods commensurate with the availability of resources, as outlined next.

1.1.1 *Doing Some With Less*

In this paradigm, we assume that our environment is characterized by the paucity of both input data and labels (annotations) in addition to the presence of severe label imbalance (see Fig. 1.1 left).

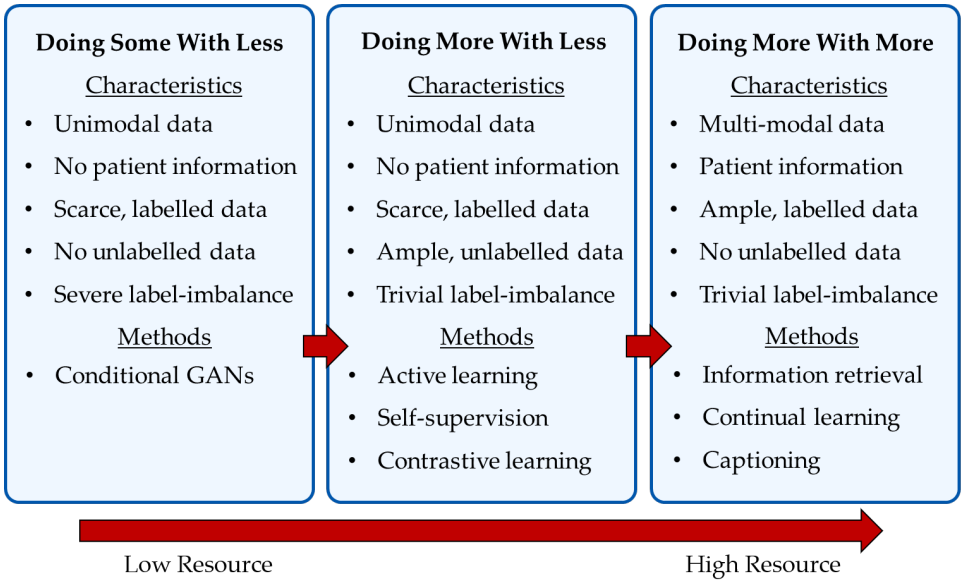


Figure 1.1: **The three main paradigms in the resource spectrum:** doing some with less, doing more with less, doing more with more, alongside the characteristics of each paradigm and the proposed methods. We will be making our way along this spectrum as the thesis progresses.

We address this challenge in the context of diagnosing the severity of medical conditions that are either infectious or chronic in nature. To do so, we build upon existing findings that have demonstrated the ability of generative adversarial networks to generate realistic images and the potential benefit of data augmentation on the generalization performance of deep learning algorithms. Specifically, we design conditional generative adversarial networks capable of generating synthetic disease-severity-specific cardiac time-series data. By augmenting our original datasets with such synthetic data, we demonstrate significant improvements in the diagnostic performance of machine learning models.

1.1.2 *Doing More with Less*

In this paradigm, we assume that our environment is characterized by abundant, unlabelled data and relatively scarce, labelled data (see Fig. 1.1 centre).

We address this challenge in three ways and in the context of identifying abnormalities in the functioning of the heart. First, inspired by findings in the consistency training literature (Xie et al., 2019), we design an active learning framework that

acquires unlabelled instances based on the degree of inconsistency of network predictions in response to perturbed network parameters and instances. We also exploit insight from selective classification (El-Yaniv and Wiener, 2010) to propose a strategy that dynamically determines whether an annotation for an unlabelled instance should be requested from a physician or generated by an algorithm instead. In the process, we find that our framework has the potential to reduce the labelling burden placed on physicians.

Beyond active learning, we take inspiration from promising findings in the contrastive learning literature (Chen et al., 2020a) and design a patient-specific contrastive learning framework as a network pre-training mechanism. In the process, we leverage temporal and spatial invariances present in cardiac signals and attract representations belonging to the same patient to one another while repelling those belonging to different patients. We find that our framework not only accelerates learning and improves generalization performance on the cardiac arrhythmia classification task, but also does so with less labelled data.

In addition to the importance of patient-specific representations to network pre-training, we realize their value in contributing to personalized diagnosis. As such, we design a supervised contrastive learning framework that learns patient-specific embeddings during training and exploits them during inference. We find that such embeddings can discriminate between disease classes and allow for the discovery of similar and dissimilar patients both within and across disparate datasets. We also find that a machine learning model trained exclusively with our patient-specific embeddings, a more compact set of training instances relative to the original dataset, achieves similar performance to one trained with the original dataset.

1.1.3 *Doing More with More*

In this paradigm, we assume that our environment is characterized by abundant, labelled data in addition to the presence of auxiliary patient attribute information such as sex and age (see Fig. 1.1 right)

We build upon our previous work in contrastive learning to learn attribute-specific clinical prototypes. These prototypes, learned in an end-to-end manner via supervised contrastive learning, efficiently summarize the cardiac state of a patient subcohort. We find that clinical prototypes are able to accurately cluster and retrieve relevant unlabelled instances from a database according to patient attribute information.

Beyond supervised contrastive learning, we explore the continual learning framework in which algorithms are exposed to data streaming in a sequential manner. Motivated by the success of replay-based continual learning methods in computer vision (Lopez-Paz and Ranzato, 2017), we design a continual learning framework for application on cardiac signals. We exploit instance-level losses to determine the importance of instances and store them in a replay buffer. We then take inspiration from our earlier work in active learning and acquire instances from this buffer based on an acquisition function. In the process, we show that our framework allows for algorithms to learn quickly on the current task while mitigating the degree to which they forget how to perform earlier tasks.

At this stage, we look beyond mere disease diagnosis and shift to summarizing such diagnoses in the form of clinical reports. Taking inspiration from image captioning (You et al., 2016) and neural machine translation (Liu et al., 2020), we design a system that, upon receiving a cardiac signal as input, returns a clinical report, in multiple languages, as a summary of the diagnostic state of the patient. In the process, we introduce a discriminative language pre-training mechanism to model the interactions of various languages. We find that our system generates accurate and reliable clinical reports in seven different languages.

We begin this dissertation by introducing cardiac data, outlining a clinical task of value; the identification of abnormalities in the functioning of the heart, and describing the various datasets that we leverage. The dissertation is then split into three main parts that explore clinical deep learning algorithms situated at various locations along the resource spectrum (see Fig. 1.1) from low to high resource availability.

Part I remains faithful to the idea of 'doing some with less' and revolves around exploiting unsupervised learning methods to improve the diagnostic performance of clinical deep learning algorithms.

In Chapter 3, we design conditional generative adversarial networks capable of generating synthetic cardiac data for data augmentation purposes. This chapter was previously published as Kiyasseh et al. (2020a).

Part II focuses on the idea of 'doing more with less' and revolves around leveraging semi and self-supervised learning methods to cope with scenarios characterized by few resources.

In Chapter 4, we design a consistency-based active learning framework that dynamically determines whether to request an annotation from an oracle or to generate one from an algorithm instead. This chapter is predominantly based on findings in Kiyasseh et al. (2020f).

In Chapter 5, we design a patient-specific contrastive learning framework that exploits abundant, unlabelled data to accelerate and improve learning on downstream tasks with minimal, labelled data. This chapter was previously published as Kiyasseh et al. (2020b).

In Chapter 6, we design a system that leverages contrastive and metric learning to perform patient-specific cardiac arrhythmia diagnosis. This chapter is predominantly based on findings in Kiyasseh et al. (2020e).

Part III addresses the idea of 'doing more with more' and revolves around achieving clinical tasks by leveraging all possible resources available.

In Chapter 7, we design a framework that clusters and retrieves unlabelled cardiac signals from a database based on patient attributes such as sex and age. This chapter is predominantly based on findings in Kiyasseh et al. (2020d).

In Chapter 8, we design a replay-based continual learning framework that identifies important cardiac signals from the past, stores them temporarily, and replays them in the future to enable efficient learning on the current task while not forgetting how to perform tasks in the past. This chapter was previously published as Kiyasseh et al. (2020c).

In Chapter 9, we propose a discriminative language pre-training mechanism that allows for the design of a multilingual captioning system that generates textual clinical reports in response to cardiac signals. This chapter is predominantly based on findings in Kiyasseh et al. (2021).

2

CARDIAC DATA

The heart is a loyal structure, beating an average of 60 times per minute for the entirety of a human's life. Its role is to pump oxygenated blood to the rest of the body for organs to exploit and deoxygenated blood to the lungs to allow for gas exchange. This function is determined and facilitated by the structure of the heart, comprising both an electrical conduction and a mechanical system (see Fig. 2.1). More specifically, the sino-atrial (SA) node generates an electrical stimulus which propagates to the atrio-ventricular (AV) node, the bundle branches, and finally the Purkinje fibers. As a result of this propagation, the heart contracts and ejects blood from its chambers. This phase is also known as systole. To eventually allow for blood to flow back into the heart, the chambers relax, a phase known as diastole.

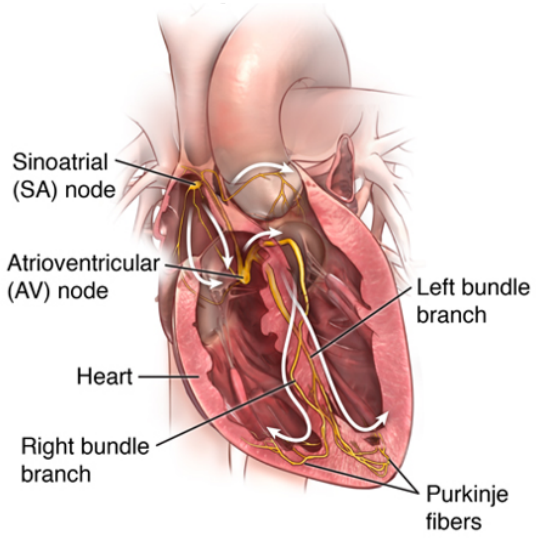


Figure 2.1: **Illustration of the heart's electrical conduction and mechanical system.** The four main chambers of the heart are the right and left atria, and the right and left ventricles. The main conduction pathway is shown in yellow, originating from the SA node and concluding at the Purkinje fibers. Figure retrieved from https://www.hopkinsmedicine.org/health/conditions-and-diseases/anatomy-and-function-of-the-hearts-electrical-system

As with any electrical system (e.g., home electricity wires) and mechanical system (e.g., home water pipes), disruptions can occur. These disruptions affect the state of the heart either transiently or permanently and can include overloaded systems, leakages, and complete blockages. To better evaluate the state of the heart, the field of cardiovascular medicine has evolved to leverage technology that probes the heart in a multitude of ways. Such probing generates rich data, from different modalities, that pertain to the heart. We refer to these as cardiac data and illustrate a subset of them in Fig. 2.2.

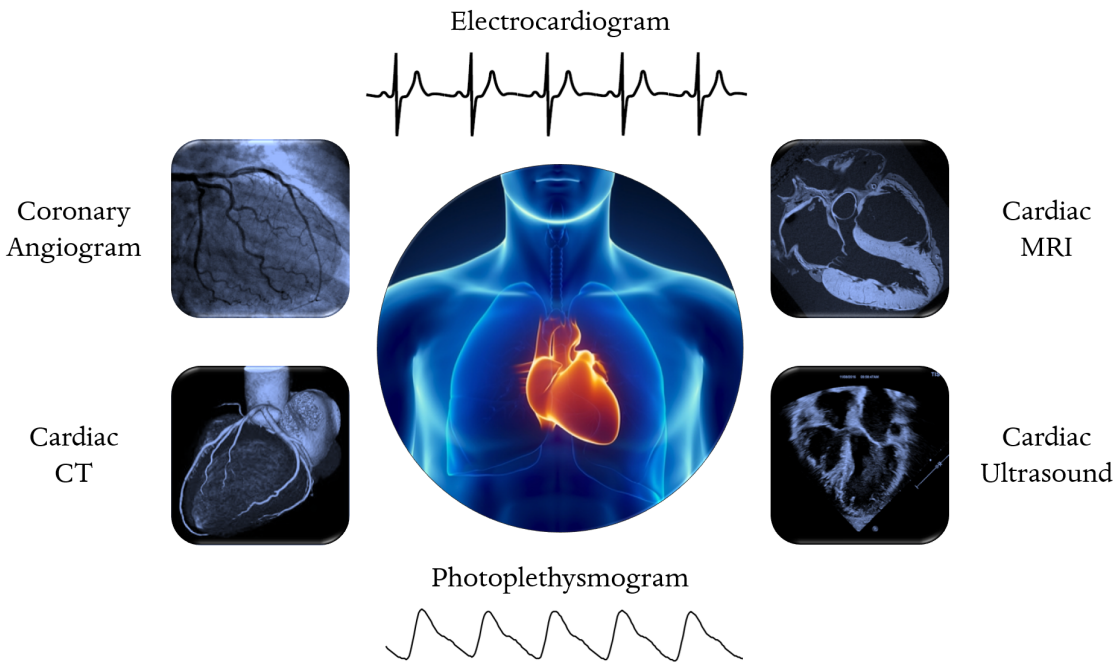


Figure 2.2: **Illustration of the heart and various cardiac data modalities.** In this thesis, we focus predominantly on cardiac time-series modalities, such as the photoplethysmogram and the electrocardiogram.

Despite the importance of cardiac imaging modalities for the diagnosis of disease and guidance of medical intervention, we focus predominantly on cardiac time-series data. Our choice of focusing on the photoplethysmogram (PPG) and the electrocardiogram (ECG) is motivated by **a)** their ubiquity in clinical settings, **b)** their existing integration into the clinical diagnosis and treatment pathways, and **c)** the

advent of wearable sensors capable of collecting such data, at low cost, beyond the clinical setting. In the next section, we provide a brief background on the PPG and ECG signal.

2.1.1 *Photoplethysmogram (PPG)*

The photoplethysmogram is a signal that reflects blood volume changes in an individual's peripheral vasculature. Such changes are brought about by the systolic and diastolic phases of the cardiac cycle. As the heart contracts to eject blood, there is a concomitant increase in the peripheral blood volume. Conversely, as the heart relaxes to fill up its chambers with blood, there is a concomitant decrease in the peripheral blood volume. The cyclical nature of systole and diastole implies that the PPG signal is also cyclical (see Fig. 2.3).

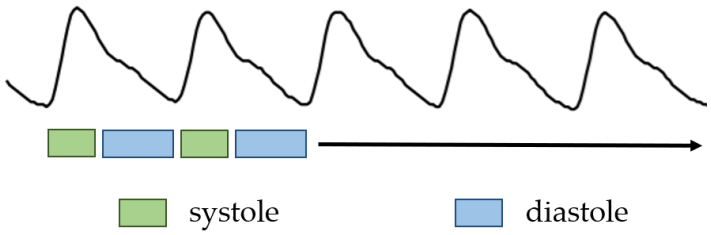


Figure 2.3: **Photoplethysmogram signal over time.** The systolic and diastolic phases of the cardiac cycle are labelled.

The PPG signal is recorded continuously for almost every patient that is hospitalized. To do so, a medical device, known as a pulse oximeter, is clipped onto a patient's finger. In addition to recording the PPG signal, the pulse oximeter also measures the concentration of oxygen in the bloodstream, an essential metric that clinicians depend upon for making decisions (Sjoding et al., 2020). This partially explains the ubiquity of the PPG signal. Beyond its use by clinicians at the patient bedside, the PPG signal contains rich information from which clinically-useful insights can be extracted. For example, it can be leveraged to diagnose abnormalities in the functioning of the heart (Elgendi, 2012) and to estimate the breathing rate of patients (Charlton et al., 2017).

2.1.2 *Electrocardiogram (ECG)*

The electrocardiogram is a signal that measures the electrical activity of the heart. We outlined the normal conduction pathway earlier which starts at the SA node and concludes at the Purkinje fibers. Given that this pathway is unidirectional, it causes certain chambers of the heart to depolarize and contract (systole) and then repolarize and relax (diastole) in a particular order. Such depolarization and repolarization is reflected in the ECG signal (see Fig. 2.4).

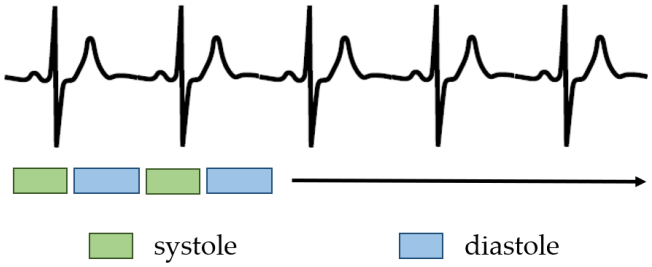


Figure 2.4: **Electrocardiogram signal over time.** The systolic and diastolic phases of the cardiac cycle are labelled (Kuhn et al., 2014). As shown, these phases roughly correspond to ventricular depolarization and repolarization, respectively.

The ECG signal is also recorded continuously for almost every patient that is hospitalized. At this point, it is important to note that the electrical activity of the heart can be measured from various angles. To enable this multi-view recording, multiple electrodes are attached to a patient's chest. Within a hospital setting, it is typical to have 12 of these recordings, also known as leads. In other words, the same electrical signal generated by the heart is projected onto 12 different ECG signals. This multi-view approach is advantageous since it increases the likelihood of detecting localized disruptions of the conduction pathway. To better illustrate this multi-view perspective, we show, in Fig. 2.5, how various leads are better positioned to detect disruptions in specific locations of the heart. For example, Leads V1-V3 (Fig. 2.5 top left) are associated with the functioning of the lateral wall of the heart. Without these leads, such disruptions would be difficult to detect.

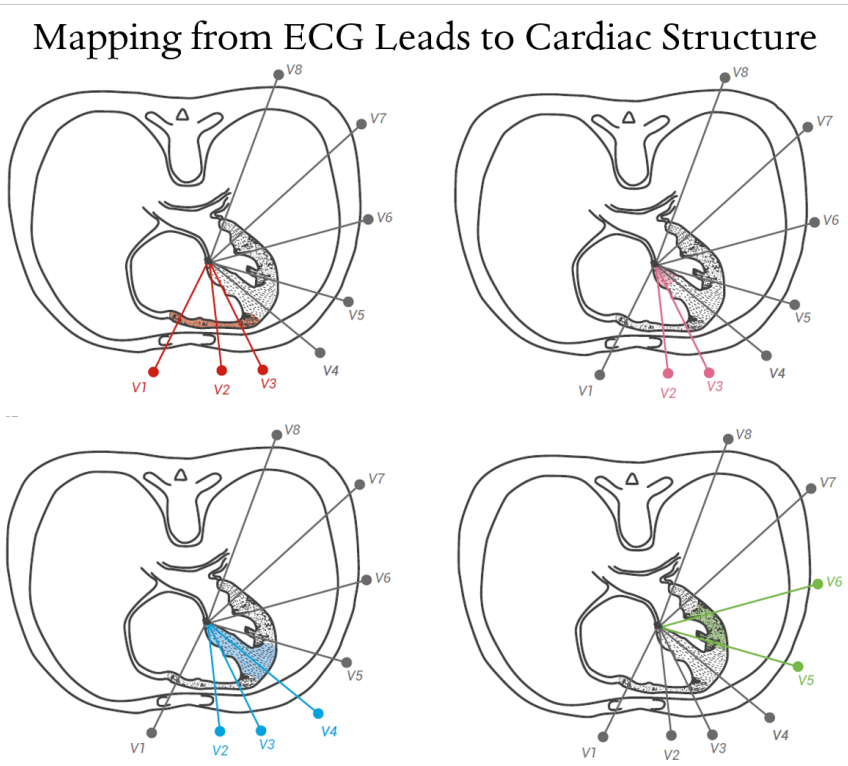


Figure 2.5: **Different leads allow for the detection of localized disruptions in the conduction pathway of the heart.** For example, Leads V1-V3 (top left) are associated with the functioning of the lateral wall of the heart (Kuhn et al., 2014). Without these leads, such disruptions would be difficult to detect.

The conduction pathway can go awry in a multitude of ways. At a high-level, this can be due to the presence of shortcuts in the pathway, a slowed conduction system, or one that is completely blocked. Such disruptions can cause a cardiac arrhythmia, an abnormality of the heart which can be identified by reading the ECG signal. This process has been mastered by cardiologists ever since the inception of the first ECG recording over a century ago (Einthoven, 1903). The growing prevalence of cardiac arrhythmias globally (Rahman et al., 2014; Lippi et al., 2020), their detrimental impact on patients' quality of life, and their diverse manifestations motivate a deeper explanation of them.

In the previous section, we outlined two modalities, namely the photoplethysmogram and the electrocardiogram, and their ability to reflect abnormalities in the functioning of the heart. Although such abnormalities, or cardiac arrhythmias, are diverse in nature, they can be categorized based on the site of origin within the heart (atria vs. ventricles) and their impact on the heart rate (tachycardia vs. bradycardia). To gain some intuition as to how cardiac arrhythmias manifest in the ECG signal, we illustrate, in Fig. 2.6, single-lead ECG signals that exhibit a small subset of cardiac arrhythmias.

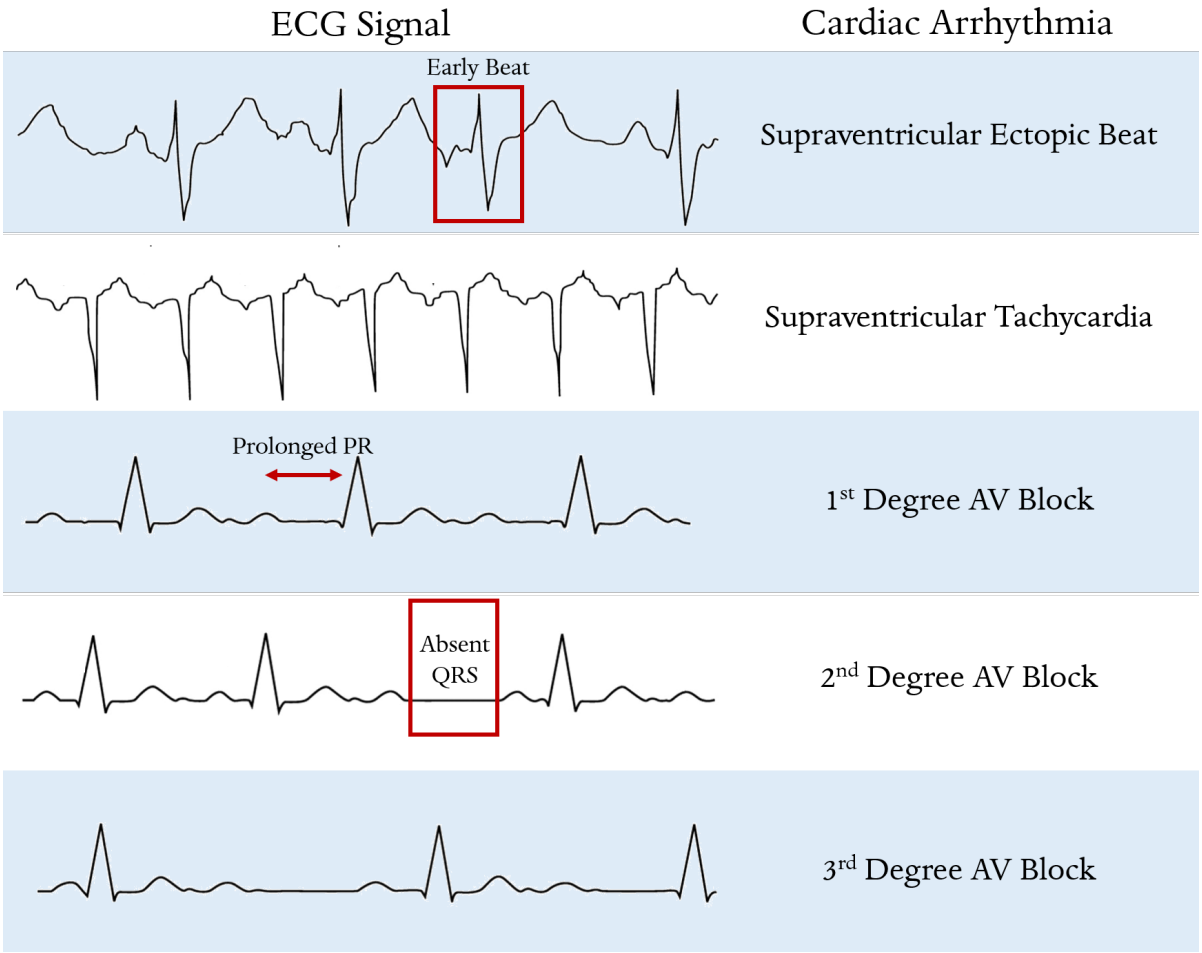


Figure 2.6: **Cardiac arrhythmias reflected in single-lead ECG signals** (Chakrabarti and Stuart, 2005). We indicate, in red, the prominent sections of the ECG signals which contribute to the cardiac arrhythmia classification. Prolonged PR refers to the interval between the P and R waves. QRS refers to the QRS complex of the ECG.

To achieve the task of cardiac arrhythmia classification, we exploit datasets that comprise either the photoplethysmogram or the electrocardiogram alongside annotations of cardiac arrhythmias. In this section, we describe and summarize the diverse set of datasets that are used to evaluate our experimental methods.

Vietnam Hand-Foot-Mouth (HFM) Disease PPG. This dataset consists of PPG recordings collected using a pulse oximeter (SmartCare Analytics Ltd., Oxford, UK) placed on the major toe of HFM-afflicted children between the ages of 3 and 6. Such data, sampled at a rate of 100 Hz, are collected from 74 patients upon admission to the pediatric intesive care unit, 6 hours after admission, and one day before discharge. Each data collection period was approximately 10 minutes in duration. Typically, HFMD severity is diagnosed based on medical criteria (Khanh et al., 2012; Hoang et al., 2019). For this dataset, diagnoses are performed by intensive care unit (ICU) physicians independently of the PPG waveform and consist of 3 classes in total.

Vietnam Tetanus PPG. This dataset consists of PPG recordings collected using a pulse oximeter (SmartCare Analytics Ltd., Oxford, UK) placed on the index finger of tetanus-afflicted adults. Such data are collected from 19 patients upon admission to the intesive care unit and one day before discharge. We only use the data from the first day of ICU admission. Each data collection period was approximately 24 hours in duration. Typically, tetanus severity is diagnosed based on clinical features outlined in the Ablett score (Ablett, 1967). For this dataset, diagnoses are performed by ICU physicians independently of the PPG waveform and consist of 3 classes in total.

China Cardiovascular Disease (CVD) PPG (Liang et al., 2018a,b). This dataset consists of PPG recordings collected via a sensor used on CVD-afflicted patients

between the ages of 21 and 86. Such data, sampled at a rate of 1KHz, are collected from 219 patients in a clinical environment. Each patient has three data collection periods each of which is 2.1 seconds in duration. The 4-class diagnosis of hypertension includes; Normotension, Pre-hypertension, Stage I, and Stage II Hypertension. Given that Normotension and Pre-hypertension are on the lower end of the severity of the medical condition, we combine data associated with such labels together. The justification for this will become more clear in Chapter 3.

PhysioNet 2015 PPG (Clifford et al., 2015). This dataset consists of PPG time-series waveforms sampled at 250Hz alongside five cardiac arrhythmia labels: Asystole, Extreme Bradycardia, Extreme Tachycardia, Ventricular Tachycardia, and Ventricular Fibrillation. Only patients with a True Positive Alarm are considered. Each frame is normalized in amplitude between the values of 0 and 1.

PhysioNet 2015 ECG (Clifford et al., 2015). This dataset consists of ECG time-series waveforms sampled at 250Hz alongside five cardiac arrhythmia labels: Asystole, Extreme Bradycardia, Extreme Tachycardia, Ventricular Tachycardia, and Ventricular Fibrillation. We only consider patients with a true positive alarm. Each frame is normalized in amplitude between the values of 0 and 1.

PhysioNet 2017 ECG (Clifford et al., 2017). This dataset consists of 8,528 single-lead ECG recordings alongside four labels: Normal, AF, Other, and Noisy. Each ECG recording varies in length between 9 and 30 seconds with a sampling rate of 300Hz. In our setup, each frame consists of 2500 samples and is not normalized in amplitude.

Cardiology ECG (Hannun et al., 2019). This dataset consists of ECG recordings collected via a chest patch from 292 patients alongside twelve cardiac arrhythmia labels: AFIB, AVB, BIGEMINY, EAR, IVR, JUNCTIONAL, NOISE, NSR, SVT, TRI-

GEMINY, VT, and WENCKEBACH. Sinus bradycardia cases are excluded from the data as done by the original authors. Each ECG recording is 30 seconds in length with a sampling rate of 200Hz. In our setup, each frame consists of 256 samples ($\approx 1s$) which is resampled to 2500 samples, and is not normalized in amplitude.

PhysioNet 2020 ECG (Perez Alday et al., 2020). This dataset consists of ECG recordings from 6,876 patients alongside nine cardiac arrhythmia labels: AF, I-AVB, LBBB, Normal, PAC, PVC, RBBB, STD, and STE. We assign multiple labels to each ECG recording as provided by the original authors. Each ECG recording varies in duration from 6 to 60 seconds with a sampling rate of 500Hz. In our setup, each frame consists of 2500 samples (5 seconds) and is normalized in amplitude between the values of 0 and 1.

Chapman ECG (Zheng et al., 2020). This dataset consists of ECG recordings from 10,646 patients alongside four high-level cardiac arrhythmia labels: Atrial Fibrillation (AF), GSVT, Sinus Bradycardia, and Sinus Rhythm. Each ECG recording is 10 seconds in duration with a sampling rate of 500Hz, which We down-sample to 250Hz. Therefore, in our setup, each frame consists of 2500 samples and is normalized in amplitude between the values of 0 and 1.

PTB-XL ECG (Wagner et al., 2020). This dataset consists of 12-lead ECG recordings from 18,885 patients alongside ECG reports and cardiac arrhythmia labels. We follow the diagnostic class labelling setup suggested by Strodthoff et al. (2020) which resulted in five classes: Conduction Disturbance (CD), Hypertrophy (HYP), Myocardial Infarction (MI), Normal (NORM), and Ischemic ST-T Changes (STTC). Each ECG recording is 10 seconds in duration with a sampling rate of 500Hz. In our setup, each frame consists of 2500 samples (5 seconds) and is standardized to follow a standard Gaussian distribution. Furthermore, we only consider recordings with one label

assigned to them.

Our focus in this thesis will be on designing clinical deep learning algorithms that are resource-efficient. As such, we present, in Table 2.1, the datasets summarized and sorted based on the amount of resources they contain. This layout will guide our choice of datasets to evaluate on at various stages of the thesis.

Table 2.1: **Summary of the datasets used to evaluate our experimental methods.** Datasets are sorted based on resource availability. For example, Cardiology does not contain multiple ECG leads, consists of data for fewer than 1k patients, and does not contain patient information or medical text. Conversely, PTB − XL satisfies all of these conditions.

Dataset	Multiple leads	Over 1k Patients	Patient Information Available	Text Available	Disease Classes
Vietnam HFMD	✗	✗	✗	✗	3
Vietnam Tetanus	✗	✗	✗	✗	3
China CVD	✗	✗	✗	✗	3
Cardiology	✗	✗	✗	✗	12
PhysioNet 2015	✗	✗	✗	✗	5
PhysioNet 2017	✗	✗	✗	✗	4
PhysioNet 2020	✓	✓	✗	✗	10
Chapman	✓	✓	✓	✗	4
PTB-XL	✓	✓	✓	✓	5

Part I

DOING SOME WITH LESS

Clinical deep learning algorithms, and particularly those that perform cardiac arrhythmia classification, are highly dependent upon the availability of a large amount of data. In the context of classification tasks, this data consists of both inputs and labels of some sort. Such high dependence on data, however, is problematic since in many scenarios, data relevant for solving the task at hand are scarce. These scenarios include low-resource clinical settings characterized by insufficient, or even the absence of, medical infrastructure. This stems from the prohibitively high cost of devices that record and display the cardiac signals of hospitalized patients. In high-resource settings, the root-causes of data paucity are multi-fold. First, this could be due to naturally-occurring phenomena such as rare genetic disorders for which scientists know little about. Additional human-based factors include stringent patient privacy regulations (Cohen and Mello, 2019; Price and Cohen, 2019) and the low level of interoperability of data that are stored in healthcare institutions (Mikk et al., 2017). We illustrate this paradigm in Fig. 2.7.

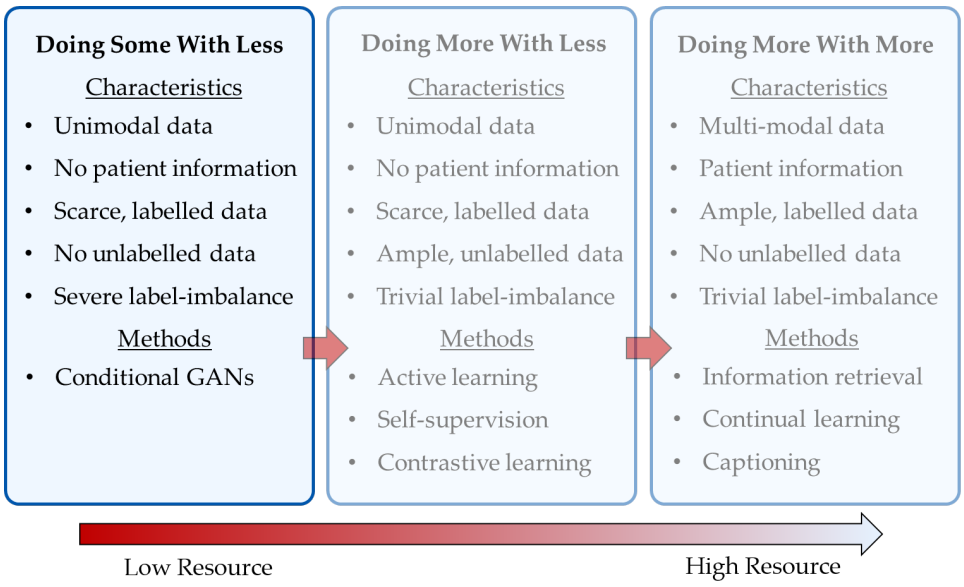


Figure 2.7: **Part I of the thesis focuses on the paradigm of 'doing some with less'.** This paradigm is characterized by scarce labelled data, few overall labels, and severe label imbalance. To overcome these challenges, we propose to leverage conditional generative adversarial networks.

3

GENERATIVE LEARNING FOR DISEASE SEVERITY DIAGNOSIS

Everybody Lies.

— House, M.D.

ACQUIRING clinical data can be costly in both low and high-resource settings. In the former, highly skilled physicians, who are required to provide ground-truth annotations, are a rarity. In the latter, physicians can be disengaged due to the overwhelming number of annotation requests. Such scenarios are ultimately characterized by a paucity of ground-truth annotations. In addition to such paucity, annotations typically suffer from a bias towards more prominent disease classes. This bias, also referred to as label imbalance, can arise due to the naturally higher prevalence of 'normal' medical cases compared to their 'abnormal' counterpart. It has been well documented that such label imbalance poses a challenge for deep learning systems (Ren et al., 2018; Zhang et al., 2020b). Based on these observations, we are interested in tackling the following question.

Research Question

How can we design clinical algorithms that better adapt to environments characterized by data paucity and label imbalance?

In this chapter, we address the outlined research question in the context of diagnosing the severity of infectious diseases and cardiac abnormalities based on the photoplethysmogram. Our contributions are threefold. First, we design three conditional generative adversarial networks (cGANs) capable of generating synthetic cardiac time-series signals. These signals, which can reflect different levels of disease

severity, are used to augment limited, labelled datasets. Second, we demonstrate that our cGAN-based data augmentation method outperforms, or is on par with, the state-of-the-art method, SpecAugment (Park et al., 2019), on four distinct datasets. Third, we propose a novel evaluation metric, entitled the area under the synthetic generalization curve (AUSGC), to better compare the performance of different cGANs.

3.1 RELATED WORK

GANs (Goodfellow et al., 2014) were first introduced as a generative model based on a mini-max formulation where two networks, the generator and discriminator, engage adversarially to outsmart one another. Shortly after, cGANs (Mirza and Osindero, 2014) were introduced as simple extensions to GANs where the generated data are conditioned on a certain variable such as a disease class or time-stamp.

Conditional generative adversarial networks for time-series. GANs have been successful in generating medical *images* for the purpose of augmenting datasets (Salehinejad et al., 2018). A recent review by Yi et al. (2019) summarizes the state-of-the-art in that domain. Given that medical image synthesis is beyond the scope of this paper, we solely focus on applications to time-series data. Although in its infancy, the application of cGANs for time-series data has seen a recent rise in activity. cGANs have been used to generate weather data conditioned on specific scenarios (Chen et al., 2018) and to generate wind and solar energy production over time conditioned on environmental variables (Wang et al., 2018a). Others introduce MuseGAN (Dong et al., 2018) to generate track-specific polyphonic music. Although their task is temporal, their data representations lack the high sampling rates of physiological signals. Gupta et al. (2018) attempt to model the potential trajectories of humans over time using an LSTM-based generator and discriminator.

In the medical domain, the work by Gregor Hartmann et al. (2018) uses a one-dimensional (1D) convolution-based GAN to generate electroencephalogram (EEG) brain signals. Inspired by this work, others generate synthetic epileptic brain activity

signals (Pascual et al., 2019) and EEG signals (Aznan et al., 2019) specifically to improve classification models. Severo et al. (2019) use a GAN to generate and open-source a privacy-protected vital sign dataset. In Brophy et al. (2019), PPG and ECG data are generated using the two-dimensional (2D) convolution-based DCGAN. Here, time-series data are converted to images before being input into the GAN. Both of these works, however, do not aim to generate class-specific signals. Although Zhang and Liu (2018) use a conditional DCGAN to generate EEG data, they perform their operations in the imaging domain and do not evaluate the representativeness of the synthetic EEG data. Closest to our work is that of Esteban et al. (2017) which uses an LSTM-based cGAN to produce various time-series data, including sine waves, some medical data, and sequential MNIST benchmark data. The medical time-series generated, however, is of summary numerics such as heart rate and oxygen saturation as opposed to high frequency medical data. Notably, they introduce an evaluation metric known as 'train on synthetic, test on real' (TSTR) which we build upon in our work. Lastly, although not used for time-series, DSGAN (Yang et al., 2019) involves a diversity sensitivity term that rewards conditional GANs for diverse data generation. We will exploit this term in our work.

Data augmentation for time-series. Given the improved results associated with data augmentation in computer vision (Krizhevsky et al., 2017), recent work converts time-series into image-representations (Wang et al., 2019b; Park et al., 2019). The work by Um et al. (2017) provides a good overview of data augmentation methods to employ on time-series data from wearable sensors. This includes random jitter, window-slicing, changing permutations, and time-warping. The latter is used before implementing a convolutional neural network (Guennec et al., 2016) and to boost the performance of a deep ResNet classification network (Fawaz et al., 2018). Unfortunately, the aforementioned approaches could be detrimental in our application especially when dealing with physiological conditions that impact a signal's frequency component. The addition of noise from a Gaussian distribution with varying standard

deviations has been used to improve the classification performance of three different models (SVM, LeNet, ResNet) on various datasets (Wang et al., 2018a). While promising, the results are inconsistent and the methodology does not seem to generalize well. In the music domain, Thickstun et al. (2017); McFee et al. (2015); Mauch and Ewert (2013) leverage an audio degradation toolbox that introduces perturbations to the original data. To avoid domain-specific augmentation problems, additive noise is proposed (DeVries and Taylor, 2017), in addition to interpolation and extrapolation in the feature space as a form of data augmentation before data are fed into a classifier. In contrast to traditional augmentation approaches, an end-to-end model that learns invariant transformations to apply to the original data is also proposed (Oh et al., 2018). Although this resulted in minor classification improvement, their approach was limited to low-frequency data.

3.2 BACKGROUND

3.2.1 *Data Augmentation*

Let us assume we have access to a dataset, $\mathcal{D} = \{\boldsymbol{x}_i, y_i\}_{i=1}^{N}$, that consists of N instances, $\boldsymbol{x} \in \mathbb{R}^D$, that are each D-dimensional, and corresponding labels, $y \in [1, C]$, where C is the total number of classes (e.g., disease-severity levels). Data augmentation focuses on expanding the original dataset, $\mathcal{D}$, with an artificial dataset, $\mathcal{D}' = \{\boldsymbol{x}', y'\}$, which consists of instances, $\boldsymbol{x}'$, and labels, y'. As a result, we arrive at an augmented dataset, $\mathcal{D}_{aug} = \mathcal{D} \cup \mathcal{D}'$, that is more representative of the underlying data distribution and may thus facilitate learning. In this chapter, we focus on generating this artificial dataset, $\mathcal{D}'$, via a generative adversarial network, as explained next.

3.3.1 *Conditional Generative Adversarial Networks*

We design a conditional GAN-based time-series data augmentation methodology to improve the performance of classification models. To achieve this, we adapt three conditional GAN models that have been successful in generating diverse images. When training such models, we exploit advice about improving training stability and performance (Salimans et al., 2016).

Vanilla cGAN with diversity sensitivity. The 'vanilla cGAN' incorporates a conditional variable at any point within the generator, G, and/or discriminator, D, network. We opt to concatenate a one-hot encoding of the class of the PPG to the input of the generator (see Fig. 3.1 top). Next, we describe the objective functions optimized for the generator and discriminator, respectively.

Generator. The generator is trained using a loss function that consists of three terms; 1) a Jensen-Shannon loss term, $\mathcal{L}_{JS}$, (3.1) that penalizes the network for generating unrealistic synthetic data, $\boldsymbol{x'}$, from some distribution P_g, 2) an auxiliary cross-entropy loss, $\mathcal{L}_{CE}$, (3.2) that penalizes the network for generating synthetic data that cannot be correctly classified as the ground truth, c, and 3) our proposed class-specific diversity sensitivity loss, $\mathcal{L}_{DS}$ (3.9), with a hyperparameter λ_{div}, that penalizes the network for generating synthetic data that are *not* diverse (see Sec. 3.3.3 for in-depth description). The effect of this loss term on the overall system can be found in Section I here.

$$\mathcal{L}_{JS} = -\mathbb{E}_{\boldsymbol{x'}\sim P_g}\left[\log(D(\boldsymbol{x'}))\right] \tag{3.1}$$

$$\mathcal{L}_{CE} = -\mathbb{E}_{\boldsymbol{x'}\sim P_g}\left[\log(p(y=c|\boldsymbol{x'}))\right] \tag{3.2}$$

$$\mathcal{L}_G = \overbrace{\mathcal{L}_{JS}}^{\text{Jensen-Shannon loss}} + \overbrace{\mathcal{L}_{CE}}^{\text{cross-entropy loss}} + \overbrace{\lambda_{div}\mathcal{L}_{DS}}^{\text{diversity sensitivity loss}} \tag{3.3}$$

Discriminator. Independently of the generator, the discriminator is trained using a loss function that also consists of three terms; i) a Wasserstein loss, $\mathcal{L}_W$ (3.4) (Arjovsky et al., 2017) that penalizes the network for classifying the synthetic data as realistic and the real data as synthetic, ii) a gradient penalty of zero, $\mathcal{L}_{GP}$ (3.5) (Thanh-Tung et al., 2019) that was found to improve training stability, and iii) an auxiliary cross-entropy loss, $\mathcal{L}_{CE}$ (3.6) that penalizes the network for incorrectly classifying the real data.

$$\mathcal{L}_W = \mathbb{E}_{\boldsymbol{x}' \sim P_g}\left[D(\boldsymbol{x}')\right] - \mathbb{E}_{\boldsymbol{x} \sim P_r}\left[D(\boldsymbol{x})\right] \tag{3.4}$$

$$\mathcal{L}_{GP} = \mathbb{E}_{\bar{x}}\left[\|\nabla_{\bar{\boldsymbol{x}}} D(\bar{\boldsymbol{x}})\|^2\right] \tag{3.5}$$

$$\mathcal{L}_{CE} = -\mathbb{E}_{x \sim P_r}\left[\log(p(y = c|\boldsymbol{x}))\right] \tag{3.6}$$

$$\mathcal{L}_D = \overbrace{\mathcal{L}_W}^{\text{Wasserstein loss}} + \overbrace{\mathcal{L}_{GP}}^{\text{gradient penalty}} + \overbrace{\mathcal{L}_{CE}}^{\text{cross-entropy loss}} \tag{3.7}$$

where P_r represents the distribution of the real data, ∇ represents the gradient operator and $\bar{\boldsymbol{x}} = \alpha \boldsymbol{x} + (1 - \alpha)\boldsymbol{x}'$ is a linear combination of the real and synthetic data with $\alpha \sim \mathcal{U}(0, 1)$ sampled from a Uniform distribution, as suggested by the original authors.

DeLiGAN with diversity sensitivity. Gurumurthy et al. (2017) proposed DeLiGAN to deal with diverse and limited data regimes. They assume that a Gaussian mixture model dictates the data generating process and, as such, attempt to learn the parameters (mean, variance) of such a model (see Fig. 3.1 left). Although the original formulation regularized the variance parameter during optimization, we opt instead to optimize the diversity sensitivity term, $\mathcal{L}_{DS}$. Furthermore, we revert to the traditional Jensen-Shannon loss term. Our generator and discriminator loss terms are thus represented by (3.3) and (3.7), respectively.

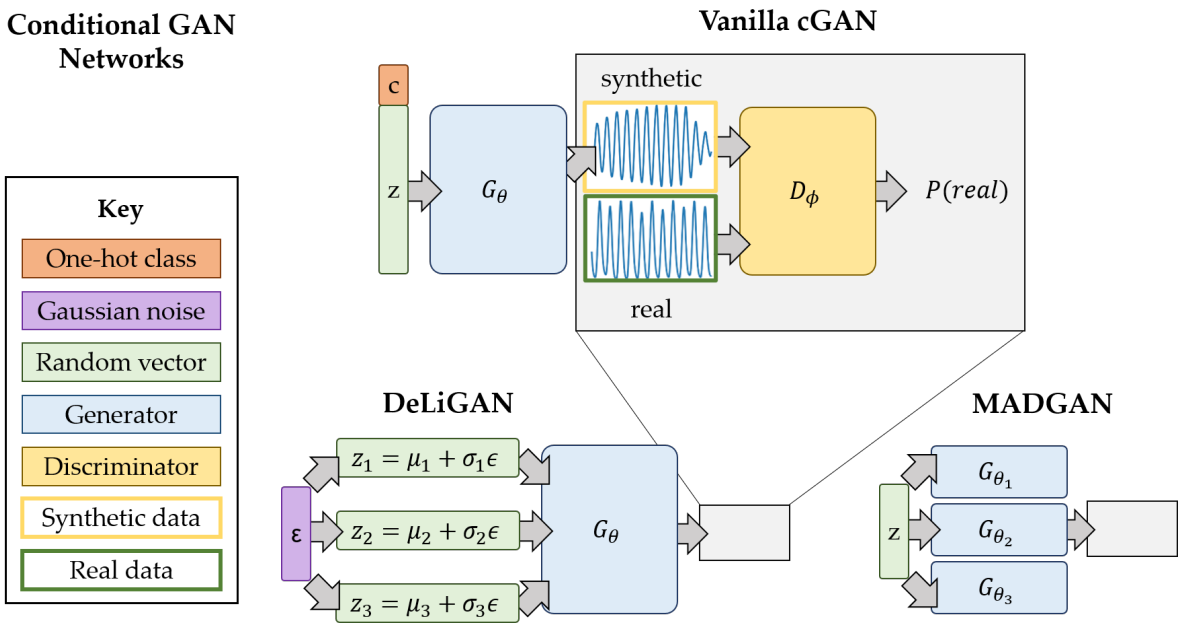


Figure 3.1: **Three distinct conditional generative adversarial networks for generating class-specific ($C = 3$) synthetic PPG time-series data**. For DeLiGAN and MADGAN, we replace the discriminator with a grey box to avoid clutter. Class information is represented by **(Vanilla cGAN)** a one-hot encoding, c, and is appended to the input vector, z, **(DeLiGAN)** the number of components in the Gaussian mixture model, and **(MADGAN)** the number of generator networks.

MADGAN. MADGAN (Ghosh et al., 2018) is proposed as a way to explicitly generate data from different classes. This is achieved by having a distinct generator for each class (see Fig. 3.1 right). Our generator loss consists of a Jensen-Shannon loss term of the form in (3.1) for each generator and is as follows:

$$\mathcal{L}_G = \mathcal{L}_{JS_1} + \mathcal{L}_{JS_2} + \mathcal{L}_{JS_3} \tag{3.8}$$

The discriminator is tasked with identifying whether the data are real or synthetic, and in the case of the latter, to further identify the generator from which it came. Our discriminator loss is therefore the same as that suggested in the original paper.

3.3.2 *Intuition Behind cGAN-based Data Augmentation*

The cGANs can be thought of as generating class-specific synthetic instances whose distribution, on average, matches that of class-specific real instances. Notice that

while traditional data augmentation involves transformations applied to individual instances, cGANs operate at the distribution level.

We hypothesize that three main elements of our cGANs encourage this "distribution matching" behaviour. First, class-specific real data must be presented to the discriminator. This is the first point of entry for information in the system. It exposes the cGAN to the distribution of real data (through mini-batches during training) that it is trying to capture. Second, distribution matching is facilitated by penalizing the generator, G, for generating synthetic instances that are unable to fool the discriminator into thinking that they are real. This penalty term is captured by the Jensen-Shannon loss term in the objective function of the generator. Once the generator is able to generate such instances, we can be more confident in the similarity of these instances to the real counterparts. Note that although we are looking to generate synthetic instances that are similar to the real instances, we do not want synthetic instances that are exactly the same as their real counterparts. The reason is that the latter scenario, when deployed for data augmentation purposes, would be equivalent to simply duplicating the original dataset. Doing so is likely to lead to overfitting and thus hinder the generalization performance of a model. The last element of importance is that of ensuring the discriminator is capable of accurately distinguishing between real and synthetic instances. This is captured by the Wasserstein loss in the discriminator objective function. Failure to do so would mean that synthetic instances are trivially misclassified as real instances, providing the generator with a false sense of achievement that it has been able to generate realistic synthetic instances. Therefore, a strong discriminator is essential for the evolution of a strong generator.

To better understand how the aforementioned distribution matching facilitates learning (and by extension to what data modalities it can generalize to), let us consider the scenario in Fig. 3.2, which is partially inspired by Antoniou et al. (2017). In the left panel, we illustrate instances in the training set associated with two distinct classes

and the decision boundary that is learned from such data. We do not augment the original training set in this case. Based on this decision boundary, an instance in the test set that belongs to class 1 might end up on the wrong side of the decision boundary and thus be misclassified as belonging to class 2. In the right panel, we show how the inclusion of synthetic instances with a similar distribution to that of the real instances could impact the decision boundary that is arrived at. As a result of this modified decision boundary, we can now correctly classify the test instance during inference.

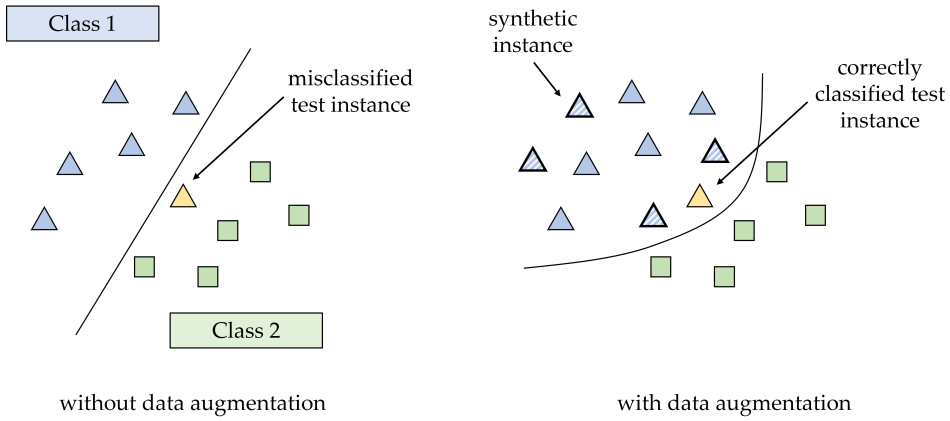


Figure 3.2: **Benefit of data augmentation with synthetic instances. (Left)** Decision boundary learned based on training instances (without data augmentation) from two classes. As a result, the test instance (yellow) is misclassified. **(Right)** Modified decision boundary learned based on the training instances augmented with synthetic instances. Data augmentation can allow for the test instance to be correctly classified.

3.3.3 *Encouraging Intra-class Diversity*

When generating synthetic data from distinct classes, "diversity" can be used to refer to distinct instances within the same class (intra-class diversity) or to those across distinct classes (inter-class diversity). Conditional generative adversarial networks, if trained successfully, are likely to result in inter-class diversity. In our context, this implies that practitioners now have the ability to generate disease severity-specific synthetic instances for data augmentation purposes. The benefit of this was demonstrated in Fig. 3.2.

It can also be important to generate synthetic instances that exhibit intra-class diversity. First and foremost, such diversity is important to avoid mode-collapse within the same class. This phenomenon would equate to the exact same instances being generated from the same class. Such an outcome would be analogous to duplicating instances and would thus quickly lead to overfitting. This statement holds regardless of whether the underlying real data distribution exhibits a high degree of intra-class diversity or not. On the other hand, encouraging too much intra-class diversity can be problematic. For example, doing so can cause synthetic instances believed to be from class 0 to be closer to real instances from class 1 than those from class 0. In other words, the cGAN no longer faithfully performs its job of exclusively generating synthetic instances which are class-specific. In this case, synthetic instances can be mislabelled, thus introducing label noise, which hampers the learning process.

Motivated by the importance of generating sufficiently diverse class-specific data, we propose a class-specific diversity sensitivity loss term, $\mathcal{L}_{DS}$ (Yang et al., 2019). Note that classes can be defined arbitrarily and may depend on the dataset used. In our context, classes represent various disease-severity levels.

$$\mathcal{L}_{DS} = -\mathbb{E}_c \left[\mathbb{E}_{z_1, z_2} \left[\frac{\| G_\theta(\boldsymbol{z_2}|c) - G_\theta(\boldsymbol{z_1}|c) \|}{\| \boldsymbol{z_2} - \boldsymbol{z_1} \|} \right] \right] \tag{3.9}$$

where the outer expectation is with respect to all classes, G_θ represents the generator network, and $\boldsymbol{z_1}$ and $\boldsymbol{z_2}$ represent any two input noise vectors belonging to the same class c. Intuitively, this term rewards the generator based on its sensitivity to a change in input. Extreme mode-collapse, for instance, results in a sensitivity of zero because the same output would be generated for two different noise inputs, i.e., $G_\theta(\boldsymbol{z_2}|c) = G_\theta(\boldsymbol{z_1}|c)$. Thus, a null reward value is returned. We incorporate this term into our proposed cGAN models in the hope of encouraging intra-class diversity. We refer to our family of conditional generative adversarial networks as PlethAugment and illustrate the complete diagnostic pipeline in Fig. 3.3.

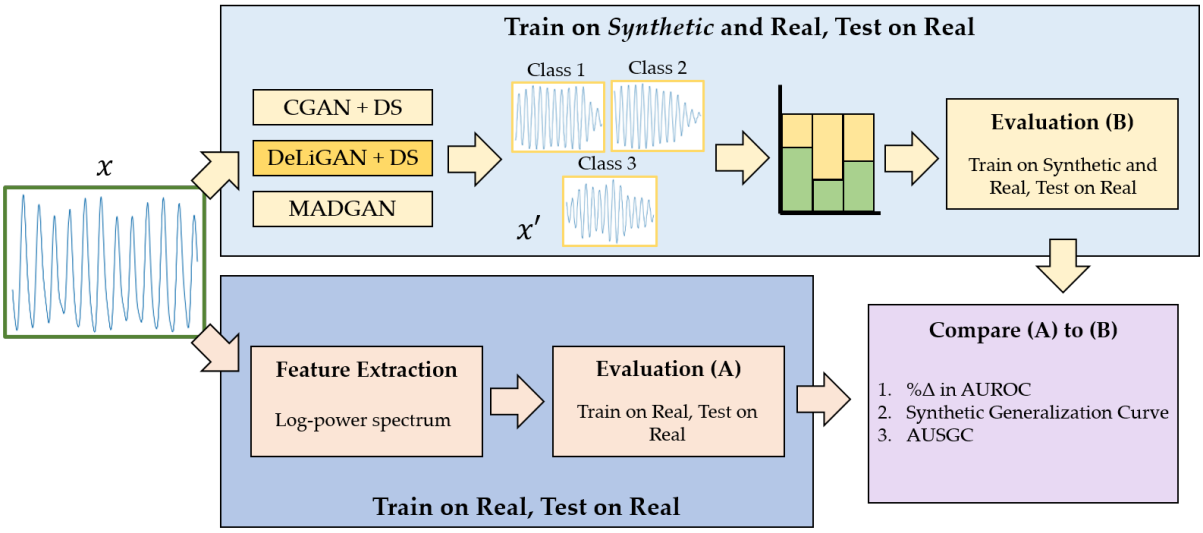


Figure 3.3: **PlethAugment pipeline.** PPG data are used to train the three distinct cGANs independently of one another to generate class-specific synthetic data. Training is performed on the augmented dataset comprising both synthetic and real instances. The performance of this model is then compared to that of its non-augmented counterpart.

3.4 EXPERIMENTAL DESIGN

3.4.1 *Data and Pre-processing*

To evaluate our method, we leverage four different datasets, each of which consist of cardiac time-series waveforms alongside labels corresponding to the severity of various diseases. We split each of the aforementioned waveforms into non-overlapping segments comprising 500 samples. In Table 3.1, we a present a summary of these datasets. We focus on these specific PPG datasets as they suffer the most from data-scarcity and label-imbalance relative to the remaining datasets.

Table 3.1: **Summary of the datasets used for evaluation.** We also show additional pre-processing information. Please click on the dataset's name for more information.

Dataset	Abbreviation	Modality	Normalization	Extra Info.
HFM	$\mathcal{D}_1$	PPG	✗	3-way classification
Tetanus	$\mathcal{D}_2$	PPG	✗	3-way classification
CVD	$\mathcal{D}_3$	PPG	✗	3-way classification
PhysioNet 2015	$\mathcal{D}_4$	PPG	✗	3-way classification

3.4.2 *Evaluating Generative Adversarial Networks*

There are many ways to evaluate GANs, as summarized by Borji (2019). Although we take inspiration from some of these techniques, our focus does not lie here. Given our desire to quantify the potential improvement in medical diagnosis offered by data augmentation, we build upon the work introduced by Esteban et al. (2017) and further propose a novel evaluation method.

With time-series data, in contrast to computer vision, assessing the quality and representativeness of synthetic data is not straightforward. Moreover, a common pitfall of generative adversarial networks is mode collapse where the generator fails to produce diverse samples (both intra- and inter-class); i.e., there exists a many-to-one or many-to-few mapping of random variable, z, to synthetic image, $\boldsymbol{x'}$. This is especially problematic in the conditional GAN case where some inter-class diversity is expected in the generated data. Thus, we evaluate our GANs in the follows ways.

Representativeness of synthetic data. We use the kernel maximum mean discrepancy (MMD) (Gretton et al., 2012), a common evaluation method for GANs. This metric quantifies the similarity of synthetic data to real data by using a kernel function, $K \in \mathbb{R}$, such as the exponentiated quadratic, which quantifies the similarity of two vectors, $\boldsymbol{x}$ and $\boldsymbol{x'}$.

$$K(\boldsymbol{x}, \boldsymbol{x'}) = e^{-\|\boldsymbol{x}-\boldsymbol{x'}\|^2} \tag{3.10}$$

If $\boldsymbol{x}$ and $\boldsymbol{x'}$ are exactly the same, then the kernel function evaluates to one. The more dissimilar they are from one another, the smaller the value is, which is lower-bounded by zero. Since the original MMD metric fails to illustrate the more granular class-specific similarities, we introduce MMD_c; a conditional MMD metric that allows us to compare *class-specific* performance across different GANs, as shown below.

$$\text{MMD}_\text{c} = \sum_{i \neq i'} K_{ii'} - 2 \sum_{i \neq j} K_{ij} + \sum_{j \neq j'} K_{jj'} \tag{3.11}$$

where K is based on (3.10) and $\boldsymbol{x}'$ and $\boldsymbol{x}$ reflect the synthetic and real data, respectively. $K_{ii'} = K(\boldsymbol{x}_i'^c, \boldsymbol{x}_{i'}'^c)$ compares synthetic data from the same class, and $K_{ij} = (\boldsymbol{x}_i'^c, \boldsymbol{x}_j^c)$ compares synthetic and real data from the same class, c.

Class diversity. Diversity in the generated data both within and across classes is important to detect. The latter, for example, helps evaluate the *conditional* component of the cGAN. Since the MMD obscures this class-specific calculation, we explicitly calculate it through exponentiated quadratic kernels (3.10).

3.4.3 *Evaluating Data Augmentation*

We call the process of training on a dataset augmented with synthetic data and testing on the real dataset 'train on synthetic and real, test on real' (TSRTR). The outcome of this, when compared to a non-augmented baseline, 'train on real and test on real' (TRTR), allows us to see the effect of a data augmentation policy, ψ, on the performance of a machine learning model. Note that augmentation can be detrimental, as observed by Tran et al. (2017). We define a data augmentation policy, ψ, as a set of three parameters that dictate how to augment the original data: 1) choice of class to imbalance, 2) degree of synthetic imbalance, and 3) ratio of synthetic to real data. For evaluation, we use leave N-patients-out cross validation on 10 diverse classification models; Naive Bayes, Linear and Quadratic Discriminant Analysis, k-Nearest Neighbours, Logistic Regression, Support Vector Machines, Decision Tree, Random Forest, Adaboost, and Multilayer Perceptron. Formally, for a model, M, trained via a particular augmentation policy, ψ, we compare its performance, AUROC_{TSRTR} to that of a model trained exclusively on real data, AUROC_{TRTR}.

$$\%\Delta_M = \frac{\text{AUROC}_{TSRTR} - \text{AUROC}_{TRTR}}{\text{AUROC}_{TRTR}} \cdot 100 \tag{3.12}$$

3.4.4 *Evaluating GANs for Data Augmentation*

The above evaluation method is limited and simply provides us with the performance of an individual classification model for a particular augmentation policy. To obtain a holistic evaluation of all classification models for all augmentation policies, and thus provide a more realistic evaluation of any GAN, we propose the *Synthetic Generalization Curve.* Such a metric quantifies the extent to which each classification model, M, is over-or under-performing relative to a baseline. Mathematically, a point on the curve, which we call the synthetic generalization (SG), can be calculated as follows:

$$SG(\text{AUROC}, \varepsilon, \psi) = \frac{1}{M} \sum_{m=1}^{M} \mathbb{1}\left(\text{AUROC}_{TSRTR} \geq (1-\varepsilon) \cdot \text{AUROC}_{TRTR}\right) \tag{3.13}$$

where $\mathbb{1}$ is an indicator function which evaluates to one if its argument is true and zero otherwise. The SG is performed for a particular augmentation policy ψ from the pool of policies Ψ. Intuitively, when $\varepsilon \neq 1$, the classification model in the augmented scheme AUROC_{TSRTR} is compared to the baseline $(1-\varepsilon) \cdot \text{AUROC}_{TRTR}$. For example, when $\varepsilon < 0$, the SG represents the percentage of classification models in the augmented scheme that outperform those in the baseline by at least $-\varepsilon \cdot 100$ percentage points. From this curve, a novel metric naturally follows: the *Area Under the Synthetic Generalization Curve* or *AUSGC*. This curve can be averaged over many augmentation polices to allow for a more realistic comparison of the performance of different types of GANs.

3.5 RESULTS

In this section, and with an eye on addressing the initial research question, we attempt to answer the following questions: (i) Does our family of cGANs generate realistic photoplethysmogram data? (ii) To what extent does augmentation with such synthetic data improve the generalization performance of algorithms? (iii) How does the synthetic generalization curve affect our ability to compare various cGANs?

3.5.1 *Performance of Proposed cGANs*

We quantify the representativeness of the synthetic data via the MMD values in Table 3.2, where a lower value implies that the synthetic data is more realistic. An average is taken over 10 seeds with each seed containing 30 and 15 randomly sampled data points from the appropriate distributions of the HFM and CVD datasets, respectively. Fewer samples are chosen for the CVD dataset due to the small sample size in the original dataset. We also propose the use of MMD values in order to discern interclass differences. Such a granular approach facilitates the identification of potential causal relationships between network/hyperparameter changes and the representativeness of synthetic data. This can ultimately guide researchers working with *conditional* GANs.

When considering all classes, we can observe that MADGAN generates data that most resembles the true underlying distribution for both datasets. A closer look at the HFM MMD_c values, however, indicates that cGAN+DS is able to produce the most realistic class 1 data. Conversely, DeLiGAN+DS appears to generate the least realistic synthetic data as observed by its relatively high MMD_c and MMD values. We believe such a situation may arise due to the over-powering effect of the constraints placed on the DeLiGAN+DS network such as the diversity-sensitivity loss. In other words, the network could have prioritized generating diverse classes over generating realistic classes. We also compare the real and synthetic data by visualizing them in a 2-dimensional t-SNE (Maaten and Hinton, 2008) subspace and calculating the pairwise L_2 distance between them. More details can be found in Appendix A.1.1.

In addition to the representativeness of the synthetic data, we must ensure that the cGANs are not suffering from mode-collapse, i.e., data generated from each class must be sufficiently diverse. This diversity is illustrated in Fig. 3.4 where the exponentiated quadratic kernel is applied to 30 randomly sampled synthetic datapoints from each class and model combination. For each such combination, the resulting symmetric

Table 3.2: **Average maximum mean discrepancy (MMD) of synthetic data.** Since ↓ MMD implies ↑ realistic synthetic datapoints, we find that MADGAN generates the most realistic class-specific synthetic PPG data. All calculates the MMD between real and synthetic data independently of class.

Dataset	Class	cGAN+DS	DeLiGAN+DS	MADGAN
HFM	1	**0.84±0.089**	0.87±0.066	0.89±0.089
	2	0.85±0.085	0.97±0.087	**0.85±0.066**
	3	0.94±0.046	1.03±0.033	**0.85±0.034**
	All	0.87±0.048	0.90±0.032	**0.69±0.034**
Tetanus	1	0.50±0.040	0.52±0.029	**0.25±0.023**
	2	0.53±0.040	0.69±0.022	**0.36±0.023**
	3	0.50±0.038	0.69±0.041	**0.44±0.027**
	All	0.50±0.017	0.60±0.031	**0.26±0.011**
CVD	1	0.88±0.082	1.05±0.089	**0.81±0.078**
	2	0.92±0.089	1.11±0.120	**0.89±0.100**
	3	**0.85±0.086**	0.87±0.067	0.91±0.099
	All	0.88±0.038	0.91±0.054	**0.66±0.034**
PhysioNet	1	0.69±0.053	0.48±0.048	**0.40±0.050**
	2	0.76±0.043	**0.51±0.036**	0.51±0.040
	3	0.73±0.060	0.58±0.051	**0.47±0.070**
	All	0.72±0.031	0.49±0.019	**0.40±0.031**

matrix is truncated to only show its lower triangular region. Darker elements indicate synthetic datapoints that are similar to one another; a potential sign of class-specific mode-collapse. Conversely, lighter values indicate datapoints that are dissimilar from one another. Although this hints at the existence of intra-class diversity, it could also be a sign that the synthetic datapoint should not even belong to that class. This latter case would confuse classification models and negatively impact their performance. The intra-class similarity matrices belonging to the remaining datasets can be found Appendix A.1.1.

To quantitatively evaluate the intra-class similarity of synthetic datapoints, we take the average of the off-diagonal elements of the 30×30 kernel matrices shown in Fig. 3.4 and present them in Table 3.3. Moreover, we mitigate the impact of a small sample size by averaging this across 10 different sets of 30 randomly sampled synthetic datapoints. Since we are aiming for high diversity, or equivalently low similarity, the lower the value the better.

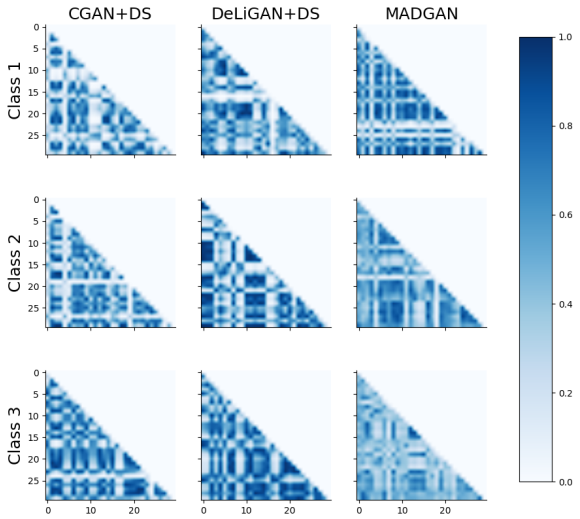


Figure 3.4: **Intra-class similarity matrices of class-specific datapoints generated by three cGANs.** For each case, 30 randomly sampled synthetic datapoints are generated by the three different cGANs (columns) for each of the three classes of HFM (rows). Results are shown for one seed.

Based on this intuition, we can observe that cGAN+DS on the HFM dataset suffers the least from mode-collapse when generating data from Class 1. The poorer diversity observed in Class 3 implies that its generator became more focused on the conditional component of the input than on the random variable. In principle, all three of the cGAN frameworks are motivated by the desire to generates instances with a satisfactory level of intra-class diversity. However, we find that the level of intra-class diversity exhibited by synthetic instances generated by MADGAN is higher than that of instances from the remaining cGAN frameworks. One hypothesis for this is related to the increased capacity of the generator(s) in the MADGAN framework relative to the remaining frameworks. Specifically, MADGAN contains three distinct generators (with non-shared parameters), one for each class. In contrast, cGAN and DeLiGAN share the same set of parameters for the distinct classes. This increased capacity could be contributing to the observed increased intra-class diversity. It is worthwhile to note the correlation of the results in Table 3.2 and Table 3.3. We observe that the most diverse (intra-class) scenarios are the ones that correspond to the most

Table 3.3: **Average intra-class similarity of synthetic data.** Since ↓ similarity implies ↑ diversity, we find that MADGAN generates the most diverse synthetic class-specific PPG data.

Dataset	Class	cGAN+DS	DeLiGAN+DS	MADGAN
HFM	1	**0.46±0.044**	0.49±0.039	0.51±0.034
	2	0.47±0.046	0.59±0.043	**0.47±0.024**
	3	0.55±0.051	0.64±0.025	**0.46±0.019**
Tetanus	1	0.44±0.025	0.38±0.019	**0.31±0.021**
	2	0.45±0.026	0.49±0.018	**0.42±0.022**
	3	0.44±0.020	0.50±0.021	**0.43±0.036**
CVD	1	0.53±0.081	0.70±0.089	**0.47±0.070**
	2	0.51±0.084	0.70±0.099	**0.48±0.081**
	3	**0.43±0.044**	0.45±0.045	0.50±0.087
PhysioNet	1	0.42±0.036	0.42±0.027	**0.34±0.027**
	2	0.42±0.033	0.47±0.025	**0.41±0.031**
	3	0.42±0.044	0.50±0.031	**0.42±0.024**

representative synthetic data. Such a finding supports the notion that encouraging diversity can be advantageous.

3.5.2 *Effect of Data Augmentation*

Augmentation methods. To compare the performance of the various data augmentation methods, we illustrate, in Fig. 3.5, the average change in the AUROC exhibited by a model with data augmentation to that exhibited by the same model without data augmentation. Specifically, the latter is a model exposed to a class-balanced subsample of the data, as we found this setup to generate a more competitive baseline.

First, we find that our family of cGAN-based data augmentation methods can improve the generalization performance of models by up to 29%. This is evident by looking at the results on the PhysioNet dataset. Second, we observe that the ranking of the three GAN-based methods are consistent across the four datasets, with cGAN+DS outperforming the others ($p < 0.05$). Such consistency is promising and is indicative of the robustness of these models. We hypothesize that the relatively poorer results of the remaining GANs by noting the potential limitations of artificially inducing interclass diversity when it may originally be present only to a small extent.

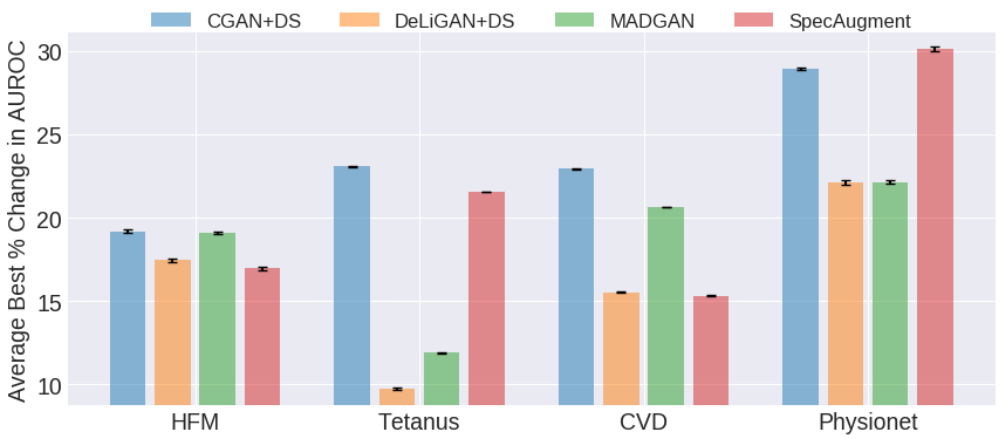


Figure 3.5: **Average best percent change in AUROC for models employing various data augmentation methods relative to those without any data augmentation.** Results are averaged across ten classification models and error bars represent one standard error from the mean. We show that data augmentation via CGAN+DS can lead to improvement in performance by up to 29%.

Furthermore, for three of the four datasets (HFM, Tetanus, and CVD), our GAN-based data augmentation outperforms that of SpecAugment in a statistically significant manner ($p < 0.05$). On the PhysioNet dataset, the difference between SpecAugment and cGAN+DS is not statistically significant. We attribute the strong performance of the GANs to their ability to generate representative and sufficiently diverse synthetic data. When performance is relatively worse than SpecAugment, as in the case of PhysioNet, we attribute this to the high degree of noise present within the dataset and also to the inability of the GANs to generate realistic datapoints (see Appendix A.1.1). Lastly, cGAN+DS and MADGAN appear to produce more consistent outcomes across datasets. This increased reliability may be a positive trait for practitioners.

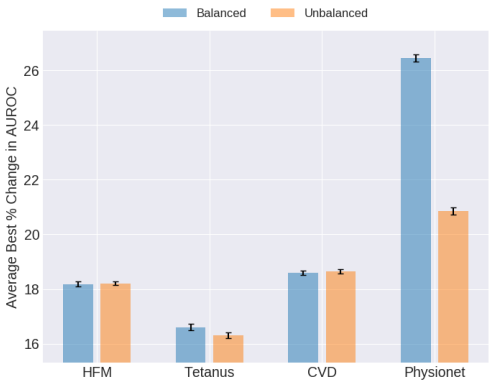
Training set imbalance. For all datasets except PhysioNet, we find that there is no significant difference in the results generated by balanced and unbalanced training sets (see Fig. 3.6a). This could be explained by certain synthetic classes being less diverse and informative than others, a finding supported by the intra-class diversity

plots. Therefore, more samples from only that class would be needed to improve performance.

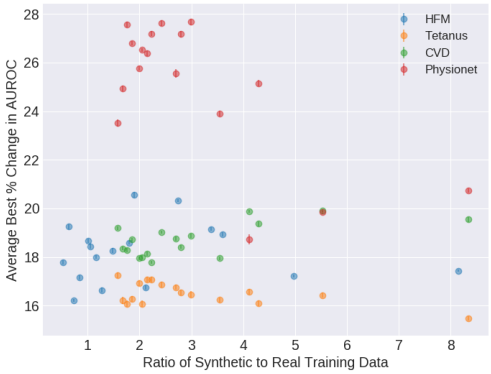
Ratio of synthetic to real data. To compare the significance of results stemming from different settings corresponding to various ratios, we use an ANOVA (parametric) and a Kruskal-Wallis (non-parametric) test. We observe that there is no significant difference between the results generated by a variety of synthetic to real data ratios (see Fig. 3.6b). This implies that the utility of the synthetic data is limited, at least for the range of ratios chosen. The improvement in classification performance, however, indicates that only a small amount of synthetic data can have a strong positive impact. Such a finding was most prominent for the PhysioNet dataset.

Class imbalance. On the HFM dataset, we observe that the classification improvement caused by introducing an imbalance in class 1 significantly outperforms ($p < 0.05$) that when imbalances are introduced in other classes (see Fig. 3.6c). Anticipating such a potential outcome, based on work by Mariani et al. (2018), the cGANs were trained with a balanced dataset to avoid class favouritism. Nevertheless, this effect is still observed and can be partly explained by the relatively strong Class 1 MMD_c values relative to the others (see Table 3.2). This phenomenon, however, is not observed with the other datasets.

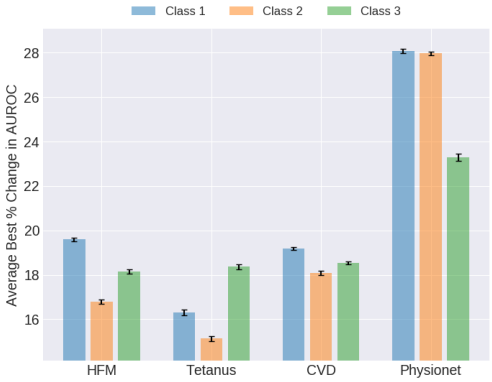
Data augmentation, although sometimes beneficial, can also be detrimental. To better understand the potential improvement *and* worsening of classification performance due to data augmentation, we illustrate our novel synthetic generalization curve in Fig. 3.7. Analagous to an ROC curve, the higher it is, the better the outcome. Moreover, increased mass when $\varepsilon < 0$ is indicative of classification improvement relative to the chosen baseline. For instance, the black dot indicates that when using synthetic data generated by MADGAN to augment the dataset, 40% of the classification models, on average, perform equivalent to or better than $1.10\times$ the baseline performance. We also observe that all methods are upper-bounded by the MADGAN method, indicating



(a) Balance vs. Imbalance



(b) Ratio of Synthetic to Real



(c) Class Imbalance

Figure 3.6: **Effect on performance of data augmentation policies.** Policies include those in which (a) classes are balanced or imbalanced, (b) the ratio of synthetic data to real data is varied, and (c) imbalance is purposefully introduced to distinct classes. Error bars represent one standard deviation from the mean.

the latter's superiority. The synthetic generalization curves for the remaining datasets can be found in Appendix A.1.1.

Building on the analogy to the ROC, we present the AUSGC values in Table 3.4. Given the range of values chosen for epsilon, the closer the AUSGC is to 1, the better the conditional GAN is in improving classification. Moreover, the smaller the standard deviation, the more consistent the conitional GAN is across the chosen augmentation policies. In other words, it is not producing highly varying behaviour. Ultimately, no statistical difference was found between the AUSGC values of the various augmentation methods. Nonetheless, we would like to emphasize that although we have used AUROC as the comparative performance metric in (3.13), this

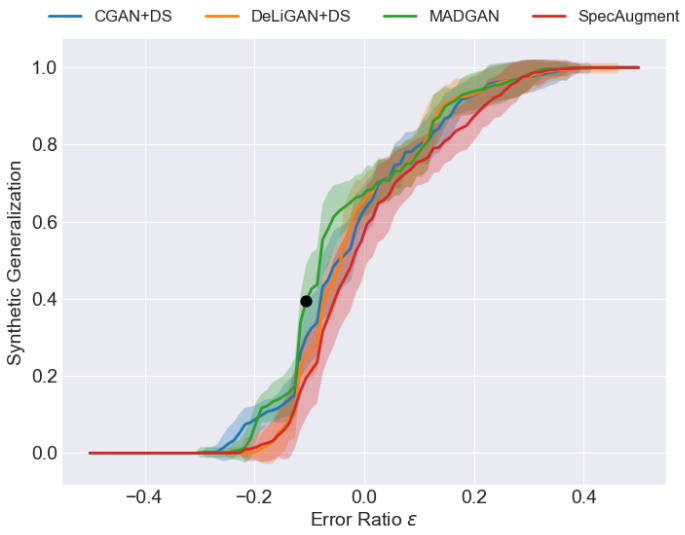


Figure 3.7: **Average synthetic generalization curve for each augmentation method when evaluated on $\mathcal{D}_3$.** The average is performed across all 54 augmentation policies. The shaded area represents one standard deviation from the mean. To interpret these curves, we direct your attention to the black dot. At this point, 40% of the machine learning models exploiting synthetic data generated by MADGAN perform 10% better than their counterparts which do not exploit synthetic data.

Table 3.4: **AUSGC averaged across all 54 augmentation policies.** We find that the methods do not produce results that are statistically significantly different.

Dataset	cGAN+DS	DeLiGAN+DS	MADGAN	SpecAugment
HFM	0.511±0.014	0.521±0.014	**0.525±0.018**	0.516±0.014
Tetanus	0.522±0.011	0.509±0.012	0.510±0.010	**0.540±0.010**
CVD	0.521±0.014	0.512±0.015	**0.534±0.016**	0.490±0.021
PhysioNet	0.521±0.013	0.517±0.012	0.500±0.012	**0.581±0.027**

curve is inherently *metric agnostic*; i.e., it can be used with any performance metric.

This allows researchers to choose their metric of interest based on the task at hand.

Part II

DOING MORE WITH LESS

In Part I, we designed clinical deep learning algorithms with the goal of 'doing some with less'. Our environment was characterized by data paucity and label imbalance. In Part II, we relax these characterizations ever so slightly and increase the amount of resources available for algorithms. As a reminder, these resources can include the quantity and quality of input data and labels, in addition to supervision provided by physicians. This paradigm is outlined in Fig. 3.8.

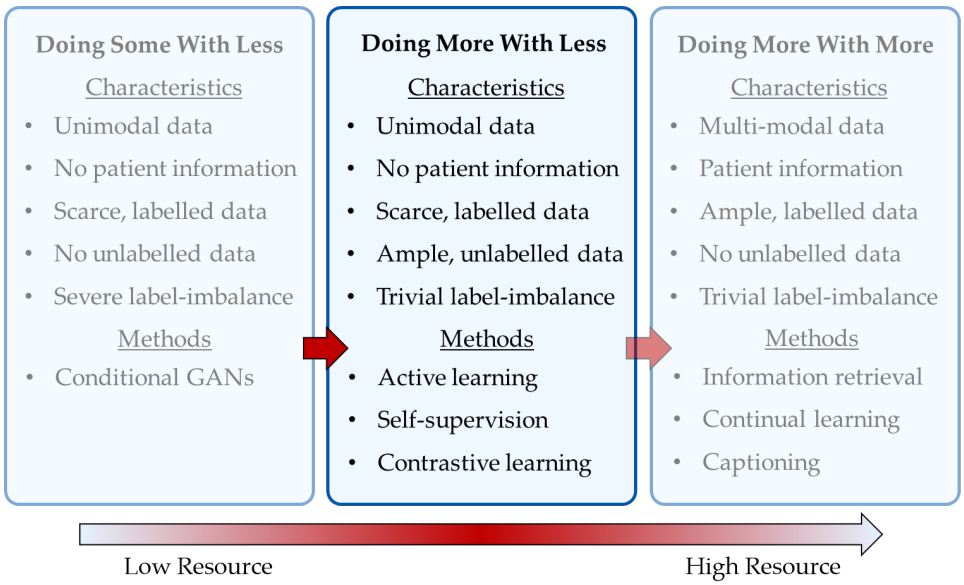


Figure 3.8: **Part II of the thesis focuses on the paradigm of 'doing more with less'.** In this paradigm, which is characterized by scarce, labelled data and abundant, unlabelled data, we propose to leverage active learning, self-supervision, and contrastive learning.

Throughout Part II, our environment is characterized by abundant, unlabelled and scarce, labelled data. This is an increasingly common scenario in healthcare where the rate of clinical data generation far exceeds the rate with which such data can be labelled by expert physicians. We also assume that algorithms have the opportunity to exploit both labelled and unlabelled data at their disposal. Our final assumption is that physicians are available *during* the algorithmic learning process. In light of these assumptions, we look to design clinical deep learning algorithms capable of 'doing more with less'; less absolute data, fewer labels, and less guidance throughout the learning process.

4

ACTIVE LEARNING FOR CARDIAC ARRHYTHMIA DIAGNOSIS

Action is the foundational key to all success

— Pablo Picasso

DATA are an asset that can be difficult to acquire. Insufficient medical infrastructure, patient privacy constraints, and the low inter-operability of medical records are some of the reasons behind this difficulty. Procuring ground-truth annotations of such data is an equally challenging feat. This typically requires the manual input of highly-skilled physicians which detracts their attention away from patients and can exacerbate their burnout. To minimize these effects, a subset of the data can be provided to physicians for annotation. However, this chosen subset may be sub-optimal for a network to learn from. As such, this manual annotation process is deemed inefficient and wasteful. Based on this observation, we are interested in tackling the following question.

Research Question

How can we design clinical algorithms that exploit abundant, unlabelled data and limited, labelled data while minimizing the labelling burden placed on physicians?

In this chapter, we address the outlined research question in the context of diagnosing cardiac arrhythmias based on the electrocardiogram. To achieve this, we exploit pool-based active learning (McCallumzy and Nigamy, 1998) and modify two of its three components; the acquisition of unlabelled instances, and the labelling of such instances by an oracle. Our contributions are multi-fold. First, we propose an

active learning framework that involves perturbing instances and observing changes in the corresponding output distribution. This setup, which we refer to as Monte Carlo Perturbations (MCP), is flexible enough to be used alongside existing off-the-shelf acquisition functions. We also show that MCP outperforms the state-of-the-art approach, Monte Carlo Dropout (MCD), in several settings. Second, we take inspiration from consistency training and propose an active learning framework that involves perturbing *both* instances and parameters and observing changes in the output distribution. We refer to this setup as Bayesian Active Learning by Consistency (BALC). To move away from uncertainty-based acquisition function, we introduce two *consistency-based* acquisition functions, BALC_{KLD} and BALC_{JSD}, and show that they can outperform state-of-the-art acquisition functions such as BALD. Third, we note that existing acquisition functions are static; they determine the informativeness of an instance at a single snapshot in time. Instead, we propose a simple modification that can be applied to any acquisition function that involves *tracking* its value over time (epochs). We show that acquisition functions with such temporal information can outperform their static counterparts. Lastly, we take inspiration from selective classification and propose a dynamic oracle selection framework, SoQal, which, for each acquired unlabelled instance, determines whether to request a label from an oracle or to provide a pseudo-label instead. We show that SoQal outperforms state-of-the-art selective classification methods and reduces the labelling burden placed on physicians.

4.1 RELATED WORK

Active learning (AL) for healthcare. Active learning, a formal description of which is relegated to Section 4.2, is a subfield of semi-supervised learning in which both labelled and unlabelled data are exploited in order to improve learning efficiency. An extensive review of such methods was conducted by Settles (2009). In the healthcare domain, Gong et al. (2019) propose to acquire instances from an electronic health

record (EHR) database using a Bayesian deep latent Gaussian model to improve mortality prediction. Smailagic et al. (2018, 2019) acquire unlabelled medical images by measuring their distance in a latent space to images in the training set. The work of Wang et al. (2019a) is similar to ours in that they focus on the electrocardiogram (ECG). Gal et al. (2017) adopt BALD (Houlsby et al., 2011) in the context of Monte Carlo Dropout to acquire instances that maximize the Jensen-Shannon divergence (JSD) across MC samples.

Oracles in active learning. Previous work attempts to learn from multiple or imperfect oracles (Dekel et al., 2012; Zhang and Chaudhuri, 2015; Sinha et al., 2019). For example, Urner et al. (2012) propose choosing the oracle that should label a particular instance. Unlike our approach, they do not explore independence from an oracle. Yan et al. (2016) consider oracle abstention in an active learning setting. Instead, we place the decision of abstention under the control of the learner. To the best of our knowledge, previous work, in contrast to ours, has assumed the existence of an oracle and has not explored a dynamic oracle selection strategy.

Consistency training. Consistency training in the context of semi-supervised learning helps enforce the smoothness assumption (Zhu, 2005). For example, Interpolation Consistency Training Verma et al. (2019) penalizes networks for not generating a linear combination of outputs in response to a linear combination of inputs. Similarly, Xie et al. (2019) penalizes networks for generating drastically different outputs in response to perturbed instances. In the process, networks learn perturbation-invariant representations. McCallumzy and Nigamy (1998) introduce an acquisition function that calculates the average Kullback-Leibler divergence, $\mathcal{D}_{KL}$, between the output of a network and the consensus output across all networks in an ensemble. Unlike ours, their approach does not exploit perturbations. Similar to our work is that of Gao et al. (2019) which incorporates into the objective function a consistency-loss based on the $\mathcal{D}_{KL}$. They actively acquire instances using the variance of the probability assigned to

each class by the network in response to perturbed versions of the same instance. In contrast, we design distinct consistency-based acquisition functions.

Selective classification. Selective classification imbues a network with the ability to abstain from making predictions. Some researchers have introduced the risk-coverage trade-off whereby the empirical risk of a model is inversely related to its rate of abstentions (Chow, 1970; El-Yaniv and Wiener, 2010). Wiener and El-Yaniv (2011) use a support vector machine (SVM) to rank and reject instances based on the degree of disagreement between hypotheses. In some frameworks, these are the same instances that active learning views as most informative. Cortes et al. (2016) outline an objective function that penalizes abstentions that are inappropriate and frequent. Most recently, Liu et al. (2019c) propose the gambler's loss to learn a selection function that determines whether instances are rejected. However, this approach is not implemented in the context of active learning. Most similar to our work is SelectiveNet (Geifman and El-Yaniv, 2019) where a multi-head architecture is used alongside an empirical selective risk objective function and a percentile threshold. However, their work assumes the presence of ground-truth labels and, therefore, does not extend to unlabelled instances.

4.2 BACKGROUND

4.2.1 *Active Learning*

Consider a learner, $f_\theta : \boldsymbol{x} \in \mathbb{R}^D \rightarrow \boldsymbol{v} \in \mathbb{R}^E$, parameterized by $\boldsymbol{\theta}$, that maps a D-dimensional instance, $\boldsymbol{x}$, to an E-dimensional representation, $\boldsymbol{v}$. Further consider a classification head, $p_\omega : \boldsymbol{v} \in \mathbb{R}^E \rightarrow \hat{y} \in \mathbb{R}^C$, that maps an E-dimensional representation, $\boldsymbol{v}$, to a C-dimensional output, $\hat{y}$, where C is the number of classes. After training on a pool of labelled data, $\mathcal{D}_L = \{\boldsymbol{X_L}, \boldsymbol{Y_L}\}$ for τ epochs, the learner is tasked with querying the unlabelled pool of data, $\boldsymbol{X_U}$, and acquiring the top b fraction of instances, $\boldsymbol{x_b} \sim \boldsymbol{X_U}$, that it deems to be most informative. The degree of informativeness of an instance, $\boldsymbol{x}$, is determined by an acquisition function, $a(\boldsymbol{x}) \in \mathbb{R}$, such

as Bayesian Active Learning by Disagreement (BALD) (Houlsby et al., 2011). These functions are typically used in conjunction with Monte Carlo Dropout (MCD) (Gal and Ghahramani, 2016) to identify instances that lie in the region of uncertainty, a region in which hypotheses (decision boundaries) disagree the most about instances (Fig. 4.1 left).

4.2.2 *Consistency Training*

Consider an unlabelled instance, $\boldsymbol{x} \sim \boldsymbol{X_U}$, that may exhibit an invariance to certain perturbations. An invariance suggests that upon applying a perturbation, ϵ, to the instance and generating its perturbed version, $\boldsymbol{x'}$, the latter will continue to reflect the same underlying physiological state of the original instance, $\boldsymbol{x}$. By passing $\boldsymbol{x}$ and $\boldsymbol{x'}$ through a network, we obtain the outputs, p_ω and p'_ω, respectively. In consistency training, the goal is to ensure that these outputs remain similar to one another despite the perturbation. In doing so, the network learns representations that are invariant to innocuous perturbations. In this work, we exploit the intuition behind consistency training to design acquisition functions, as outlined next.

4.3 METHODS

4.3.1 *Monte Carlo Perturbations*

Acquisition functions dependent upon perturbations applied to the network parameters, such as in MCD, can overlook, and thus fail to acquire, informative instances. To see this, consider the following example. Without loss of generality, let us assume an unlabelled instance is in proximity to some decision boundary and is classified by the network as belonging to some arbitrary disease class. Such proximity should deem the instance informative for the training process (Settles, 2009). In the MCD setting, perturbations are applied to parameters, generating T decision boundaries which in turn influence the network outputs, $\{p^i_\omega\}^T_{i=1}$, of the unlabelled instance. In Fig. 4.2 (red rectangle), we visualize the distribution of these outputs after having

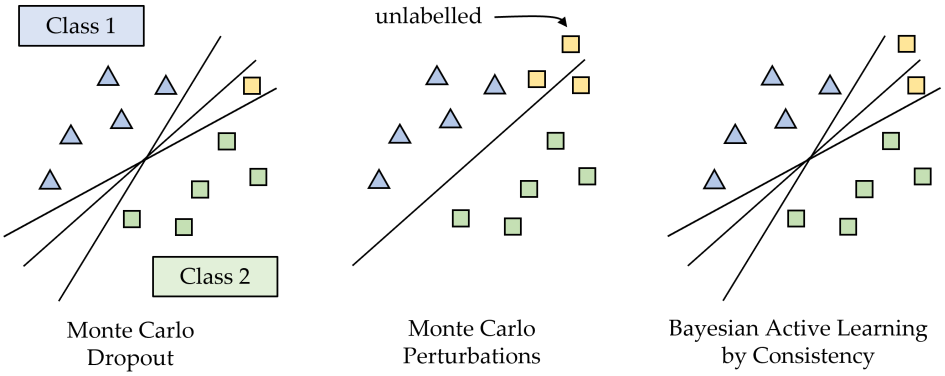


Figure 4.1: **Overview of perturbation frameworks used in conjunction with acquisition functions in active learning.** Labelled instances from Class 1 and 2 and unlabelled instances (yellow) alongside the version space of **(Left)** Monte Carlo Dropout where each MC sample is viewed as a distinct hypothesis (decision boundary), **(Centre)** Monte Carlo Perturbations where there is one hypothesis but several perturbations of the unlabelled instance, and **(Right)** Bayesian Active Learning by Consistency where there are several hypotheses in addition to the unlabelled instance and its perturbed counterpart.

applied $T = 3$ parameter perturbations. If the perturbations happen to be too small in magnitude, for example, then the network will exhibit a similar output distribution ($p_\omega^1 = p_\omega^2 = \cdots$) across the parameter perturbations. With acquisition functions detecting *changes* in the output distribution, this informative unlabelled instance (due to proximity to decision boundary) would thus be *erroneously* deemed uninformative.

One way to avoid missing these informative instances is by stochastically perturbing instances (instead of network parameters) and observing changes in the network outputs. We refer to this setup as Monte Carlo Perturbations (MCP). This results in a single decision boundary but multiple perturbed versions of the instance (see Fig. 4.1 centre). The intuition is that, for an instance in proximity to the decision boundary, its outputs will differ more significantly across perturbations than those of an instance farther away. Therefore, by quantifying these output changes, as is done with almost any acquisition function, we can identify informative instances for acquisition.

The main advantage of MCP over MCD is the increased control and interpretability of the applied perturbations; perturbations applied to instances are likely to be more

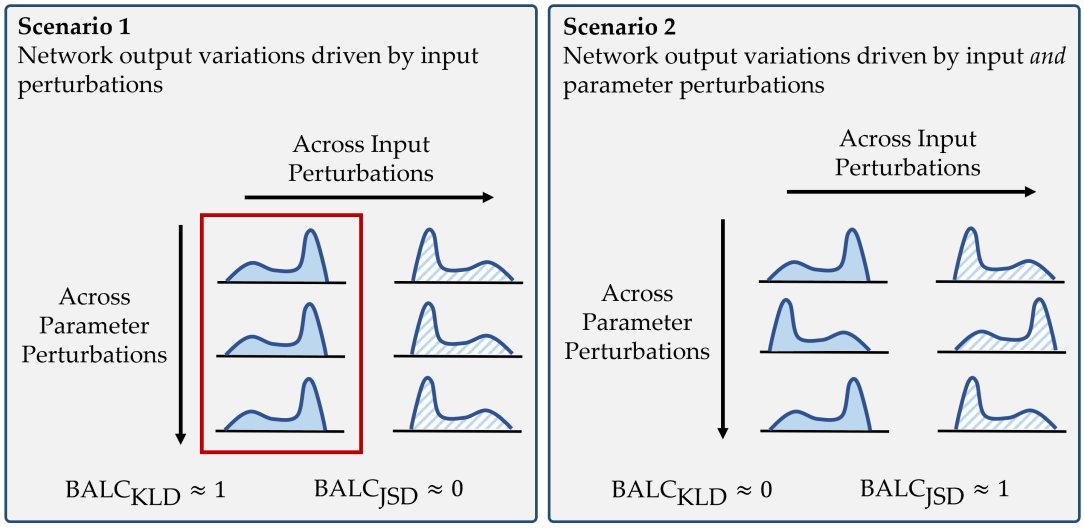


Figure 4.2: **Two scenarios demonstrating the effect of input and parameter perturbations on the behaviour of two proposed acquisition functions,** BALC_{KLD} **and** BALC_{JSD}**. (Scenario 1)** network output variations caused primarily by input perturbations. The red rectangle illustrates a potential limitation of MCD. Given that these parameter perturbations do not result in network output variations and MCD is dependent upon said perturbations, unlabelled instances can be erroneously deemed uninformative. **(Scenario 2)** network output variations caused by both input and parameter perturbations. We show that while BALC_{KLD} is likely to acquire instances due to input perturbations, BALC_{JSD} considers both input and parameter perturbations when performing acquisitions.

understandable than those applied to parameters. We derive MCP more formally below for use with BALD.

$$\begin{aligned}\text{BALD}_{\text{MCP}} &= \text{JSD}(p^1, p^2, \ldots, p^T) \\ &= \mathbb{H}(p(y|\boldsymbol{x})) - \mathbb{E}_{p(\boldsymbol{x}'|D_{train})}\left[\mathbb{H}(p(y|\boldsymbol{x}, \boldsymbol{x}'))\right]\end{aligned} \tag{4.1}$$

$$\begin{aligned}\mathbb{H}(p(y|x)) &= \mathbb{H}\left(\int p(y|\boldsymbol{x}')p(\boldsymbol{x}'|\boldsymbol{x})d\boldsymbol{x}'\right) \\ &= \mathbb{H}\left(\int p(y|\boldsymbol{x}')q_\phi(\boldsymbol{x}'|\boldsymbol{x})d\boldsymbol{x}'\right) \\ &\approx \mathbb{H}\left(\frac{1}{T}\sum_{t=1}^{T} p(y|\boldsymbol{x}_t')\right)\end{aligned}$$

$$\begin{aligned}\mathbb{E}_{p(\boldsymbol{x}'|D_{train})}\left[\mathbb{H}(p(y|\boldsymbol{x},\boldsymbol{x}'))\right] &= \mathbb{E}_{q_\phi(\boldsymbol{x}'|\boldsymbol{x})}\left[\mathbb{H}(p(y|\boldsymbol{x},\boldsymbol{x}'))\right] \\ &\approx \frac{1}{T}\sum_{t=1}^{T}\left[\mathbb{H}(p(y|\boldsymbol{x}'_t))\right] \\ &= \frac{1}{T}\sum_{t=1}^{T}\left[-\sum_{c=1}^{C} p(y=c|\boldsymbol{x}'_t)\log p(y=c|\boldsymbol{x}'_t)\right]\end{aligned}$$

where $\mathbb{H}$ represents entropy, $\boldsymbol{x}'$ represents the perturbed input, T is the number of Monte Carlo samples, and $\boldsymbol{x}'_t \sim q_\phi(\boldsymbol{x}'|\boldsymbol{x})$ is the t-th sample from some perturbation generator.

4.3.2 *Bayesian Active Learning by Consistency*

It could be argued that the same limitations experienced by MCD also extend to MCP. After all, both frameworks apply perturbations to either network parameters or instances, respectively. We acknowledge this potential limitation and thus propose a framework, entitled Bayesian Active Learning by Consistency (BALC), in which perturbation are applied to *both* network parameters and instances. Therefore, in this setting, we would have multiple decision boundaries and perturbed instances (see Fig. 4.1 right). BALC consists of three main steps: 1) we perturb an instance, $\boldsymbol{x} \in \mathbb{R}^D$, to generate $\boldsymbol{x}' \in \mathbb{R}^D$, 2) we perturb the network parameters, $\boldsymbol{\theta}$, to generate $\boldsymbol{\theta}'$, and 3) we pass both instances, $\boldsymbol{x}$ and $\boldsymbol{x}'$, through the *perturbed* network, generating outputs, $p(y|\boldsymbol{x},\boldsymbol{\theta}')$ and $p(y|\boldsymbol{x}',\boldsymbol{\theta}') \in \mathbb{R}^C$, respectively. Note that we drop the explicit dependence on $\boldsymbol{\omega}$ for clarity. We perform these steps for T stochastic parameter perturbations and generate two matrices of network outputs, $\boldsymbol{G}(\boldsymbol{x}), \boldsymbol{G}'(\boldsymbol{x}') \in \mathbb{R}^{T\times C}$, as shown in Fig. 4.3.

Recall that our goal is still to identify unlabelled instances that are in proximity to the decision boundary. To achieve this goal while exploiting $\boldsymbol{G}$ and $\boldsymbol{G}'$, we propose two consistency-based acquisition functions, BALC_{KLD} and BALC_{JSD}. These functions quantify the changes in the network outputs in response to perturbations applied to *both* network parameters and instances.

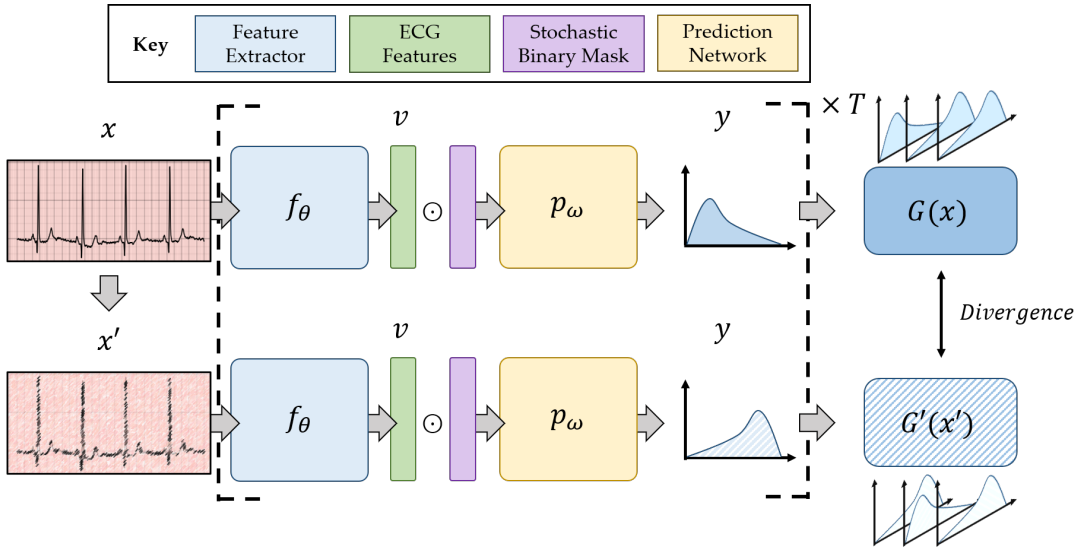


Figure 4.3: **Pipeline of consistency-based active-learning framework.** We perturb an instance, x, to generate, x'. We extract representations, v, from each of the original and perturbed instance. A stochastic dropout mask is applied to these representations before obtaining a probability distribution over the classes. This is repeated T times to generate a pair of matrices, $\boldsymbol{G}(x)$ and $\boldsymbol{G}'(x')$, whose divergence is calculated via BALC_{KLD} or BALC_{JSD}.

To calculate $\text{BALC}_{\text{KLD}}(x) \in \mathbb{R}$, we first empirically fit two C-dimensional Gaussian distributions, $\mathcal{N}(x)$ and $\mathcal{N}(x')$ to $\boldsymbol{G}$ and $\boldsymbol{G}'$, respectively. From this, we obtain the empirical mean vector, $\boldsymbol{\mu} = \frac{1}{T}\sum_i^T \boldsymbol{G}_i$ and covariance matrix, $\boldsymbol{\Sigma} = (\boldsymbol{G} - \boldsymbol{\mu})^T(\boldsymbol{G} - \boldsymbol{\mu})$, for each matrix, $\boldsymbol{G}$ and $\boldsymbol{G}'$. We then calculate the Kullback-Leibler divergence, D_{KL}, between $\mathcal{N}(x)$ and $\mathcal{N}(x')$.

$$\text{BALC}_{\text{KLD}}(x) = \mathcal{D}_{KL}(\mathcal{N}(x) \parallel \mathcal{N}(x')) \tag{4.2}$$

where $\mathcal{N}(x) = \mathcal{N}(\boldsymbol{\mu}(x), \boldsymbol{\Sigma}(x))$ and $\mathcal{N}(x') = \mathcal{N}(\boldsymbol{\mu}(x'), \boldsymbol{\Sigma}(x'))$.

We note that BALC_{KLD} is likely to detect changes in the network outputs due to perturbations applied to an instance. To see this, we return to Fig. 4.2 and present two distinct scenarios. In scenario 1, changes in network outputs are caused solely by instance perturbations. On the other hand, in scenario 2, changes in network outputs are caused by both instance and parameter perturbations. We show that $\text{BALC}_{\text{KLD}} \approx 1$ and 0 in these two scenarios, respectively. Since the higher the value of an acquisition function, the more informative an instance is, these scenarios suggest that BALC_{KLD}

has a preference for instance perturbations. In order to detect changes in the network outputs due to *both* instance and parameter perturbations, we introduce another consistency-based acquisition function, BALC_{JSD}.

$\text{BALC}_{\text{JSD}}(\boldsymbol{x}) \in \mathbb{R}$ comprises the difference of two terms. The first term, Ⓐ, calculates the D_{KL} of network outputs due to a single instance perturbation and averages this across T parameter perturbations. The second term, Ⓑ, averages the network outputs, across parameter perturbations, independently for the original and perturbed instance before calculating the D_{KL} of the resulting mean outputs. To see BALC_{JSD} in action, please refer to the illustrative scenarios in Fig 4.2.

$$\text{BALC}_{\text{JSD}}(\boldsymbol{x}) = \overbrace{\frac{1}{T}\sum_{i=1}^{T}\left[\mathcal{D}_{KL}(\boldsymbol{G_i}(\boldsymbol{x}) \parallel \boldsymbol{G_i'}(\boldsymbol{x'}))\right]}^{\text{Ⓐ across parameter perturbations}} - \overbrace{\mathcal{D}_{KL}(\frac{1}{T}\sum_{i=1}^{T}\boldsymbol{G_i}(\boldsymbol{x}) \parallel \frac{1}{T}\sum_{i=1}^{T}\boldsymbol{G_i'}(\boldsymbol{x'}))}^{\text{Ⓑ across input perturbations}} \tag{4.3}$$

We can rewrite (4.3) to more explicitly illustrate the dependence of the outputs on the perturbed network parameters and inputs.

$$\text{BALC}_{\text{JSD}}(\boldsymbol{x}) = \mathbb{E}_{p(\boldsymbol{\theta}|D_{train})}\left[\mathcal{D}_{KL}(p(y|\boldsymbol{x},\boldsymbol{\theta}) \parallel p(y|\boldsymbol{x'},\boldsymbol{\theta}))\right] - \mathcal{D}_{KL}(p(y|\boldsymbol{x}) \parallel p(y|\boldsymbol{x'})) \tag{4.4}$$

$$\begin{aligned}
\mathbb{E}_{p(\boldsymbol{\theta}|D_{train})}\left[\mathcal{D}_{KL}(p(y|\boldsymbol{x},\boldsymbol{\theta}) \parallel p(y|\boldsymbol{x'},\boldsymbol{\theta}))\right] &= \mathbb{E}_{q(\boldsymbol{\theta})}\left[\mathcal{D}_{KL}(p(y|\boldsymbol{x},\boldsymbol{\theta}) \parallel p(y|\boldsymbol{x'},\boldsymbol{\theta}))\right] \\
&\approx \frac{1}{T}\sum_{t=1}^{T}\left[\mathcal{D}_{KL}(p(y|\boldsymbol{x},\boldsymbol{\theta_t'}) \parallel p(y|\boldsymbol{x'},\boldsymbol{\theta_t'}))\right] \\
&= \frac{1}{T}\sum_{t=1}^{T}\left[\sum_{c=1}^{C} p(y=c|\boldsymbol{x},\boldsymbol{\theta_t'}) \log \frac{p(y=c|\boldsymbol{x},\boldsymbol{\theta_t'})}{p(y=c|\boldsymbol{x'},\boldsymbol{\theta_t'})}\right]
\end{aligned}$$

$$
\begin{aligned}
\mathcal{D}_{KL}(p(y|\boldsymbol{x}) \parallel p(y|\boldsymbol{x}')) &= \mathcal{D}_{KL}\left(\int p(y|\boldsymbol{\theta},\boldsymbol{x})p(\boldsymbol{\theta})d\boldsymbol{\theta} \,\middle\|\, \int p(y|\boldsymbol{\theta},\boldsymbol{x}')p(\boldsymbol{\theta})d\boldsymbol{\theta}\right) \\
&= \mathcal{D}_{KL}\left(\int p(y|\boldsymbol{\theta},\boldsymbol{x})q(\boldsymbol{\theta})d\boldsymbol{\theta} \,\middle\|\, \int p(y|\boldsymbol{\theta},\boldsymbol{x}')q(\boldsymbol{\theta})d\boldsymbol{\theta}\right) \\
&\approx \mathcal{D}_{KL}\left(\frac{1}{T}\sum_{t=1}^{T} p(y|\boldsymbol{\theta}'_t,\boldsymbol{x}) \,\middle\|\, \frac{1}{T}\sum_{t=1}^{T} p(y|\boldsymbol{\theta}'_t,\boldsymbol{x}')\right) \\
&= \frac{1}{C}\sum_{c=1}^{C}\left[\frac{1}{T}\sum_{t=1}^{T} p(y=c|\boldsymbol{\theta}'_t,\boldsymbol{x}) \log \frac{\frac{1}{T}\sum_{t=1}^{T} p(y=c|\boldsymbol{\theta}'_t,\boldsymbol{x})}{\frac{1}{T}\sum_{t=1}^{T} p(y=c|\boldsymbol{\theta}'_t,\boldsymbol{x}')}\right]
\end{aligned}
$$

where the integral is approximated by T Monte Carlo samples, $\boldsymbol{\theta}'_t \sim q(\boldsymbol{\theta})$ represents the parameters sampled from the Monte Carlo distribution, and C represents the number of classes in the task formulation.

4.3.3 *Tracked Acquisition Function*

So far, we have discussed active learning frameworks that, similar to those in the literature, quantify the informativeness of an unlabelled instance at a *single* snapshot in time (e.g., at a particular epoch, τ). This static setup, however, faces two limitations. First, it depends on an appropriate choice of epoch for the acquisition of instances, which is non-trivial to identify a priori. For example, an acquisition function can be of little value if calculated when network parameters have yet to be updated sufficiently. Second, the diversity and number of hypotheses, obtained via parameter perturbations, can be limited by this single-epoch view. This is detrimental given the established benefit of eliminating unsuitable hypotheses at a greater rate ($\downarrow$ version space) (Cohn et al., 1994).

To overcome the limitations of a static acquisition function, we propose to *track* such a function over time (e.g., epochs) before deploying it for the acquisition of instances. In the process, we become less dependent on the choice of epoch for acquisition and are likely to increase the diversity of hypotheses at our disposal. We note that this approach is flexible enough to be used alongside existing acquisition functions. Formally, consider an acquisition function, $\alpha(\boldsymbol{x})$, which would traditionally be calculated once at epoch, τ, for each instance, $\boldsymbol{x}$. In our formulation, we calculate

$a(\boldsymbol{x}, t)$ at each epoch, $t \in [1, \tau]$. At epoch, τ, which we refer to as the acquisition epoch, we calculate the area under the tracked acquisition function, AUTAF(x) $\in \mathbb{R}$.

$$\text{AUTAF}(\boldsymbol{x}) = \int_1^{\tau} a(\boldsymbol{x}, t)dt \approx \sum_{t=1}^{\tau} \left(\frac{a(\boldsymbol{x}, t + \Delta t) + a(\boldsymbol{x}, t)}{2} \right) \Delta t \tag{4.5}$$

where the integral is approximated using the trapezoidal rule and Δt is the interval between epochs at which acquisition values are calculated.

4.3.4 *Selective Oracle Questioning*

Up until this point, we have exclusively discussed methods that target the first step of active learning, that of acquisition of unlabelled instances. In this section, we direct our attention to the second stage of active learning, that of annotation of unlabelled instances. Our goal is to minimize the labelling burden that is placed on an oracle (e.g., physician). To do so, we design a framework that, given an acquired unlabelled instance, dynamically determines whether to request a label from an oracle or to pseudo-label the instance instead. This framework, which we refer to as selective oracle questioning in active learning, SoQal, is outlined next.

Oracle selection network. Let us consider a learner, $g_\phi : \boldsymbol{v} \in \mathbb{R}^E \rightarrow r \in [0, 1]$, parameterized by $\boldsymbol{\phi}$, that maps an E-dimensional representation, $\boldsymbol{v}$, to a scalar, r, as shown in Fig. 4.4. We refer to this learner as an oracle selection network due to its involvement in the selection of an oracle for annotation, as will become apparent.

Learning a proxy for misclassifications. The intuition underlying our oracle selection framework is as follows. Given an instance, a network should be more likely to request a label from an oracle than to pseudo-label it if the network constantly misclassifies this instance. In short, if the network knows it will classify an instance incorrectly, then it should ask for help. Although, in principle, this idea has merit, it assumes that the network is able to identify whether an instance will be misclassified. While this is possible with *labelled* data (due to presence of ground-truth), it is *not*

possible with *unlabelled* data, the prime focus of active learning. As such, we need a way to identify whether unlabelled instances would have been misclassified if they were to be pseudo-labelled by the network. In other words, we need a reliable proxy for misclassifications.

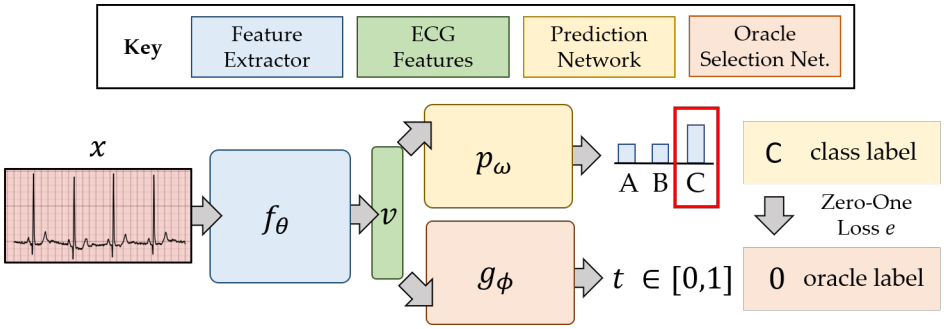


Figure 4.4: **Pipeline of consistency-based active-learning framework with a dynamic oracle selection network.** An instance, $\boldsymbol{x}$, is provided as input to the feature extractor, f_θ. The representation, v, is passed through g_ϕ to generate task predictions, and through g_ϕ to help determine whether a label is requested from an oracle. The zero-one loss of g_ϕ acts as the ground-truth label for r where $r \approx 1$ indicates a hard-to-classify instance that would benefit from an oracle label. Otherwise, the instance is pseudo-labelled by taking the argmax of the outputs of the prediction network.

To begin designing this proxy, we turn our attention to the oracle selection network and its scalar output, r. As we explain next, we will learn the parameters of $\boldsymbol{\phi}$ such that higher values of r correspond to incorrect network predictions. To achieve this, the output, r, requires a ground-truth label from which to learn. We choose this ground-truth label as the zero-one loss, e, incurred by the prediction network, p_ω, on an instance, $\boldsymbol{x}$. Intuitively, when $e = 1$, the network should request a label from an oracle. Note that this ground-truth label, e, is only available while training on *labelled* data. We explain how to exploit r on acquired *unlabelled* instances later. In summary, while training on labelled data, we optimize an objective function comprising a categorical cross-entropy loss for the prediction network, $\mathcal{L}_{CPL}$, with ground-truth class labels, c, and a weighted binary cross-entropy loss for the oracle selection

network, $\mathcal{L}_{OSL}$. In the process, we learn the parameters, $\boldsymbol{\theta}$, $\boldsymbol{\omega}$, and $\boldsymbol{\phi}$ in an end-to-end manner.

$$\mathcal{L} = \overbrace{\mathcal{L}_{CPL}}^{\text{class prediction loss}} + \overbrace{\mathcal{L}_{OSL}}^{\text{oracle selection loss}} \tag{4.6}$$

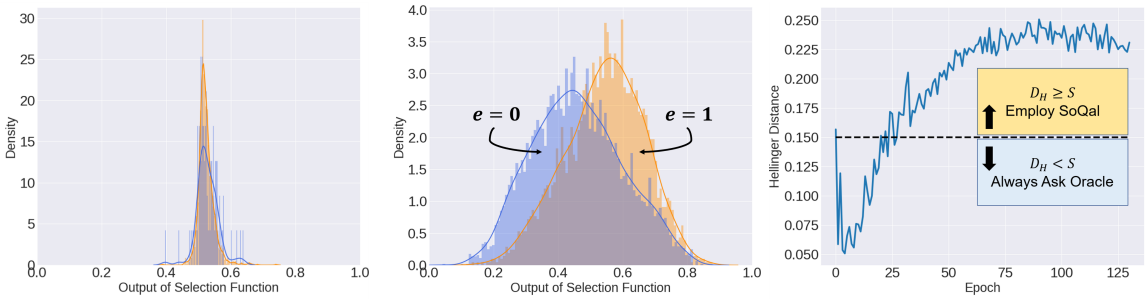
$$\mathcal{L}_{CPL} = -\sum_{i=1}^{B} \log\left(p_{\boldsymbol{\omega}}(y_i = c|\boldsymbol{x}_i, \boldsymbol{\phi})\right)$$

$$\mathcal{L}_{OSL} = -\sum_{i=1}^{B} \beta e_i \log\left(g_{\boldsymbol{\phi}}(r|\boldsymbol{x}_i)\right) - (1 - e_i) \log\left(1 - g_{\boldsymbol{\phi}}(r|\boldsymbol{x}_i)\right)$$

Weighted oracle selection loss. During the early stages of training on *labelled* data, a network struggles to classify instances correctly. In our framework, this implies that we will encounter more terms with $e = 1$ (misclassification) than those with $e = 0$ (correct classification). However, the opposite is true as training progresses and the network becomes more adept at classifying instances. Therefore, regardless of the stage of training (early vs. late), there will be an imbalance in the ground-truth labels, e, provided to the oracle selection network. This imbalance is analogous to traditional label imbalance and sends a strong supervisory signal to g_ϕ through the oracle selection loss, $\mathcal{L}_{OSL}$. Not accounting for this label imbalance would lead to systematically higher r values during the early stage of training and systematically lower r values during the late stages of training. This can be problematic when depending on r as a reliable proxy for misclassifications. To overcome this obstacle, we introduce a dynamic weight, $\beta = \frac{\sum \mathbb{1}(e=0)}{\sum \mathbb{1}(e=1)}$, where $\mathbb{1}$ is the indicator function. As training progress, $\beta < 1 \rightarrow \beta > 1$, as the ratio of correctly classified ($e = 0$) to misclassified ($e = 1$) instances within a mini-batch changes. Next, we outline how to make decisions about whether, for an unlabelled instance, to request a label from an oracle by using the proxy, r.

Making decisions with proxy. We exploit r for the binary decision of whether to request a label from an oracle or to generate a pseudo-label instead. Since $r \in [0, 1]$, we can do this by applying a simple threshold at 0.5. However, this threshold may

be sub-optimal, particularly if the values of r are not calibrated. Therefore, when designing a robust strategy, we must account for the *distribution* of the r values that corresponds to each of the two decisions and the separability of such distributions. These distributions are illustrated in Figs. 4.5a and 4.5b during the early and late stages of training, respectively. They are also colour-coded based on whether the r values correspond to correctly-classified ($e = 0$) or misclassified ($e = 1$) *labelled* training instances. We can see that correctly-classified instances tend to have lower r values, as expected.



(a) Early stage of training (b) Late stage of training (c) Hellinger distance

Figure 4.5: **Components of the selective oracle questioning framework.** Distribution of the outputs, r, of g_ϕ colour-coded based on the zero-one classification error during the (a) early and (b) late stages of training. (c) the Hellinger distance, $\mathcal{D}_H$, between the two distributions of r increases during training as they become more separable.

We now outline the steps involved in making the binary decision. First, after each training epoch, t, we fit the r values (generated by the *labelled* data) in Fig. 4.5b to two unimodal Gaussian distributions, $\mathcal{N}_0(\mu_0, \sigma_0^2)$ and $\mathcal{N}_1(\mu_1, \sigma_1^2)$ corresponding to $e = 0$ and $e = 1$, respectively. If these distributions exhibit low separability across e, then this suggests that the oracle selection network has yet to learn to distinguish between correctly-classified and misclassified instances, and is thus generating an unreliable proxy, r. To quantify this separability, we use a metric to which a threshold can be easily applied, such as the Hellinger distance, $\mathcal{D}_H \in [0, 1]$. The progression of this distance during training is shown in Fig. 4.5c. We can see that the separability of the two distributions increases as a function of training epoch, a positive sign indicating that the oracle selection network is becoming more reliable. Low separability which is

captured by $\mathcal{D}_H < S$, where S is a user-defined threshold, reflects an unreliable proxy. In this case, a conservative labelling strategy that defers to the oracle is deployed. We note that the value of S can be altered depending on the relative level of trust one has in the network and oracle. We also explore how S affects performance in Sec. 4.5.5.

Pseudo-labelling. When $\mathcal{D}_H \geq S$, this suggests that the proxy is reliable enough to make decisions. As such, for each acquired *unlabelled* instance, $\boldsymbol{x}_u \sim \boldsymbol{X_U}$, we obtain $r_u = g_{\boldsymbol{\phi}}(\boldsymbol{x}_u)$ and compare the values of $\mathcal{N}_0$ and $\mathcal{N}_1$ when evaluated at r_u. If $\mathcal{N}_1 > \mathcal{N}_0$, then the value of r is too high (indicating a potential network misclassification), and a label is requested from an oracle. Otherwise, the instance is pseudo-labelled via $\text{argmax}_c\, p_\omega(y = c|\boldsymbol{x})$. The probability of requesting a label from an oracle is denoted by $p(\text{A})$.

$$p(\text{A}) = \begin{cases} 1, & \mathcal{D}_H < S \\ 1, & \mathcal{N}_1 > \mathcal{N}_0 \text{ and } \mathcal{D}_H \geq S \\ 0, & \text{otherwise} \end{cases} \tag{4.7}$$

where $\mathcal{N}_1 = \mathcal{N}\left(r = r_u|\mu_1, \sigma_1^2, e = 1\right)$ and $\mathcal{N}_0 = \mathcal{N}\left(r = r_u|\mu_0, \sigma_0^2, e = 0\right)$. The entire active learning framework is shown in Algorithms 4.1 and 4.2 where the blue lines are the components associated with the (optional) tracking of acquisition functions. We also note that our oracle selection framework is independent of the acquisition of unlabelled instances. As such, it is flexible enough to be used alongside other acquisition functions.

Algorithm 4.1 Bayesian Active Learning by Consistency

Input: acquisition epochs τ, epoch interval Δt, labelled data $\mathcal{D}_L$, unlabelled data $\boldsymbol{X_U}$, network parameters $\boldsymbol{\theta}$ $\boldsymbol{\omega}$ $\boldsymbol{\phi}$, MC samples T, acquisition fraction b

while training **do**
  if epoch in Δt **then**
    for $\boldsymbol{x} \sim X_U$ **do**
      $\boldsymbol{x'} = \boldsymbol{x} + \boldsymbol{\epsilon}, \quad \epsilon \sim \mathcal{N}(0, \sigma^2)$
      for t-th MC sample in T **do**
        obtain $p(y|\boldsymbol{x}, \boldsymbol{\theta'_t})$ ▷ original input
        obtain $p(y|\boldsymbol{x'}, \boldsymbol{\theta'_t})$ ▷ perturbed input
      calculate $\alpha(\boldsymbol{x})$ using (4.3) or (4.2)
      $\alpha(\boldsymbol{x}, t) = \alpha$
  if epoch in τ **then** ▷ acquire unlabelled instances
    calculate α using (4.5)
    SortDescending(α)
    $\boldsymbol{x_b} \subset \boldsymbol{X_U}$
    $y_b = \text{SoQal}(\boldsymbol{x_b})$ ▷ selective oracle questioning
    $\boldsymbol{X_U} \in (\boldsymbol{X_U} \setminus (\boldsymbol{x_b}, y_b))$
    $\mathcal{D}_L \in (\mathcal{D}_L \cup (\boldsymbol{x_b}, y_b))$

Algorithm 4.2 SoQal

Input: unlabelled instances $\boldsymbol{x_b}$, Hellinger distance $\mathcal{D}_H$, Hellinger threshold S

for $\boldsymbol{x}_u \sim \boldsymbol{x_b}$ **do**
  $r_u = g_\phi(\boldsymbol{x}_u)$
  if $\mathcal{D}_H > S$ **then**
    calculate $p(\text{A})$ from (4.7)
    if $p(\text{A}) = 1$ **then**
      $y_b \subset \boldsymbol{Y_U}$ ▷ request label from physician
    else
      $y_b = \underset{c}{\text{argmax}}\ p_\omega(y = c|\boldsymbol{x}_u)$ ▷ pseudo-label

4.4 EXPERIMENTAL DESIGN

4.4.1 *Data and Pre-processing*

To evaluate our method, we leverage four different datasets, each of which consists of cardiac time-series waveforms alongside cardiac arrhythmia labels. We split each of the aforementioned waveforms into non-overlapping segments comprising 2500 samples. In Table 4.1, we present a summary of these datasets. For the PhysioNet 2015 dataset, we exclude segments with ground-truth annotations that have been

found to be false positives. This process mitigates the amount of label noise present in the dataset. As for the cardiology dataset, we exclude segments with ground-truth annotations of sinus bradycardia in attempt to remain faithful to the original use of the data by Hannun et al. (2019).

Table 4.1: **Summary of the datasets used for evaluation.** We also show additional pre-processing information. Please click on the dataset's name for more information.

Dataset	Abbreviation	Modality	Normalization	Exclusion Criteria
PhysioNet 2015	$\mathcal{D}_1$	PPG	✓	false positive cases
PhysioNet 2015	$\mathcal{D}_2$	ECG	✓	false positive cases
PhysioNet 2017	$\mathcal{D}_3$	ECG	✗	-
Cardiology	$\mathcal{D}_4$	ECG	✗	sinus bradycardia cases

We are also interested in exploring how our framework performs when exposed to varying amounts of labelled and unlabelled data during the training procedure. Therefore, when conducting our experiments, we outline the fraction, $F \in [0.1, 0.3, 0.5, 0.7, 0.9]$, of the entire training data which are labelled. For example, $F = 0.1$ implies that 10% of the data are labelled and 90% are unlabelled. In Table 4.2, we present the number of instances such fractions correspond to. We also provide an in-depth description of the hyperparameters used during training in Appendix B.1.2.

4.4.2 *Active Learning Scenarios*

In our experiments, we explore three distinct active learning scenarios characterized by the presence and quality of the oracle.

Scenario 1 - Without Oracle. Active learning frameworks typically assume the presence of an oracle. However, as we outlined earlier, oracles are not always available. This is particularly the case within low and high-resource clinical settings. To reflect this extreme setting, we evaluate the performance of our active learning framework *without* an oracle. This can be thought of as being analogous to semi-supervised learning, although the latter is beyond the scope of our work. In the next scenario, we relax this assumption of oracle absence.

Table 4.2: **Number of segments in the training, validation, and test splits corresponding to fractions,** $F = [0.1, 0.3, 0.5, 0.7, 0.9]$. The values in the brackets indicate the number of patients in those subsets. The splits were performed uniformly at random at the patient-level.

Dataset	Fraction F	Training Lab.	Training Unlab.	Validation	Test
$\mathcal{D}_1$	0.1	401 (18 patients)	4,233 (171)	1,124 (47)	1,435 (58)
	0.3	1,285 (55)	3,349 (134)		
	0.5	2,187 (92)	2,447 (97)		
	0.7	3,132 (129)	1,502 (60)		
	0.9	4,184 (166)	450 (23)		
$\mathcal{D}_2$	0.1	401 (18)	4,233 (171)	1,124 (47)	1,435 (58)
	0.3	1,285 (55)	3,349 (134)		
	0.5	2,187 (92)	2,447 (97)		
	0.7	3,132 (129)	1,502 (60)		
	0.9	4,184 (166)	450 (23)		
$\mathcal{D}_3$	0.1	1,776 (545)	16,479 (4,914)	4,582 (1,364)	5,824 (1705)
	0.3	5,399 (1636)	12,856 (3,823)		
	0.5	9,054 (2727)	9,201 (2,732)		
	0.7	12,733 (3818)	5,522 (1,641)		
	0.9	16,365 (4909)	1,890 (550)		
$\mathcal{D}_4$	0.1	452 (20)	4,110 (181)	1,131 (50)	1,386 (62)
	0.3	1,368 (60)	3,194 (141)		
	0.5	2,280 (101)	2,282 (100)		
	0.7	3,200 (140)	1,362 (61)		
	0.9	4,079 (180)	483 (21)		

Scenario 2 - Noise-free Oracle. In this scenario, we relax the assumption that physicians are unavailable for the provision of annotations. Instead, we assume that a physician is available, and focus on alleviating the labelling burden that is placed on this oracle. We also assume that the oracle can provide accurate labels, i.e., those that are noise-free. This is analogous to the traditional active learning scenario.

Scenario 3 - Noisy Oracle. In this scenario, we assume that physicians are either ill-trained, fatigued, or unable to diagnose a medical case due to its difficulty. We simulate this setting by introducing two types of label noise. We stochastically flip the ground-truth label (unseen by the network) of each unlabelled instance to 1) any other label randomly **(Random)**, or 2) its nearest neighbour, in a smaller dimensional subspace, from a *different* class **(Nearest Neighbour)**. Whereas the first form of noise is extreme, the latter is more realistic as it may reflect uncertain physician diagnoses. Furthermore, to simulate noise of different magnitude, we inject noise with probability $\gamma = [0.05, 0.1, 0.2, 0.4, 0.8]$.

4.4.3 *Baseline Methods*

Acquisition Functions. We compare our proposed active learning frameworks and acquisition functions to the state-of-the-art acquisition functions used in conjunction with MCD. These include *Var Ratio*, *Entropy*, and *BALD*, definitions of which can be found in Appendix B.1.1. To determine whether active learning adds value in our setting, we also compare to the scenario in which active learning is not employed *(No AL)*.

Selective Oracle Questioning. We experiment with baselines that exhibit varying degrees of oracle dependence. **No Oracle** is a scenario in which 0% of the labels that correspond to unlabelled instances are oracle-based and are instead pseudo-labelled by taking the argmax of the network predictions. *Epsilon Greedy* is a stochastic strategy (Watkins, 1989) that we adapt to exponentially decay the reliance of the network on an oracle as a function of the number of acquisition epochs. *Entropy Response* assumes that high entropy predictions generated by a network are indicative of instances that the network is unsure of. Therefore, we introduce a threshold, S_{Entropy}, such that if it is exceeded, an oracle is requested to label the chosen instance. The most dependent baseline is *100% Oracle*, a traditionally-employed strategy in AL where 100% of the labels are oracle-based.

We do not compare our methods to Softmax Response (Geifman and El-Yaniv, 2017) and SelectiveNet (Geifman and El-Yaniv, 2019), despite their strong performance for selective classification, as they do not trivially extend to the setting in which labels are unavailable.

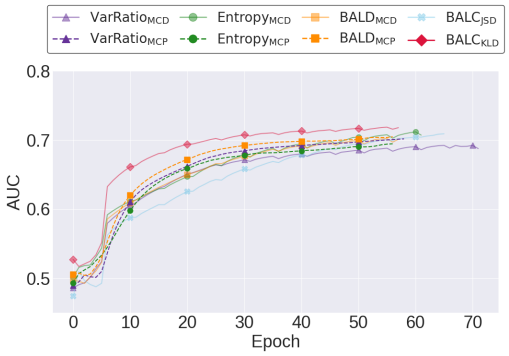
4.5 RESULTS

In this section, and with an eye on addressing the initial research question, we attempt to answer the following questions: (i) How does our consistency-based active learning framework perform in the absence of an oracle? (ii) How does our dynamic oracle

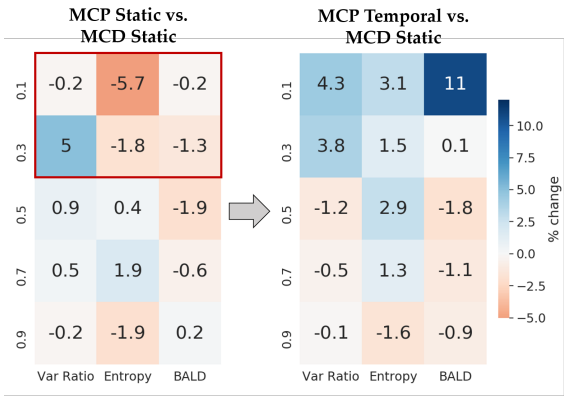
questioning framework perform in the presence of a noise-free oracle? (iii) How does our dynamic oracle questioning framework perform in the presence of a noisy oracle?

4.5.1 *Active Learning without Oracle*

We begin by exploring the performance of various active learning frameworks without an oracle. In Fig. 4.6a, we illustrate the area under the receiver operating characteristic curve (AUC) on the validation set of $\mathcal{D}_2$ when a network is initially exposed to $F = 0.3$ of the labelled training data.



(a) $\mathcal{D}_2$ at $F = 0.3$



(b) Utility of tracked acquisition functions on $\mathcal{D}_1$

Figure 4.6: **Performance of various active learning frameworks without an oracle.** Validation AUC on **(a)** $\mathcal{D}_2$ at $F = 0.3$. **(b)** Mean percent change in test AUC when comparing MCP with static and tracked acquisition functions to MCD with their static counterparts on $\mathcal{D}_1$. We show results for Var Ratio, Entropy, and BALD, at all fractions, $F \in [0.1, 0.3, 0.5, 0.7, 0.9]$ and across five seeds. In (a), we show that BALC_{KLD} outperforms state-of-the-art acquisition functions. In (b), we show that temporal acquisition functions add value relative to static ones.

In Fig. 4.6a, we find that a network exploiting BALC_{KLD} learns faster than one exploiting the remaining acquisition functions. For example, BALC_{KLD} and BALD_{MCD} achieve an AUC ≈ 0.69 after 20 and 40 epochs of training, respectively. This reflects a two-fold increase in the learning efficiency of BALC_{KLD} relative to BALD_{MCD}. We also find that a network exploiting BALC_{KLD} outperforms networks exploiting the remaining acquisition functions. For example, BALC_{KLD} achieves a final AUC ≈ 0.72 whereas the next best method, $\text{Entropy}_{\text{MCD}}$ achieves AUC ≈ 0.70. We put forth the following hypothesis for this observed improved performance. BALC_{KLD} is a

consistency-based active learning framework which implies that acquired, unlabelled instances may still be pseudo-labelled correctly by the network, which facilitates learning. In contrast, uncertainty-based acquisition functions, such as BALD_{MCD} acquire unlabelled instances to which the network is most uncertain. Therefore, pseudo-labels generated by the same network for these instances are likely to be incorrect. This, in turn, would hinder the learning process. We obtain a similar outcome for the remaining experiments (see Appendix B.1.3).

So far, we have discussed *static* acquisition functions, those which are not tracked and thus do not explicitly incorporate temporal information (over epochs). We now attempt to quantify the marginal benefit of incorporating such temporal information. In Fig. 4.6b, we present two panels. The panel on the left illustrates the percent change in the AUC when comparing MCP to MCD while using static acquisition functions. The panel on the right illustrates the same results, however, after having incorporated temporal information into the acquisition functions used in conjunction with MCP. Hence, the naming MCP Temporal.

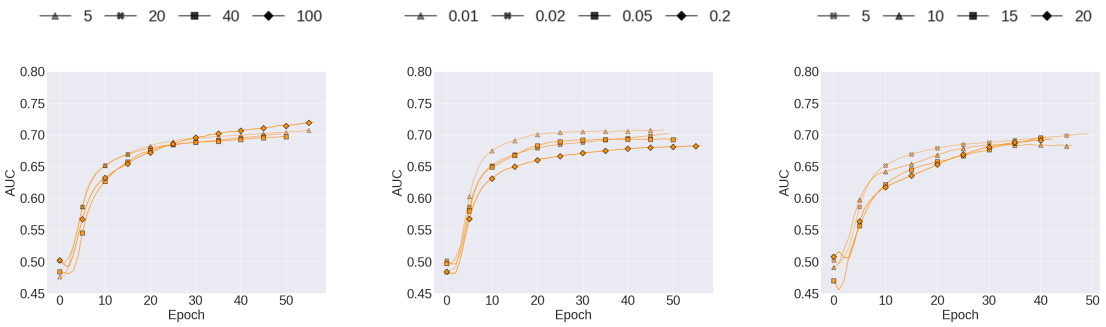
We find that tracked acquisition functions are most useful when the initial size of the labelled dataset is small ($\downarrow F$ values) (red rectangle). For example, transitioning from MCP Static to MCP Temporal when exploiting BALD at $F = 0.1$ improves the AUC by 11%. Improvements are also observed for the remaining acquisition functions. We hypothesize that this improvement is due to the increased diversity and number of hypotheses that are considered when incorporating temporal information relative to the static setting. This, in turn, might eliminate unsuitable hypotheses at a greater rate.

4.5.2 *Sensitivity Analysis of Hyperparameters*

Our active learning framework is contingent upon the choice of several hyperparameters. The first is the number of Monte Carlo samples, T, used for generating perturbations. We hypothesize that a larger number of MC samples, by increasing the

number of hypotheses considered, will lead to improved performance. The second is the acquisition fraction, b, that determines the number of unlabelled instances to acquire. Although it is convenient to hypothesize that the acquisition of more instances is advantageous, we must keep in mind that this could prove problematic without an oracle (due to incorrect pseudo-labelling). The third hyperparameter is the acquisition epoch, τ, which is the epoch interval at which acquisitions are made. We hypothesize that performing acquisitions too frequently can be detrimental to learning. This is because the network is not provided with sufficient time to learn from the most recently-acquired instances. Similarly, infrequent acquisitions can starve the network from much needed supervisory signals and also hinder performance. In Fig. 4.7, we illustrate the performance of a network trained with BALD_{MCP} as a function of these hyperparameters.

In Fig. 4.7a, we find that having more MC samples, T, can benefit the final performance of the network. For example, with $T = 5$ and 100, the AUC ≈ 0.70 and 0.72, respectively. We also note that increasing the number of samples is not guaranteed to improve performance. For example, the network performs worse with $T = 40$ than with $T = 5$. One potential explanation for this outcome is that the



(a) Effect of number of MC samples, T

(b) Effect of acquisition fraction, b

(c) Effect of acquisition epoch, τ

Figure 4.7: **Sensitivity analysis of the hyperparameters employed in the active learning framework.** We explore the effect on performance of **(a)** the number of Monte Carlo samples, T, while holding constant $b = 0.02$ and $\tau = 5$, **(b)** the acquisition fraction, b, while holding constant $T = 20$ and $\tau = 5$, and **(c)** the acquisition epoch, τ, while holding constant $T = 20$ and $b = 0.02$. The experiments are conducted with the BALD_{MCP} acquisition function. The curves reflect the mean across five random seeds.

increased *number* of hypotheses, brought about by the MC samples, may not exhibit a high level of *diversity*, which is believed to facilitate learning.

In Fig. 4.7b, we find that a network with a lower acquisition fraction, b, outperforms one with a higher fraction. For example, with $b = 0.01$ and 0.2, the final AUC ≈ 0.71 and 0.68. This is in line with our intuition. Since this active learning scenario is without an oracle, the acquisition of more instances that are pseudo-labelled implies that more label noise is injected into the learning process. Such label noise is a detriment to learning.

In Fig. 4.7c, we find that a low acquisition epoch, τ, (frequent acquisitions), allows a network to learn quickly (fewer epochs) and achieve strong generalization performance. For example, with $\tau = 5$ and 20, the network achieves AUC $= 0.65$ after 10 and 20 epochs, respectively. This reflects a two-fold increase in learning efficiency. Furthermore, with $\tau = 5$, the final performance of AUC ≈ 0.70 is higher than that achieved by networks with larger values of τ. These findings suggest that networks in this setting prefer frequent acquisitions.

4.5.3 *Active Learning with Noise-free Oracle*

In the previous section, we explored the performance of active learning frameworks without an oracle. In this section, we assume the presence of a noise-free oracle and explore the performance of various oracle questioning methods when deployed alongside a subset of acquisition functions. In Table 4.3, we present the results of these experiments on all datasets at $F = 0.1$.

In Table 4.3, we find that SoQal consistently outperforms Entropy Response and Epsilon Greedy across $\mathcal{D}_1$ - $\mathcal{D}_3$. For example, when using BALD_{MCD} on $\mathcal{D}_2$, SoQal achieves an AUC $= 0.707$ whereas Entropy Response and Epsilon Greedy achieve AUC $= 0.584$ and 0.609, respectively. Such a finding suggests that SoQal is better equipped to know *when* and for which *instance* a label should be requested from an oracle. One could argue that the improved performance of SoQal can be attributed to

Table 4.3: **Mean test AUC of oracle questioning methods with a noise-free oracle at $F = 0.1$.** We present the results for a subset of the acquisition functions when evaluated on datasets, $\mathcal{D}_1 - \mathcal{D}_4$. The mean and standard deviation are reported across five random seeds. Bold results indicate the top-performing method. We show that SoQal outperforms the remaining oracle questioning methods, Entropy Response and Epsilon Greedy across $\mathcal{D}_1 - \mathcal{D}_3$.

Dataset	Ac. Function α				Oracle Questioning Method		
		No AL	No Oracle	100% Oracle	Entropy Response	Epsilon Greedy	SoQal (ours)
$\mathcal{D}_1$	BALD_{MCD}	0.577 ± 0.014	0.465 ± 0.017	0.653 ± 0.013	0.496 ± 0.039	0.491 ± 0.028	$\mathbf{0.621 \pm 0.021}$
	BALD_{MCP}		0.464 ± 0.023	0.676 ± 0.020	0.517 ± 0.043	0.501 ± 0.043	$\mathbf{0.645 \pm 0.015}$
	BALC_{KLD}		0.500 ± 0.023	0.634 ± 0.030	0.548 ± 0.034	0.548 ± 0.042	$\mathbf{0.598 \pm 0.055}$
	Temporal BALC_{KLD}		0.496 ± 0.024	0.659 ± 0.033	0.536 ± 0.040	0.521 ± 0.059	$\mathbf{0.646 \pm 0.067}$
$\mathcal{D}_2$	BALD_{MCD}	0.679 ± 0.040	0.573 ± 0.063	0.713 ± 0.053	0.584 ± 0.041	0.609 ± 0.071	$\mathbf{0.707 \pm 0.038}$
	BALD_{MCP}		0.589 ± 0.045	0.735 ± 0.028	0.638 ± 0.043	0.637 ± 0.044	$\mathbf{0.677 \pm 0.042}$
	BALC_{KLD}		0.602 ± 0.044	0.722 ± 0.018	0.582 ± 0.017	0.643 ± 0.033	$\mathbf{0.677 \pm 0.024}$
	Temporal BALC_{KLD}		0.575 ± 0.017	0.735 ± 0.011	0.612 ± 0.050	0.605 ± 0.019	$\mathbf{0.648 \pm 0.057}$
$\mathcal{D}_3$	BALD_{MCD}	0.716 ± 0.012	0.581 ± 0.014	0.802 ± 0.008	0.588 ± 0.013	0.673 ± 0.015	$\mathbf{0.721 \pm 0.025}$
	BALD_{MCP}		0.623 ± 0.020	0.798 ± 0.007	0.676 ± 0.058	0.665 ± 0.028	$\mathbf{0.720 \pm 0.044}$
	BALC_{KLD}		0.631 ± 0.010	0.787 ± 0.008	0.629 ± 0.004	0.643 ± 0.041	$\mathbf{0.731 \pm 0.033}$
	Temporal BALC_{KLD}		0.600 ± 0.005	0.794 ± 0.002	0.630 ± 0.014	0.654 ± 0.019	$\mathbf{0.730 \pm 0.024}$
$\mathcal{D}_4$	BALD_{MCD}	0.486 ± 0.023	0.486 ± 0.011	0.585 ± 0.011	0.489 ± 0.030	0.474 ± 0.037	0.468 ± 0.021
	BALD_{MCP}		0.493 ± 0.030	0.605 ± 0.024	0.504 ± 0.026	0.492 ± 0.024	0.499 ± 0.029
	BALC_{KLD}		0.505 ± 0.032	0.588 ± 0.033	0.504 ± 0.039	0.473 ± 0.010	0.495 ± 0.012
	Temporal BALC_{KLD}		0.511 ± 0.030	0.532 ± 0.027	0.496 ± 0.023	0.496 ± 0.023	0.503 ± 0.010

its high dependence on the oracle. We quantify the dependence of SoQal on an oracle in Sec. 4.5.5.

We also find that SoQal performs on par with the remaining methods on $\mathcal{D}_4$. We hypothesize that this is due to the cold-start problem (Konyushkova et al., 2017); active learning networks are unable to start learning how to solve the task at hand due to the limited availability of initial, labelled training data. We support this claim with experiments in Sec. 4.5.5. Moreover, we remind readers that by increasing the value of S in the SoQal experiments, networks can cede more control to the oracle and thus further improve performance, an effect we also quantify in Sec. 4.5.5.

4.5.4 *Active Learning with Noisy Oracle*

So far, we have presented active learning scenarios characterized by either the absence of oracles or the presence of those capable of providing noise-free labels. In this section, we explore the performance of our oracle questioning methods with a noisy oracle. In Fig. 4.8, we illustrate the AUC on the test set of $\mathcal{D}_1$ of these methods as a function of various types and levels of noise. For comparison, the horizontal dashed lines reflect the performance of the methods with a *noise-free* oracle.

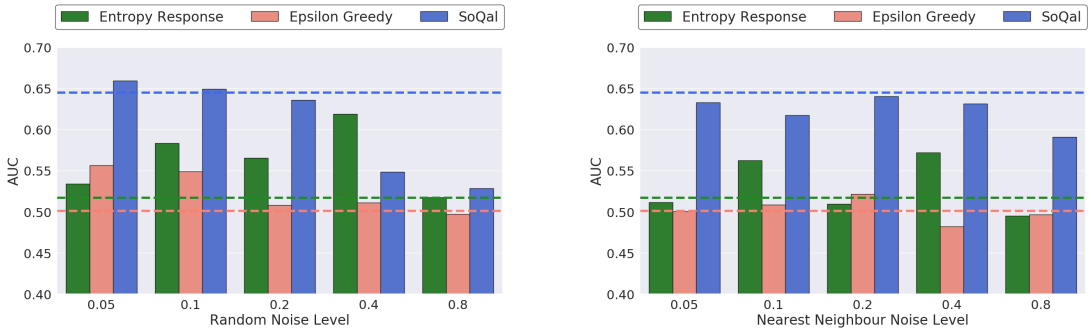


Figure 4.8: **Average test AUC of the oracle questioning methods without label noise and with various magnitudes of label noise on $\mathcal{D}_1$ while using** BALD_{MCP}. The horizontal dashed lines indicate the performance of network trained *without* label noise. We show that SoQal, with up to 80% random or nearest neighbour label noise, continues to outperform the remaining methods when trained *without* label noise. This finding suggests that SoQal is better equipped than the remaining methods to deal with label noise.

In Fig. 4.8, we find that SoQal outperforms the remaining methods regardless of noise type and magnitude (except at $\gamma = 0.4$ random noise). For example, at 5% random noise, SoQal achieves an AUC ≈ 0.66 whereas Epsilon Greedy and Entropy Response achieve an AUC ≈ 0.56 and ≈ 0.53, respectively. We also find that injecting label noise can sometimes improve performance. By comparing the performance of the network in the *absence* of label noise (blue horizontal dashed line in Fig. 4.8) to one in the *presence* of 5% label noise, we see that the latter performs better than the former. For example, SoQal achieves AUC $\approx 0.64 \rightarrow 0.66$ with no noise and 5% random noise, respectively. This suggests that the 5% label noise which is injected into the system could have contributed to that improved performance. One hypothesis for this is that the original dataset contained label noise (e.g., perhaps due to improper labelling by data curators) and by our introducing label noise, we nudged the incorrect labels (inherent label noise) to their correct value. Furthermore, we find that SoQal is better able to deal with label noise than the remaining methods. Specifically, SoQal at 80% random noise achieves AUC ≈ 0.53 whereas networks exploiting Epsilon Greedy and Entropy Response *without* label noise achieve AUC ≈ 0.50 and ≈ 0.52, respectively. This observation, which is more pronounced when dealing with

nearest neighbour noise, points to the utility of SoQal in the presence of a noisy oracle. We arrive at similar conclusions when experimenting with other datasets and acquisition functions.

4.5.5 *Dependence of SoQal on Oracle*

Recall that one of our original goals is to reduce the labelling burden that is placed on an oracle (e.g., physician). In this section, we look to quantify the degree to which our active learning framework is dependent on an oracle. To do so, we define a metric, which we refer to as the oracle ask rate (OAR), that quantifies the proportion of all acquired instances whose labels are requested from an oracle. For example, an OAR $= 50\%$ implies that 50% of the acquired instances had labels provided by an oracle, with the remaining 50% pseudo-labelled by the network. In Fig. 4.9a (left), we present the oracle ask rate of SoQal as a function of the type and magnitude of label noise.

Label noise and dependence on oracle. In Fig. 4.9a (left), we find that SoQal alleviates the labelling burden placed on an oracle. For example, SoQal when used in conjunction with BALD_{MCP} and without label noise, achieves OAR $\approx 0.68 < 1.0$. This suggests that 32% fewer label requests were sent to an oracle compared to an active learning framework that requests 100% of its labels from an oracle. However, these fewer label requests do result in slightly lower performance compared to the 100% method (see earlier Table 4.3). This is expected since noise-free oracle-provided labels are likely to be of higher quality than those generated by a network (via pseudo-labels).

We also find that SoQal can become more dependent on an oracle with increased levels of label noise (either random or nearest neighbour). For example, with random noise $\gamma = 0.05 \rightarrow 0.8$, the average OAR $\approx 0.72 \rightarrow 0.80$. This increased dependence can be problematic, particularly when the oracle is highly likely to annotate unlabelled instances incorrectly. Nonetheless, SoQal, even with extreme label noise,

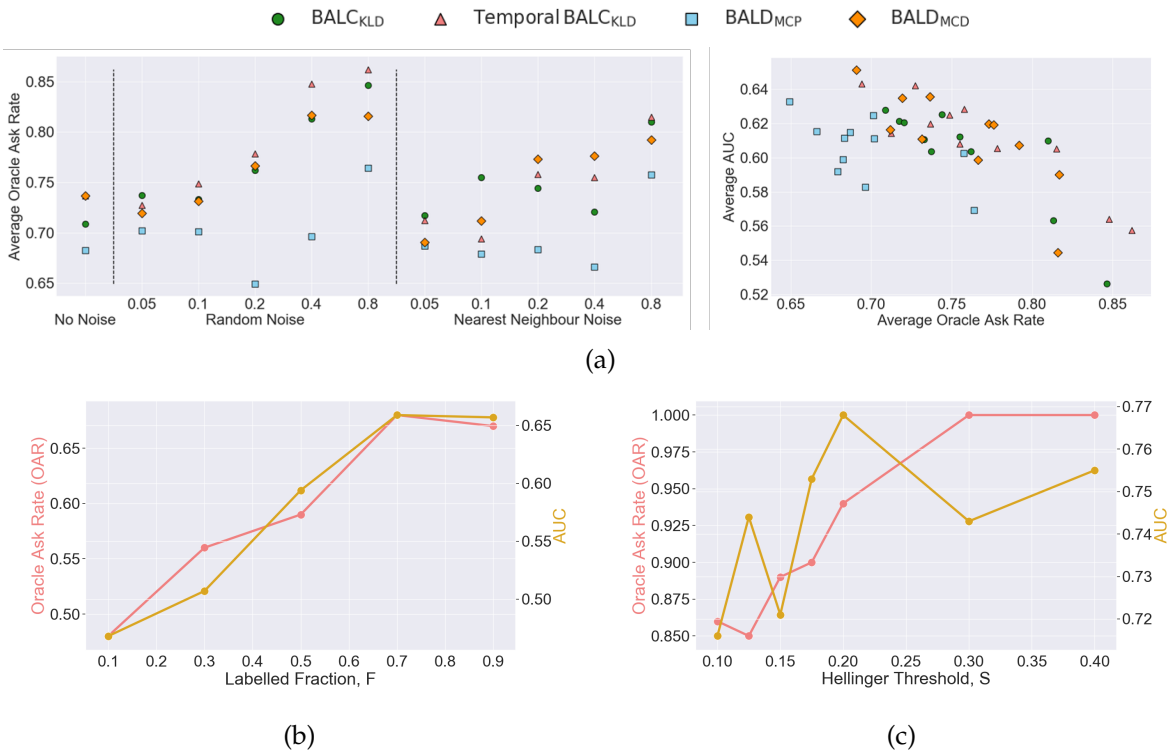


Figure 4.9: **Dependence of SoQal on oracle as a function label noise, data availability, and the Hellinger threshold. (a)** (left) Average oracle ask rate (OAR) as a function of various types and levels of label noise, shown for a subset of acquisition functions. (right) Correlation between oracle ask rate and generalization performance. We also present the OAR and corresponding AUC while using BALD$_{MCD}$ as a function of **(b)** the labelled fraction, F, of labelled data and **(c)** the Hellinger threshold, S. All results are averaged across five random seeds. In (a) (right), we show that the reduced dependence of a network on an oracle can be advantageous. This is particularly true if the oracle exhibits label noise. In (b), we show that increasing the amount of labelled data increases the network's dependence on an oracle. In (c), we show that tuning S provides researchers with flexibility over how often to request a label from an oracle.

achieves OAR < 1.0 (fewer requests) while continuing to outperform the remaining questioning methods (see earlier Fig. 4.8).

In Fig. 4.9a (right), we further explore the relationship between the oracle ask rate and generalization performance. We find that in the scenarios *with* label noise (both random and nearest neighbour), the decreased dependence of SoQal on an oracle is associated with improved generalization performance. This is shown by the negative correlation between the average OAR and the AUC. For example, with OAR $\approx 0.85 \rightarrow 0.65$, the AUC $\approx 0.53 \rightarrow 0.63$. This suggests that SoQal performs better while requesting *fewer* labels. Such a finding reaffirms the observation that

SoQal appropriately learns *when* to request a label from an oracle or to pseudo-label instead.

Data availability and dependence on oracle. The previous oracle questioning results we presented were based on experiments conducted on limited, labelled datasets (i.e., $F = 0.1$). In this section, we explore the dependence of SoQal on an oracle when provided with more labelled data (i.e., $F > 0.1$). In Fig. 4.9b, we present the OAR and corresponding AUC as a function of the amount of available labelled data, F. As expected, we find that performance improves with more data. For example, as $F = 0.1 \rightarrow 0.9$, AUC $\approx 0.50 \rightarrow 0.65$. We also find that SoQal continues to alleviate the labelling burden placed on an oracle even when more labelled data are available. For example, at $F = 0.9$, SoQal exhibits OAR $\approx 0.67 < 1.0$, reflecting a 33% reduction in labelling burden.

Intuitively, the availability of more labelled data (in the absence of label noise) should allow a network to learn the task quite well and quickly (within a few epochs). As such, we would expect it to pseudo-label more instances correctly and thus have less of a need for an oracle. Our oracle selection framework, however, does not explicitly account for the amount of labelled data available for training. In our existing setup, we can account for this by apriori lowering the Hellinger threshold, S, to reflect our increased confidence in the ability of the network to perform pseudo-labelling.

Hellinger threshold and dependence on oracle. We claimed that the Hellinger threshold, S, can be tuned according to the relative trust one has in the network and the oracle. A higher threshold ($\uparrow S$) implies that less trust is placed in the network than in the oracle. In Fig. 4.9c, we present the OAR and corresponding AUC as a function of the Hellinger threshold, S.

Consistent with expectations, we find that the dependence of SoQal on the oracle increases with the threshold. For example, as $S = 0.10 \rightarrow 0.40$, OAR $\approx 0.86 \rightarrow 1.0$. Such a finding suggests that researchers can set the threshold, S, a priori based on the extent to which they would like to request labels from an oracle. We also

find that this increased dependence sometimes results in worse performance. For example, although transitioning from $S = 0.20 \rightarrow 0.30$ leads to OAR $\approx 0.95 \rightarrow 1.0$, the AUC $\approx 0.77 \rightarrow 0.74$. We hypothesize that this worse performance is due to inherent label noise in the datasets. Therefore, a strategy which depends on an oracle 100% of the time is worse than one which delegates some of the labelling to the network instead. Such a finding provides further evidence in support of a dynamic oracle selection strategy.

5

CONTRASTIVE LEARNING FOR CARDIAC ARRHYTHMIA DIAGNOSIS

> *The holy grail of machine learning is to find an in-sample estimate of the out-of-sample error. If you get that, you are done, minimize it and go home*
>
> — Yaser Abu Mustafa, Caltech

HEALTHCARE is an industry that generates troves of unlabelled, clinical data. Designing algorithms that exploit such data, at scale, can help uncover meaningful insight which guides the diagnosis, prognosis, and treatment of medical conditions. In Chapter 4, we proposed one such algorithm which exploited the active learning framework and assumed that physicians can be available during the learning process. To further mitigate the burden placed on physicians, we make the assumption that physicians are *unavailable* during the algorithmic learning process. Consequently, we are now interested in tackling the following question.

Research Question

How can we design clinical algorithms that exploit abundant, unlabelled data in the absence of physicians to extract insight from limited, labelled data?

In this chapter, we address the outlined research question in the context of diagnosing cardiac arrhythmias based on the electrocardiogram. To achieve this, we exploit the 'pre-train then transfer' paradigm in which neural network parameters are pre-trained on an upstream, and potentially arbitrary, task before being transferred to solve a downstream task of clinical utility. Specifically, we build upon a self-supervised approach, known as contrastive learning (Chen et al., 2020a), in which representations of instances with a shared context (e.g., similar medical diagnosis) are

encouraged to be similar to one another and dissimilar from representations with a different context. Our contributions are twofold. First, we propose a family of patient-specific contrastive learning methods, entitled CLOCS, that exploit both temporal and spatial invariances present within ECG signals. Second, we show that CLOCS outperforms both domain-specific and generic state-of-the-art methods, BYOL (Grill et al., 2020) and SimCLR (Chen et al., 2020a), when performing a linear evaluation of, and fine-tuning on, downstream tasks involving cardiac arrhythmia classification.

5.1 RELATED WORK

Self-supervision for medical time-series. In the context of deep learning, self-supervision focuses on learning representations of datapoints without ground-truth annotations. Miotto et al. (2016) propose DeepPatient, a 3-layer stacked denoising autoencoder that attempts to learn a patient representation using electronic health record (EHR) data. Although performed on a large proprietary dataset, their approach is focused on EHRs and does not explore contrastive learning for physiological signals. Sarkar and Etemad (2020) apply existing self-supervised methods on ECG recordings in the context of affective computing. The methods implemented include defining pretext classification tasks such as temporal inversion, negation, time-warping, etc. Their work is limited to affective computing, does not explore contrastive learning, and does not exploit multi-lead data as we do. Lyu et al. (2018) explore a sequence to sequence model to learn representations from EHR data in the eICU dataset. In the process, they minimize the reconstruction error of the input time-series. Li et al. (2020d) leverage the aforementioned unsupervised learning technique on a large clinical dataset, CPRD, to obtain uncertainty estimates for predictions.

Contrastive learning. In contrastive predictive coding, Oord et al. (2018) use representations of current segments to predict those of future segments. More recently, Tian et al. (2019) propose contrastive multi-view coding where multiple views of the same image are treated as 'shared context'. He et al. (2019); Chen et al. (2020a); Grill

et al. (2020) exploit the idea of instance discrimination (Wu et al., 2018) and interpret multiple views as stochastically augmented forms of the same instance. They explore the benefit of sequential data augmentations and show that cropping and colour distortions are the most important. These augmentations, however, do not trivially extend to the time-series domain. Shen et al. (2020) propose to create mixtures of images to smoothen the output distribution and thus prevent the model from being overly confident. Time Contrastive Learning (Hyvarinen and Morioka, 2016) performs contrastive learning over temporal segments in a signal and illustrate the relationship between their approach and ICA. In contrast to our work, they formulate their task as prediction of the segment index within a signal and perform limited experiments that do not exploit the noise contrastive estimation (NCE) loss (Bachman et al., 2019). Time Contrastive Networks (Sermanet et al., 2017) attempt to learn commonalities across views and differences across time. In contrast, our work focuses on identifying commonalities across *both* spatial and temporal components of data.

5.2 BACKGROUND

5.2.1 *Contrastive Learning*

Let us assume the presence of a learner $f_\theta : \boldsymbol{x} \in \mathbb{R}^D \rightarrow \boldsymbol{h} \in \mathbb{R}^E$, parameterized by $\boldsymbol{\theta}$, which maps a D-dimensional instance, $\boldsymbol{x}$, to an E-dimensional representation, $\boldsymbol{h}$. We also have an unlabelled dataset, $\boldsymbol{X_U} \in \mathbb{R}^{N \times D}$, where N is the total number of instances.

Each unlabelled instance, $\boldsymbol{x}_i \in \boldsymbol{X_U}$, is exposed to a set of transformations, T^A and T^B, such that $\boldsymbol{x}_i^A = T^A(\boldsymbol{x}_i)$ and $\boldsymbol{x}_i^B = T^B(\boldsymbol{x}_i)$. Such transformations can consist of two different data augmentation procedures such as random cropping and flipping. These transformed instances now belong to an augmented dataset, $\boldsymbol{X}' \in \mathbb{R}^{N \times D \times V}$, where V is equal to the number of applied transformations. In contrastive learning, representations, $\boldsymbol{h}_i^A = f_\theta(\boldsymbol{x}_i^A)$ and $\boldsymbol{h}_i^B = f_\theta(\boldsymbol{x}_i^B)$, are said to share context and are referred to as a positive pair because (a) they are derived from the same original

instance, $\boldsymbol{x}_i$, and (b) the transformations applied to the original instance were class-preserving. Representations within a positive pair are encouraged to be similar to one another and dissimilar to representations of all other instances, $\boldsymbol{h}_j^A, \boldsymbol{h}_j^B$, $\forall j$, $j \neq i$, where the similarity of a pair of representations, $s(\boldsymbol{h}_i^A, \boldsymbol{h}_i^B)$, is quantified via a metric, s, such as cosine similarity. In the process, the goal is to learn representations that are invariant to different transformations of the same instance.

5.3 METHODS

5.3.1 *Positive and Negative Pairs of Representations*

Representations that are derived from the same *instance* are typically assumed to share context. This approach, however, fails to capture commonalities present *across* instances. In the medical domain, for example, multiple physiological recordings from the same patient may share context. It is important to note that if the multitude of physiological recordings associated with a patient were collected over large time-scales (e.g., on the order of years) and in drastically different scenarios (e.g., at rest vs. during a stress test), then the shared context across these recordings is likely to diminish. This could be due to changing patient demographics and disease profiles. With the previous caveat in mind, we propose to leverage commonalities present in multiple physiological recordings by redefining a positive pair to refer to representations of transformed instances that belong to the same *patient*. We outline how to arrive at these transformed instances next.

5.3.2 *Transformation Operators*

When choosing the transformation operators, T, that are applied to each instance, the principal desideratum is that they capture invariances in the ECG recording. Motivated by the observation that ECG recordings reflect both temporal and spatial information, we propose to exploit both temporal and spatial invariance. We provide an intuition for such invariances in Fig. 5.1.

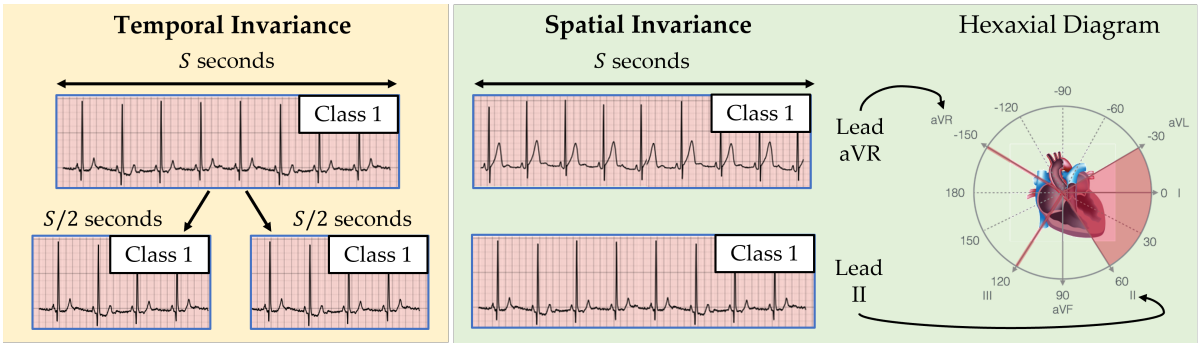


Figure 5.1: **ECG recordings reflect both temporal and spatial information.** This is because they measure the electrical activity of the heart using different leads (views) over time. **(Temporal Invariance)** Abrupt changes to the ECG recording are unlikely to occur on the order of seconds, and therefore adjacent segments of shorter duration will continue to share context. **(Spatial Invariance)** Recordings from different leads (at the same time) will reflect the same cardiac function, and thus share context.

As is pertains to temporal invariance (Fig. 5.1 left), we assume that upon splitting an ECG recording, associated with Class 1, into several segments, each of them remain associated with Class 1. We justify this assumption based on human physiology where abrupt changes in cardiac function (on the order of seconds) are unlikely to occur. If these segments were collected years apart, for example, our assumption may no longer hold. As for spatial invariance (Fig. 5.1 right), we leverage the hexaxial diagram which illustrates the location of the leads relative to the heart. We assume that temporally-aligned ECG recordings from different leads (views) are associated with the same class. This is based on the idea that multiple leads (collected at the same time) will reflect the same underlying cardiac function. Occasionally, this assumption may not hold, if, for example, a cardiac condition afflicts a specific part of the heart, making it detectable by only a few leads. We now describe how to exploit these invariances for contrastive learning.

Contrastive Multi-segment Coding (CMSC). Given an ECG recording, $\boldsymbol{x}_i$, with duration S seconds, we can extract V non-overlapping temporal segments, each with duration S/V seconds. If $V = 2$, for example, $\boldsymbol{x}_i^{t1} = T^{t1}(\boldsymbol{x}_i)$ and $\boldsymbol{x}_i^{t2} = T^{t2}(\boldsymbol{x}_i)$ where t indicates the timestamp of the temporal segment (see Fig. 5.1 left). We exploit

temporal invariances in the ECG by defining representations of these adjacent and non-overlapping temporal segments as positive pairs.

Contrastive Multi-lead Coding (CMLC). Different projections of the same electrical signal emanating from the heart are characterized by different leads, L. For example, with two leads, $L1$ and $L2$, then $\boldsymbol{x}_i^{L1} = T^{L1}(\boldsymbol{x}_i)$ and $\boldsymbol{x}_i^{L2} = T^{L2}(\boldsymbol{x}_i)$ (see Fig. 5.1 right). We exploit spatial invariances in the ECG by defining temporally-aligned representations of these different projections as positive pairs.

Contrastive Multi-segment Multi-lead Coding (CMSMLC). We simultaneously exploit both temporal and spatial invariances in the ECG by defining representations of non-overlapping temporal segments and different projections as positive pairs. For example, in the presence of two temporal segments with timestamps, $t1$ and $t2$, that belong to two leads, $L1$ and $L2$, then $\boldsymbol{x}_i^{t1,L1} = T^{t1,L1}(\boldsymbol{x}_i)$ and $\boldsymbol{x}_i^{t2,L2} = T^{t2,L2}(\boldsymbol{x}_i)$.

5.3.3 *Patient-Specific Noise Contrastive Estimation Loss*

Given our patient-centric definition of positive pairs, we propose to optimize a patient-specific noise contrastive estimation loss. More formally, Given a mini-batch of K instances, we apply a pair of transformation operators and generate $2K$ transformed instances (a subset of which is shown in Fig. 5.2). We encourage a pair of representations, $\boldsymbol{h}_i^A$ and $\boldsymbol{h}_k^B$, $i, k \in P$, from the same patient, P, to be similar to one another and dissimilar to representations from other patients. We quantify this similarity using the cosine similarity, s, with a temperature scaling parameter, τ, (5.3) as is performed in Tian et al. (2019); Chen et al. (2020a). We extend this to all representations in the mini-batch to form a similarity matrix of dimension $K \times K$. In this matrix, we identify positive pairs by associating each instance with its patient ID. By design, this includes the diagonal elements and the network is penalized if a high degree of similarity is not exhibited by these positive pairs (5.2) (left). If the same patient reappears within the mini-batch, then we also consider off-diagonal elements, resulting in the loss shown in (5.2) (right). The frequency of these off-diagonals is inconsistent due to the

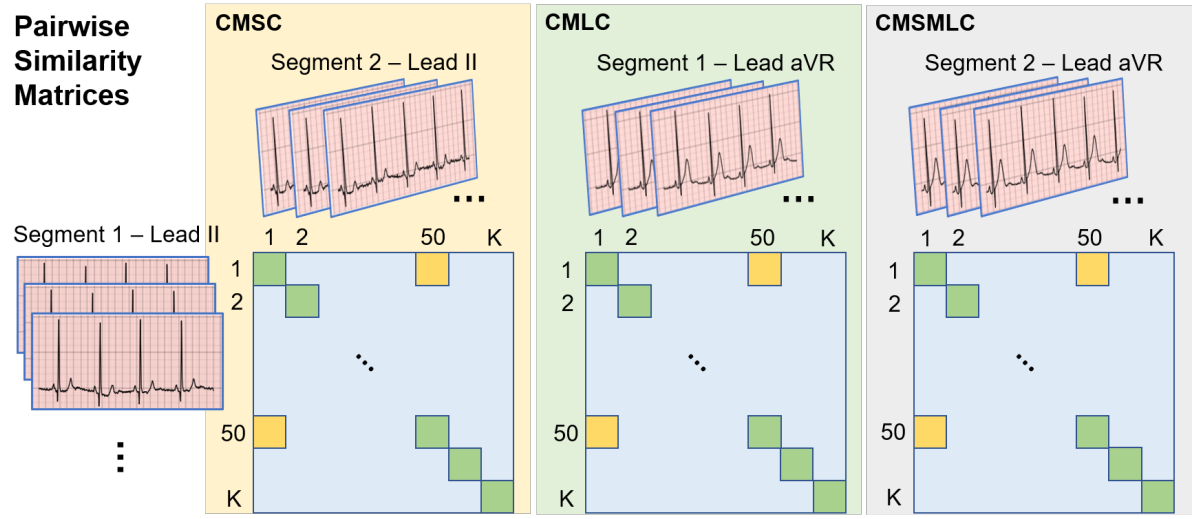


Figure 5.2: **Three proposed patient-specific contrastive learning frameworks.** Similarity matrix for a mini-batch of K instances in **(Left)** Contrastive Multi-segment Coding (CMSC), **(Centre)** Contrastive Multi-lead Coding (CMLC), and **(Right)** Contrastive Multi-segment Multi-lead Coding (CMSMLC). Additional matrices would be generated based on all pairs of applied transformation operators, T^A and T^B. Exemplar transformed ECG instances are illustrated along the edges. To identify positive pairs, we associate each instance with its patient ID. By design, diagonal elements (green) correspond to the same patient, contributing to (5.2) (left). Similarly, instances 1 and 50 (yellow) belong to the same patient, contributing to (5.2) (right). The blue area corresponds to negative examples as they pertain to instances from different patients. Here, we show similarity matrices that are the same across the frameworks. This may not, and is not required to, hold in practical settings due to the random shuffling of instances during training. The distinction across the frameworks is in the "views" of the ECG signals (see phrases above ECG signals). Such a distinction reflects whether we are considering temporal invariances, spatial invariances, or both.

random shuffling of data. In the final objective function (5.1), we include the diagonal (5.2) (left) and off-diagonal (5.2) (right) loss terms twice, to account for negative pairs in both views, and incorporate all pairwise combinations of transformation operators, T^A and T^B.

$$\mathcal{L} = \mathbb{E}_{T^A,T^B}\left[\mathcal{L}^{h^A,h^B}_{diag} + \mathcal{L}^{h^B,h^A}_{diag} + \mathcal{L}^{h^A,h^B}_{off-diag} + \mathcal{L}^{h^B,h^A}_{off-diag}\right] \tag{5.1}$$

$$\mathcal{L}^{h^A,h^B}_{diag} = -\mathbb{E}_{i\in P}\left[\log \frac{e^{s(\boldsymbol{h}^A_i,\boldsymbol{h}^B_i)}}{\sum_j e^{s(\boldsymbol{h}^A_i,\boldsymbol{h}^B_j)}}\right] \quad \mathcal{L}^{h^A,h^B}_{off-diag} = -\mathbb{E}_{i,k\in P}\left[\log \frac{e^{s(\boldsymbol{h}^A_i,\boldsymbol{h}^B_k)}}{\sum_j e^{s(\boldsymbol{h}^A_i,\boldsymbol{h}^B_j)}}\right] \tag{5.2}$$

$$s(\boldsymbol{h}_i^A, \boldsymbol{h}_i^B) = \frac{\boldsymbol{h}_i^A \cdot \boldsymbol{h}_i^B}{\|\boldsymbol{h}_i^A\| \|\boldsymbol{h}_i^B\|} \cdot \frac{1}{\tau} \tag{5.3}$$

5.4 EXPERIMENTAL DESIGN

5.4.1 *Data and Pre-processing*

To evaluate our method, we leverage four different datasets, each of which consists of cardiac time-series waveforms alongside cardiac arrhythmia labels. We split each of the aforementioned waveforms into non-overlapping segments comprising 2500 samples. In Table 5.1, we present a summary of these datasets.

Table 5.1: **Summary of the datasets used for evaluation.** We also show additional pre-processing information. Please click on the dataset's name for more information.

Dataset	Abbreviation	Modality	Normalization	Exclusion Criteria
PhysioNet 2017	$\mathcal{D}_1$	ECG	✗	-
Cardiology	$\mathcal{D}_2$	ECG	✗	-
PhysioNet 2020	$\mathcal{D}_3$	ECG	✓	-
Chapman	$\mathcal{D}_4$	ECG	✓	-

5.4.2 *Pre-training Implementation*

We conduct our pre-training experiments on the training set of two of the four datasets: PhysioNet 2020 and Chapman. We chose these datasets as they contain multi-lead data.

In **CMSC**, we extract a pair of non-overlapping temporal segments of $S = 2500$ samples. This is equivalent to either 10 or 5 seconds worth of ECG data from the Chapman and PhysioNet 2020 datasets, respectively. Therefore, our model is presented with a mini-batch of dimension $K \times S \times 2$ where K is the batchsize, and S is the number of samples.

In **CMLC**, we explore two scenarios with a different number of leads corresponding to the same instance. Our mini-batch dimension is $K \times S \times L$, where L is the number of leads.

Lastly, in **CMSMLC**, we incorporate an additional temporal segment in each mini-batch. Therefore, our mini-batch dimension is $K \times 2S \times L$. To ensure a fair comparison between all methods, we expose them to an equal number of patients and instances during training. In CMLC or CMSMLC, we either pre-train using 4 leads (II, V2, aVL, aVR) or all 12 leads. We chose these 4 leads as they cover a large range of axes (see Fig. 5.1 right). We refer to this family of contrastive learning frameworks as CLOCS.

To complete our understanding of the pre-training methods, we outline, in Table 5.2, the dimensions of the inputs fed to the network. These dimensions are expressed in the form of $N \times S \times L$ where N is the total number of instances, S is the frame length (samples) of each instance, and L (if applicable) is the number of leads used. Where L is not explicitly mentioned, we report values with four leads as this was primarily used for all experiments conducted. Further implementation details can be found in Appendix B.2.1.

We chose to pre-train the networks on the Chapman and PhysioNet 2020 datasets due to their size (absolute number of instances) and their inclusion of multi-lead data. This choice is motivated by recent literature which has demonstrated the importance of the size of pre-training datasets on downstream performance (Chen et al., 2020b). Moreover, since we look to exploit multi-lead data in our CMLC and CMSMLC variants of the CLOCS framework, we chose datasets that had such data available.

Furthermore, the pre-training datasets we have chosen do indeed share some similarities with those used for downstream fine-tuning and evaluation. They both reflect cardiac time-series signals. It has been hypothesized in the literature that the similarity of datasets used in upstream and downstream tasks plays a role in the performance gain observed in the latter (Raghu et al., 2019). We leave it to future work to explore the utility of CLOCS when, for example, transferring across modalities.

Table 5.2: **Dimension of the input data, $N \times S \times L$, used during the training and validation phases of the various self-supervised pre-training methods.** $S = 2500$ is the number of samples in each instance fed to the network. L is the number of leads (projections) used during pre-training.

Dataset	Method	Training	Validation
PhysioNet 2020	BYOL	$51,880 \times S$	$12,948 \times S$
	SimCLR	$51,880 \times S$	$12,948 \times S$
	CMSC	$24,080 \times 2S$	$6,076 \times 2S$
	CMLC	$24,080 \times S \times L$	$6,076 \times S \times L$
	CMSMLC	$6,020 \times 2S \times L$	$1,519 \times 2S \times L$
Chapman	BYOL	$25,543 \times S$	$8,512 \times S$
	SimCLR	$25,543 \times S$	$8,512 \times S$
	CMSC	$25,543 \times 2S$	$8,512 \times 2S$
	CMLC	$25,543 \times S \times L$	$8,512 \times S \times L$
	CMSMLC	$6,382 \times 2S \times L$	$2125 \times 2S \times L$

5.4.3 *Evaluation on Downstream Task*

We evaluate our pre-trained methods in two scenarios. In **Linear Evaluation of Representations**, we are interested in evaluating the utility of the fixed feature extractor in learning representations. Therefore, the pre-trained parameters are frozen and multinomial logistic regression is performed on the downstream supervised task. In **Transfer Capabilities of Representations**, we are interested in evaluating the inductive bias introduced by pre-training. Therefore, the pre-trained parameters are used as an initialization for training on the downstream supervised task. In Table 5.3, we present the number of instances in the training, validation, and test sets of the downstream tasks.

Table 5.3: **Number of instances used during the supervised training of the downstream tasks.** For multi-lead datasets*, these represent sample sizes for the four leads (II, V2, aVL, aVR). Values in brackets indicate the number of patients in each of the sets.

Dataset	Training	Validation	Test
PhysioNet 2017	18,256 (5,459)	4,581 (1,364)	5,824 (1,705)
Cardiology	4,584 (201)	1,109 (50)	1,386 (62)
PhysioNet 2020*	51,880 (4,402 patients)	12,948 (1,100)	15,820 (1,375)
Chapman*	25,543 (6,387)	8,512 (2,129)	8,520 (2,130)

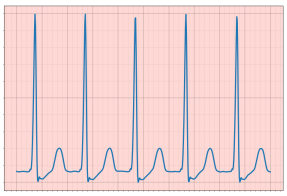
5.4.4 *Baseline Methods*

We compare our pre-training methods to networks that are initialized randomly (**Random Init.**), via supervised pre-training (**Supervised**), or via a multi-task pre-training mechanism introduced specifically for ECG signals (**MT-SSL**) (Sarkar and Etemad, 2020). We also compare to **BYOL** (Grill et al., 2020) and **SimCLR** (Chen et al., 2020a), which encourage representations of instances and their perturbed counterparts to be similar to one another, with the aim of learning transformation-invariant representations that transfer well. As SimCLR has been shown to be highly dependent on the choice of perturbations, we explore the following time-series perturbations (see Fig. 5.3 for visualizations):

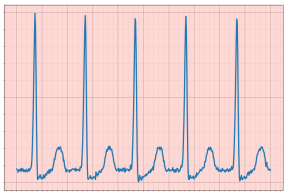
- **Gaussian** - we add $\epsilon \sim \mathcal{N}(0, \sigma)$ to the time-series signal where we chose σ based on the amplitude of the signal. This was motivated by the work of Han et al. (2020) who recently showed the effect of additive noise on ECG signals.
- **Flip** - we flip the time-series signal temporally ($\mathbf{Flip}_Y$), reversing the arrow of time, or we invert the time-series signal along the x-axis ($\mathbf{Flip}_X$).
- **SpecAugment** (Park et al., 2019) - we take the short-time Fourier transform of the time-series signal, generating a spectrogram. We then mask either temporal ($\mathbf{SA}_t$) or spectral ($\mathbf{SA}_f$) bins of varying widths before converting the spectrogram to the time domain. We also explore the application of sequential perturbations to the time-series signal.

5.5 RESULTS

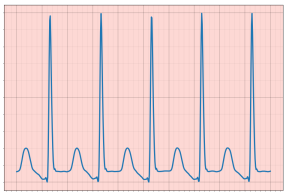
In this section, and with an eye on addressing the initial research question, we attempt to answer the following questions: (i) How does our framework perform in the linear evaluation scenario? (ii) How does our framework perform in the fine-tuning scenario?



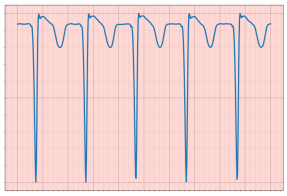
(a) Original (unperturbed)



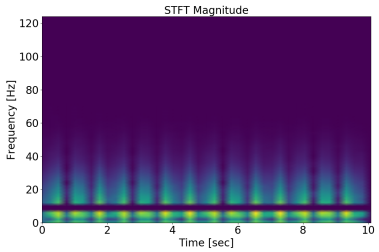
(b) Gaussian Noise



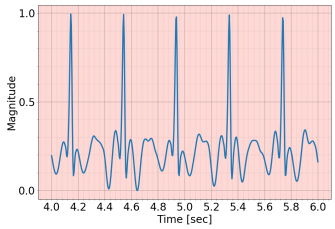
(c) Flip_Y



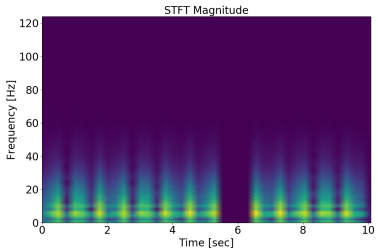
(d) Flip_X



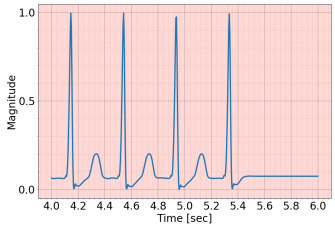
(e) Spectrogram with spectral mask, SA_f



(f) Frame with spectral mask



(g) Sepctrogram with temporal mask, SA_t



(h) Frame with temporal mask

Figure 5.3: **Illustration of perturbations applied to the raw ECG signals.** ECG segment (a) without any perturbations, (b) with additive Gaussian noise, (c) after being flipped temporally, Flip_Y, and (d) after being flipped along the x-axis, Flip_X. In (e) and (g), we show spectrograms with spectral and temporal masks, respectively. In (f) and (h), we show the time-domain equivalent of the respective masked spectrograms. Note that the time-domain segments only span seconds 4-6.

(iii) Does our framework lead to more efficient learning on downstream tasks? (iv) Does our framework lead to the learning of patient-specific representations?

5.5.1 *Linear Evaluation of Representations*

In this section, we evaluate the utility of the self-supervised representations learned using four leads on a downstream linear classification task. In Table 5.4, we show the test AUC on Chapman and PhysioNet 2020 using 50% of the labelled data ($F = 0.5$) after having learned representations, with dimension $E = 128$, using the same two datasets.

Figure 5.4: **Test AUC of the linear evaluation of the representations at** $F = 0.5$**,** after having pre-trained on Chapman or PhysioNet 2020 with $E = 128$. Pre-training and evaluating multi-lead datasets* using 4 leads (II, V2, aVL, aVR). Mean and standard deviation are shown across 5 seeds.

Dataset	Chapman*	PhysioNet 2020*
MT-SSL	0.677 ± 0.024	0.665 ± 0.015
BYOL	0.643 ± 0.043	0.595 ± 0.018
SimCLR	0.738 ± 0.034	0.615 ± 0.014
CMSC	**0.896 ± 0.005**	**0.715 ± 0.033**
CMLC	0.870 ± 0.022	0.596 ± 0.008
CMSMLC	0.847 ± 0.024	0.680 ± 0.008

We show that CMSC outperforms BYOL and SimCLR on both datasets. On the Chapman dataset, CMSC and SimCLR achieve an AUC = 0.896 and 0.738, respectively, illustrating a 15.8% improvement. Such a finding implies that the representations learned by CMSC are richer and thus allow for improved generalization. We hypothesize that this is due to the setup of CMSC whereby the shared context is across segments (temporally) and patients. Moreover, we find that CLOCS (any one of the 3 proposed methods) outperforms SimCLR in 100% of all conducted experiments, even when pre-training and evaluating with all 12 leads (see Appendix B.2.2).

5.5.2 *Effect of Perturbations on Performance*

So far, we have presented CLOCS without having incorporated, during pre-training, any of the perturbations mentioned in Sec. 5.4.4. However, contrastive learning methods, and in particular SimCLR, are notorious for their over-dependence on the choice of perturbations. To explore this dependence, we apply a diverse set of

stochastic perturbations, P, during pre-training and observe its effect on generalization performance. We follow the setup introduced by Chen et al. (2020a) and apply either a **single perturbation** to each instance, $\boldsymbol{x}_i$, whereby $\boldsymbol{x}_i^1 = P^1(\boldsymbol{x}_i)$, or **sequential perturbations** whereby $\boldsymbol{x}_i^{1,2} = P^2(P^1(\boldsymbol{x}_i))$.

We apply such perturbations while pre-training with SimCLR or CMSC on PhysioNet 2020 using 4 leads and, in Fig. 5.5, illustrate the test AUC in the linear evaluation scenario. We show that, regardless of the type and number of perturbations, CMSC continues to outperform SimCLR. For example, the *worst-performing* CMSC implementation (Flip_Y) results in an AUC $= 0.661$ which is still greater than the *best-performing* SimCLR implementation (Gaussian $\rightarrow \text{SA}_t$) with an AUC $= 0.636$. In fact, we find that pre-training with CMSC *without* applying any perturbations (see Table 5.4) still outperforms the best-performing SimCLR implementation. Such a finding suggests that CMSC's already strong performance is more likely to stem from its redefinition of the 'shared context' to include both time and patients than from the choice of perturbations.

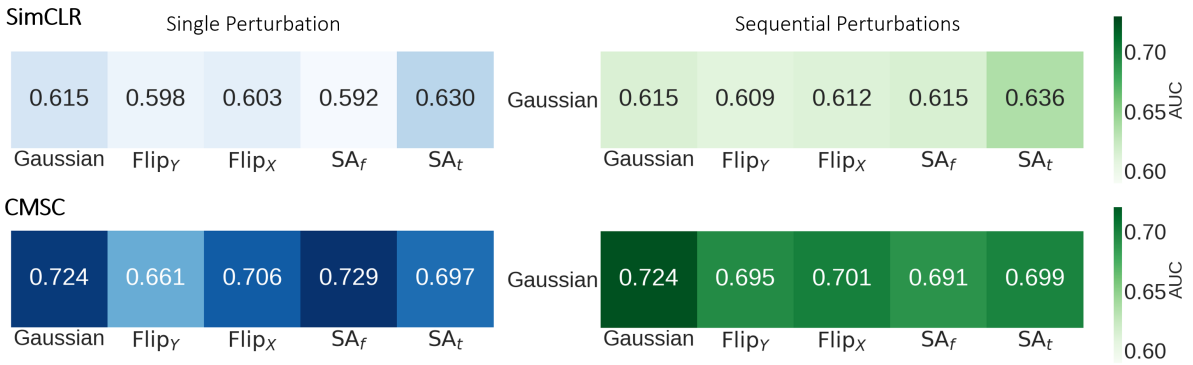


Figure 5.5: **Effect of single (blue) and sequential (green) perturbations applied to the (top) SimCLR and (bottom) CMSC implementations on linear evaluation.** Sequential perturbations involve a Gaussian perturbation followed by one of the remaining four types. Pre-training and evaluation was performed on PhysioNet 2020 using 4 leads. Evaluation was performed at $F = 0.5$ and results are averaged across 5 seeds. We show that CMSC outperforms SimCLR regardless of the applied perturbation.

5.5.3 *Transfer Capabilities of Representations*

In this section, we evaluate the utility of initializing a network for a downstream task with parameters learned via self-supervision using four leads. In Table 5.4, we show the test AUC on downstream datasets at $F = 0.5$ for the various self-supervised methods with $E = 128$.

We show that, with a few exceptions, self-supervision is advantageous relative to a Random Initialization. This can be seen by the higher AUC achieved by the former relative to the latter. We also show that, depending on the downstream dataset, either CMSC or CMSMLC outperform BYOL and SimCLR. For example, when pre-training on Chapman and fine-tuning on Cardiology, CMSMLC achieves an AUC $= 0.717$, a 4.1% improvement compared to SimCLR. This implies that by encouraging representations across space, time, and patients to be similar to one another, networks are nudged into a favourable parameter space. In Appendix B.2.2, we extend these findings and illustrate that CLOCS outperforms SimCLR in at least 75% of all experiments conducted, on average. When pre-training, fine-tuning, and evaluating using all 12 leads, we show that CMSC outperforms all other methods in at least 90% of all experiments conducted (see Appendix B.2.2). When comparing the CLOCS frameworks, we find that, on average, CMSC outperforms the remaining two methods (CMLC and CMSMLC) in most experimental scenarios (please see here for evidence that corroborates this claim). We hypothesize that this is due it being more likely that the assumption of temporal invariance holds than that of spatial invariance.

5.5.4 *Doing More With Less Labelled Data*

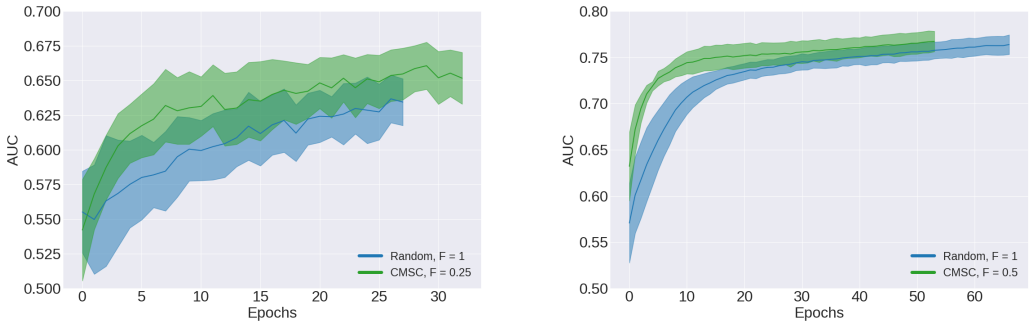
Having established that self-supervision can nudge networks to a favourable parameter space, we set out to investigate whether such a space can lead to strong generalization with less labelled data in the downstream task. In Fig. 5.6, we illustrate

Table 5.4: **Test AUC in the fine-tuning scenario at** $F = 0.5$, after having pre-trained on Chapman or PhysioNet 2020 with $E = 128$. Pre-training, fine-tuning, and evaluating multi-lead datasets* using 4 leads. Mean and standard deviation are shown across 5 seeds.

Pretraining Dataset	Chapman*			PhysioNet 2020*		
Downstream Dataset	Cardiology	PhysioNet 2017	PhysioNet 2020*	Cardiology	PhysioNet 2017	Chapman*
Random Init.	0.678 ± 0.011	0.763 ± 0.005	0.803 ± 0.008	0.678 ± 0.011	0.763 ± 0.005	0.907 ± 0.006
Supervised	0.684 ± 0.015	0.799 ± 0.008	0.827 ± 0.001	0.730 ± 0.002	0.810 ± 0.009	0.954 ± 0.003
Self-supervised Pre-training						
MT-SSL	0.650 ± 0.009	0.741 ± 0.012	0.774 ± 0.010	0.661 ± 0.011	0.746 ± 0.016	0.923 ± 0.007
BYOL	0.678 ± 0.021	0.748 ± 0.014	0.802 ± 0.013	0.674 ± 0.022	0.757 ± 0.010	0.916 ± 0.009
SimCLR	0.676 ± 0.011	0.772 ± 0.010	0.823 ± 0.011	0.658 ± 0.027	0.762 ± 0.009	0.923 ± 0.010
CMSC	0.695 ± 0.024	0.773 ± 0.013	**0.830 ± 0.002**	**0.714 ± 0.014**	0.760 ± 0.013	**0.932 ± 0.008**
CMLC	0.665 ± 0.016	0.767 ± 0.013	0.810 ± 0.011	0.675 ± 0.013	0.762 ± 0.007	0.910 ± 0.012
CMSMLC	**0.717 ± 0.006**	**0.774 ± 0.004**	0.814 ± 0.009	0.698 ± 0.011	**0.774 ± 0.012**	0.930 ± 0.012

the validation AUC of networks initialized randomly, with access to 100% of the data, or via CMSC, with access to less data, and fine-tuned on two different datasets.

We find that fine-tuning a network based on a CMSC initialization drastically improves data-efficiency. In Fig. 5.6a, we show that a network initialized with CMSC and exposed to only 25% of the labelled data outperforms one that is initialized randomly and exposed to 100% of the labelled data. This can be seen by the consistently higher AUC during, and at the end of, training. A similar outcome can be seen in Fig. 5.6b. This suggests that self-supervised pre-training exploits data efficiently such that it can do more (learn more quickly and achieve stronger generalization) with less (fewer labelled instances) on downstream classification tasks.

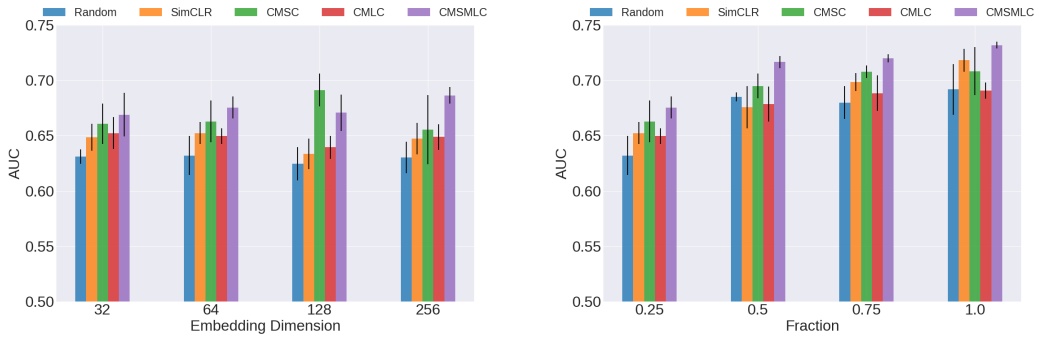


(a) PhysioNet 2020 → Cardiology (b) Chapman → PhysioNet 2017

Figure 5.6: **Validation AUC of a network initialized randomly or via CMSC and which is exposed to different amounts of labelled training data,** F**.** Results are averaged across 5 seeds. Shaded area represents one standard deviation.

5.5.5 *Effect of Embedding Dimension and Availability of Labelled Data*

The dimension of the representation learned during self-supervision and the availability of labelled training data can both have an effect on model performance. In this section, we investigate these claims. In Figs. 5.7a and 5.7b, we illustrate the test AUC for all pre-training methods as a function of $E \in [32, 64, 128, 256]$ and $F \in [0.25, 0.50, 0.75, 1]$.



(a) Chapman → Cardiology, $F = 0.25$ (b) Chapman → Cardiology, $E = 64$

Figure 5.7: **Effect of (a) embedding dimension, E, and (b) labelled fraction, F, on the test AUC** when pre-training on Chapman and fine-tuning on Cardiology. Results are averaged across 5 seeds. Error bars represent one standard deviation.

In Fig. 5.7a, we show that networks initialized randomly or via SimCLR are not significantly affected by the embedding dimension. This can be seen by the AUC ≈ 0.63 and ≈ 0.65, for these two methods across all values of E. In contrast, the embedding dimension has a greater effect on CMSC where AUC $\approx 0.66 \rightarrow 0.69$ as $E = 32 \rightarrow 128$. This implies that CMSC is still capable of achieving strong generalization performance despite the presence of few labelled data ($F = 0.25$). We hypothesize that the strong performance of CMSC, particularly at $E = 128$, is driven by its learning of patient-specific representations (see Sec. 5.5.6) that cluster tightly around one another, a positive characteristic especially when such representations map to the same downstream class.

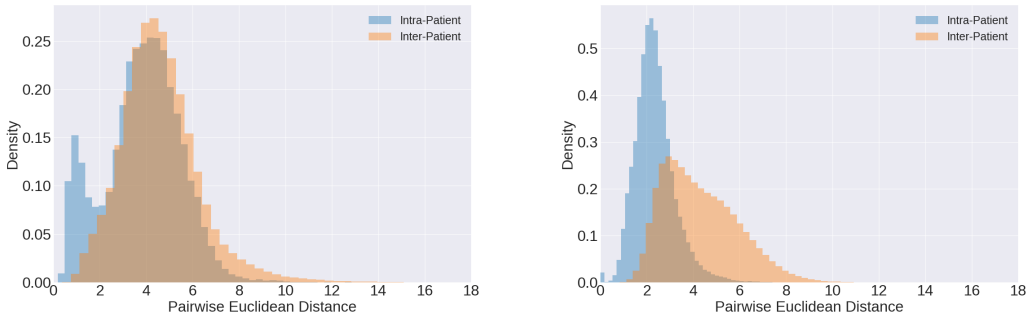
In Fig. 5.7b, we show that increasing the amount of labelled training data benefits the generalization performance of all methods. This can be seen by the increasing

AUC values as $F = 0.25 \rightarrow 1$. We also show that at all fraction values, CMSMLC outperforms its counterparts. For example, at $F = 1$, CMSMLC achieves an AUC $=$ 0.732 whereas SimCLR achieves an AUC $=$ 0.718. Such superiority still holds at $F = 0.25$ where the two methods achieve an AUC $=$ 0.675 and 0.652, respectively. This outcome emphasizes the robustness of CMSMLC to scarce labelled training data.

5.5.6 *Learning Patient-Specific Representations*

We had redefined 'shared context' to refer to representations from the same patient, which in turn should produce patient-specific representations. To validate this hypothesis, we calculate the pairwise Euclidean distance between representations of the same patient (Intra-Patient) and those of different patients (Inter-Patient). On average, the former should be smaller than the latter. In Fig. 5.8, we illustrate the two distributions associated with the intra and inter-patient distances at $E = 128$. We also find that increasing the embedding dimension shifts these distributions to higher values (see Appendix B.2.2).

We show that these two distributions have large mean values and overlap significantly when implementing SimCLR, as seen in Fig. 5.8a. This is expected as SimCLR is blind to the notion of a patient. In contrast, when implementing CMSC,



(a) SimCLR (b) CMSC

Figure 5.8: **Distribution of pairwise Euclidean distance between representations ($E = 128$) belonging to the same patient (Intra-Patient) and those belonging to different patients (Inter-Patient).** Self-supervision was performed on PhysioNet 2020. Notice the lower average intra-patient distance and improved separability between the two distributions with CMSC than with SimCLR.

the intra-patient distances are lower than those found in SimCLR, as seen in Fig. 5.8b. Moreover, the intra and inter-patient distributions are more separable. This implies that pre-training with CMSC leads to patient-specific representations. We note that this phenomenon takes place while concomitantly learning better representations, as observed in previous sections.

6

PATIENT-SPECIFIC CARDIAC ARRHYTHMIA DIAGNOSIS

If you wish to make an apple pie from scratch, you must first invent the universe

— Carl Sagan, Cosmos

In Part II so far, we have presented scenarios characterized by abundant, unlabelled data and scarce, labelled data. We designed and demonstrated the potential utility of active and contrastive learning methods to improve the diagnosis of cardiac arrhythmias with fewer resources (absolute amount of data and labels). However, as we continue to slide along the resource spectrum, we become capable of performing the task of cardiac arrhythmia diagnosis while *explicitly* leveraging patient information.

Modern medical research is arguably anchored around the 'gold standard' of evidence provided by randomized control trials (RCTs) (Cartwright, 2007). However, RCT-derived conclusions are population-based and fail to capture nuances at the individual patient level (Akobeng, 2005). This is primarily due to the complex mosaic that characterizes a patient from demographics, to physiological state, and treatment outcomes. Similarly, network-generated predictions remain population-based and difficult to interpret. Such properties are a consequence of a network's failure to incorporate patient-specific structure during training or inference. As a result, physicians are reluctant to integrate such systems into their clinical workflow. This reluctance is contrasted by the favourable perception of personalized medicine, the ability to deliver the right treatment to the right patient at the right time (Hamburg and Collins, 2010). As a result, we are interested in tackling the following question.

Research Question

How can we design clinical algorithms that efficiently exploit patient information to generate patient-specific diagnoses?

In this chapter, we address the outlined research question in the context of diagnosing cardiac arrhythmias based on the electrocardiogram. To achieve this, we conceptually borrow insight from the field of personalized medicine. Formally, we learn patient-specific embedding, entitled patient cardiac prototypes (PCPs), that efficiently summarize the cardiac state of a patient. With these embeddings in mind, our contributions are threefold. First, we illustrate the potential of PCPs to be exploited for personalized cardiac arrhythmia diagnosis. Second, we show that PCPs are efficient dataset distillers. In other words, they are a compact substitute for, and can be used in lieu of, the original dataset to train a network and maintain strong generalization performance. Lastly, we show that PCPs can be used to discover similar and dissimilar patients both within and across distinct datasets.

6.1 RELATED WORK

Contrastive learning. Contrastive learning is a self-supervised method that encourages representations of instances with commonalities to be similar to one another (please refer to Section 5.2 for further background on contrastive learning). This is performed for each instance and its perturbed counterpart (Oord et al., 2018; Chen et al., 2020a,c; Grill et al., 2020) and for different visual modalities (views) of the same instance (Tian et al., 2019). Such approaches are overly-reliant on the choice of perturbations and necessitate a large number of comparisons. Instead, Caron et al. (2020) propose to learn prototypes to cluster images. Most similar to our work is that of Cheng et al. (2020) which both show the benefit of encouraging patient-specific representations to be similar to one another.

Metric-learning. Metric-learning is a subfield of machine learning in which the similarity of representations is quantified and learned. Although it has recently

been used in the design of *meta*-learning algorithms, the two notions are distinct from one another. The latter is primarily interested in designing algorithms that adapt quickly and perform well on unseen tasks given few datapoints (Vilalta and Drissi, 2002). For example, Prototypical Networks (Snell et al., 2017) average instance representations to obtain label-specific prototypes. During inference, the similarity of instance representations to these prototypes (e.g., via Euclidean distance) determines the classification. Relational Networks (Sung et al., 2018) build on this idea by tasking an actual neural network to learn the similarity of representations to prototypes. Analogies have also been created between prototypes and the parameters of the final classification head in a network. To that end, Gidaris and Komodakis (2018) and Qiao et al. (2018) exploit hypernetworks (Ha et al., 2016), functions whose purpose is to *generate* parameters, and propose to generate linear classification parameters for few-shot learning on visual tasks. In contrast to previous work, we learn patient-specific embeddings in an end-to-end manner. Additionally, during inference, we compute the cosine similarity between representations and these embeddings, not to perform the final classification but rather to inform the input of a hypernetwork.

Patient similarity. Patient similarity aims to discover relationships between patient data (Sharafoddini et al., 2017). To quantify these relationships, Pai and Bader (2018) and (Pai et al., 2019) propose Patient Similarity Networks for cancer survival classification. Exploiting electronic health record data, Zhu et al. (2016) use Word2Vec to learn patient representations, and Suo et al. (2017) propose to exploit patient similarity to guide the re-training of models, an approach which is computationally expensive. Instead, our work naturally learns PCPs as efficient descriptors of the cardiac state of a patient.

6.2.1 *Learning Patient Cardiac Prototypes via Contrastive Learning*

Let us assume we have a dataset, $\mathcal{D} = \{\boldsymbol{x}_i, y_i\}_{i=1}^{N}$, comprising N instances, $\boldsymbol{x}$, and cardiac arrhythmia labels, y, corresponding to a total of Ω_{train} patients in the training set. Each patient is associated with $N/\Omega_{\text{train}} > 1$ instances. This could be due to the provision of multiple medical tests during the same hospital visit or several visits. We also define a learner, $f_\theta : \boldsymbol{x} \in \mathbb{R}^D \rightarrow \boldsymbol{h} \in \mathbb{R}^E$, parameterized by $\boldsymbol{\theta}$, that maps a D-dimensional instance, $\boldsymbol{x}$, to an E-dimensional representation, $\boldsymbol{h}$. We aim to learn a set of embeddings, each of which efficiently summarizes the cardiac state of a patient. To that end, we associate each patient in the training set with a unique, learnable embedding, $\boldsymbol{p} \in \mathbb{R}^E$, to form the set of embeddings, $P = \{\boldsymbol{p}_j\}_{j=1}^{\Omega_{train}}$. Hereafter, we refer to such embeddings as patient cardiac prototypes (PCPs). We next explain how to learn PCPs such that they become patient specific.

To make PCPs patient-specific, we exploit the contrastive learning framework which comprises a sequence of attractions and repulsions. The idea is to attract representations of instances of the same patient to the single PCP of that same patient, and to repel them from the PCPs of the remaining patients. Formally, we encourage the representation, $\boldsymbol{h}_i = f_\theta(\boldsymbol{x}_i)$, of an instance, $\boldsymbol{x}_i$, associated with the k-th patient to be similar to the k-th PCP, $\boldsymbol{p}_k$, and dissimilar from the remaining PCPs, $\boldsymbol{p}_j$, $j \neq k$. To capture this behaviour, we exploit the InfoNCE loss (6.1). Intuitively, it penalizes the learner for placing less probability mass on the similarity, $s(\boldsymbol{h}_i, \boldsymbol{p}_k)$, of the representation and prototype pair that should be most similar (based on patient ID) than on the similarity of other pairs, of which there are $\Omega_{train} - 1$. We quantify the cosine similarity between such pairs with a temperature parameter, τ.

$$\mathcal{L}_{NCE} = -\sum_{i=1}^{B} \log \left[\frac{e^{s(\boldsymbol{h}_i, \boldsymbol{p}_k)}}{\sum_j^{\Omega_{train}} e^{s(\boldsymbol{h}_i, \boldsymbol{p}_j)}} \right] \qquad s(\boldsymbol{h}_i, \boldsymbol{p}_j) = \frac{\boldsymbol{h}_i \cdot \boldsymbol{p}_j}{\|\boldsymbol{h}_i\| \|\boldsymbol{p}_j\|} \cdot \frac{1}{\tau} \tag{6.1}$$

As a result of this many-to-one mapping from representations to PCP, the latter will become invariant to *intra-patient* differences present in the data. This outcome is desirable only if we assume that such intra-patient differences point to the same underlying physiological state of the patient. One way to satisfy this assumption is by, for example, exclusively considering patient data that spans a short time-frame (e.g., seconds).

6.2.2 *Generating Patient-Specific Parameters via Hypernetworks*

Equipped with PCPs, we direct our attention to the question of 'how do we exploit PCPs to generate patient-specific diagnoses?' Before addressing this question, we note that neural network parameters are often deterministic; they are held constant during inference. Therefore, when making a prediction (e.g., medical diagnosis), a network depends almost exclusively on the instance. Although an instance is likely to reflect patient information, a network does *not* explicitly exploit such information. In light of this, we design a framework in which a subset of network parameters, instead of being deterministic, are conditioned on patient information. We achieve this by exploiting hypernetworks (Ha et al., 2016) and our PCPs, as explained next.

A hypernetwork, a neural network in and of itself, generates parameters for another neural network. Formally, a hypernetwork is a function, $g_\phi : \boldsymbol{h} \in \mathbb{R}^E \rightarrow \boldsymbol{\omega} \in \mathbb{R}^{E \times C}$, parameterized by $\boldsymbol{\phi}$, that maps an E-dimensional representation, $\boldsymbol{h}$, to a matrix of parameters, $\boldsymbol{\omega}$, where C is the number of class labels. The output parameters, $\boldsymbol{\omega}$, can now be used to parameterize a linear classification head, $p_\omega : \boldsymbol{h} \in \mathbb{R}^E \rightarrow \hat{y} \in \mathbb{R}^C$, which maps an E-dimensional representation, $\boldsymbol{h}$, to an output probability distribution, $\hat{y}$. To condition the parameters, $\boldsymbol{\omega}$, on patient information, we exploit PCPs differently during the training and inference stages of the framework, as outlined next.

Training stage. During training, each representation, $\boldsymbol{h}_i = f_\theta(\boldsymbol{x}_i)$, of an instance, $\boldsymbol{x}_i$, serves multiple purposes (see Fig. 6.1 left). First, it is attracted to its corresponding PCP, $\boldsymbol{p}_k$, as outlined earlier. To do so, we optimize the InfoNCE loss. Second, it is used

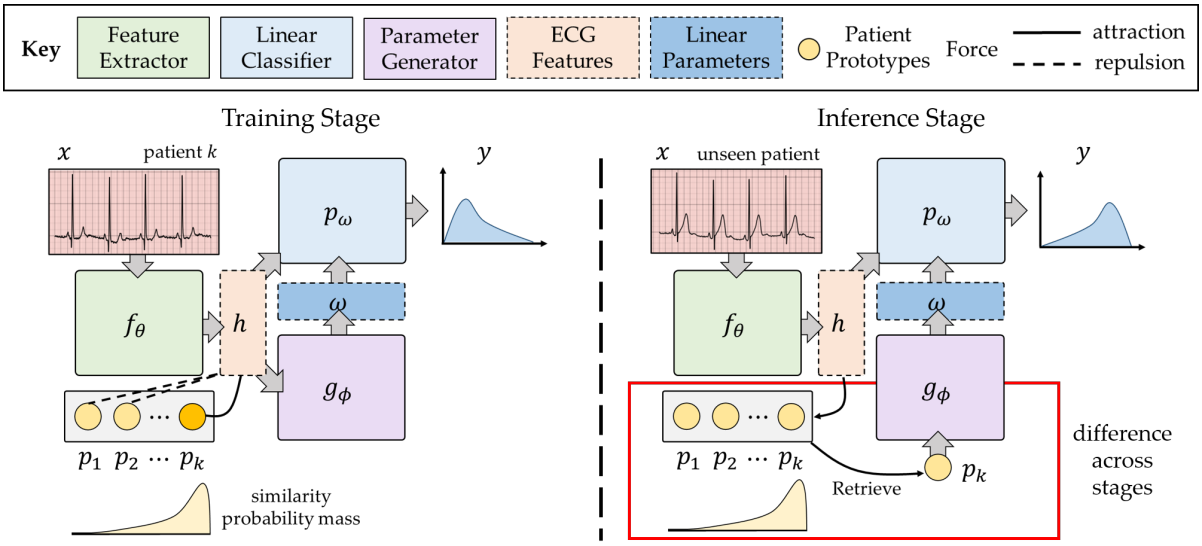


Figure 6.1: **Training and inference stages of the personalized diagnosis pipeline.** (**Training stage**) a representation, $\boldsymbol{h}$, associated with patient k is a) fed into a hypernetwork, g_ϕ, to generate parameters, ω, for a linear classification layer, p_ω, b) input directly into p_ω to output a class probability distribution, and c) encouraged to be similar to the corresponding patient cardiac prototype (PCP), $\boldsymbol{p}_k$. (**Inference stage**) the representation of an instance associated with an unseen patient retrieves the nearest PCP, $\boldsymbol{p}_k$, which is then input into the hypernetwork. This generates patient-specific linear parameters for classification.

as an input to the hypernetwork to generate *instance-specific* parameters, $\boldsymbol{\omega}_i = g_\phi(\boldsymbol{h}_i)$. Note that our goal of generating *patient-specific* parameters is of most value during inference (explained next). Third, the representation is input into the classification head, p_ω, as is usual with neural networks. Given the ground-truth disease class, c, of each instance in a mini-batch of size B, we can optimize the categorical cross-entropy loss ($\mathcal{L}_{CE}$). In summary, during the training stage, we learn the parameters of the feature extractor, $\boldsymbol{\theta}$, the hypernetwork, $\boldsymbol{\phi}$, and the PCPs, $\{\boldsymbol{p}_j\}_{j=1}^{\Omega_{train}}$, in an end-to-end manner by optimizing the combined loss ($\mathcal{L}_{combined}$).

$$\mathcal{L}_{CE} = -\sum_{i=1}^{B} \log p_{\omega_i}(\boldsymbol{y_i} = c|\boldsymbol{h_i}) \qquad \mathcal{L}_{combined} = \mathcal{L}_{CE} + \mathcal{L}_{NCE} \tag{6.2}$$

Inference stage. During the inference stage, in which patient-specific diagnoses are of most value, we propose several modifications to the pipeline (see Fig. 6.1 right). First, each representation, $\boldsymbol{h}_i$, is no longer attracted to or repelled from PCPs. This is primarily because the patients in the inference stage do *not* overlap with those in the

training stage; they are mutually exclusive by design. Instead, each representation, $\boldsymbol{h}_i$, searches through the set of PCPs, P, and retrieves the single PCP to which it is closest, $\boldsymbol{p}_k = \text{argmax}_{\boldsymbol{p}_j} \; s(\boldsymbol{h}_i, \boldsymbol{p}_j)$, where s is some similarity metric such as cosine similarity. Second, the retrieved PCP, $\boldsymbol{p}_k$, (instead of $\boldsymbol{h}_i$, as was done during training) is now used as an input to the hypernetwork, g_ϕ. As a result, we generate *patient-specific* parameters, $\boldsymbol{\omega}_i = g_\phi(\boldsymbol{p}_k)$, for each unseen instance in the held-out set of data. More precisely, we generate parameters conditioned on the single PCP associated with the patient in the training set that is deemed most similar to the patient observed during inference. In summary, the patient-specific diagnosis is underpinned by this retrieval and parameter-generation process.

6.3 EXPERIMENTAL DESIGN

6.3.1 *Data and Pre-processing*

To evaluate our method, we leverage three different datasets, each of which consists of cardiac time-series waveforms alongside cardiac arrhythmia labels. We split each of the aforementioned waveforms into non-overlapping segments comprising 2500 samples. In Table 6.1, we present a summary of these datasets.

Table 6.1: **Summary of the datasets used for evaluation.** We also show additional pre-processing information. Please click on the dataset's name for more information.

Dataset	Abbreviation	Modality	Normalization	Additional Info.
PhysioNet 2020	$\mathcal{D}_1$	ECG	✓	-
Chapman	$\mathcal{D}_2$	ECG	✓	-
PTB-XL	$\mathcal{D}_3$	ECG	✓	2-way classification

6.4 RESULTS

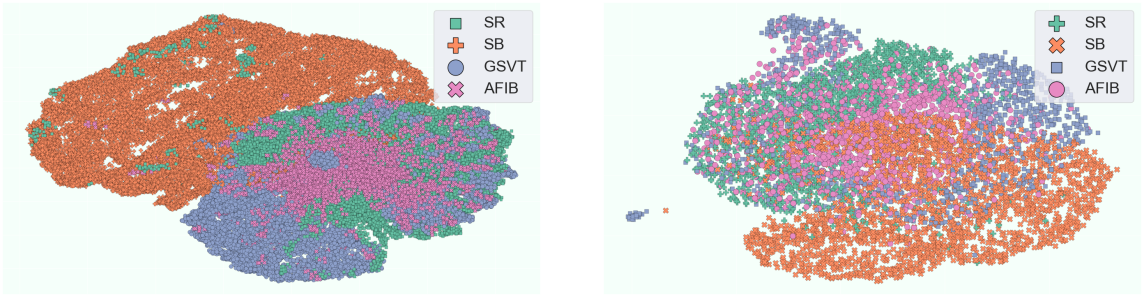
In this section, and with an eye on addressing the initial research question, we attempt to answer the following questions: (i) Are patient cardiac prototypes distinct and patient-specific? (ii) To what extent does the retrieval mechanism impact the performance of the network? (iii) How can we exploit patient cardiac prototypes

to perform (iv) patient-specific diagnosis, (v) dataset distillation, and (vi) patient retrieval?

6.4.1 *Visualization of Patient Cardiac Prototypes*

We designed patient cardiac prototypes purposefully to allow for the diagnosis of medical conditions in a patient-specific manner, the retrieval of patients that are similar to one another, and the distillation of datasets. To determine whether these goals could be achieved, we first wanted to visualize the patient cardiac prototypes that were learned by our framework. Concretely, we were looking to address two questions. First, *are patient cardiac prototypes distinct from one another and discriminative along the dimension of disease class?* The importance of learning *distinct* patient cardiac prototypes can be explained as follows. Recall that predictions during inference are dependent upon linear parameters generated by a deterministic hypernetwork conditioned on such prototypes. Therefore, if prototypes were to collapse to a single point, a process which we refer to as mode-collapse, then linear parameters will also experience this detrimental outcome. With the same linear parameters being generated regardless of the input patient data, the former becomes independent of the latter and thus will *not* be patient-specific.

In Fig. 6.2, we illustrate the two-dimensional UMAP projection (McInnes et al., 2018) of representations of instances in the training set and patient cardiac prototypes. We show that our framework learns patient cardiac prototypes that are distinct from one another. This can be seen by the diversity (and thus lack of mode-collapse) of the PCPs in Fig. 6.2b. As outlined earlier, this is a healthy outcome. Furthermore, we show that PCPs are also discriminative along the dimension of cardiac arrhythmia disease class. This can be seen by the separability of the PCPs that are reflected by distinct colours and shapes. Note that since each PCP reflects an individual patient, and that each patient may have multiple instances (and thus representations), there



(a) Representations in training set

(b) Patient cardiac prototypes (PCPs)

Figure 6.2: **Two-dimensional UMAP projection of (a) representations, h, of instances in the training set of the Chapman dataset and (b) PCPs, p, learned on the training set.** The colours reflect four cardiac arrhythmia labels, Sinus Rhythm (SR), Sinus Bradycardia (SB), GSVT, and Atrial Fibrillation (AFIB). We show that PCPs are distinct and class-discriminative.

are fewer PCPs than representations. This explains the relative sparsity of projections in Fig. 6.2b compared to those in Fig. 6.2a.

The Chapman dataset comprises four high-level cardiac arrhythmia classes. Putting aside the imperfections of visualizations of learned representations (e.g., with t-SNE and UMAP), there could be two main reasons for the observed overlap between instances from a different class. First, from a clinical perspective, some of these classes, although distinct at a high-level, do share similarities. For example, this is the case with GSVT (supra-ventricular tachycardia) and AFIB (atrial fibrillation), where the latter can be considered to be a specific case of a supra-ventricular tachycardia. Therefore, we would expect some overlap in this case. Second, the degree of overlap is also highly dependent on the network architecture that is used to learn representations. For example, a more sophisticated architecture could learn representations that are more discriminative along the class dimension and would thus appear to be more separable when visualized. Throughout the thesis, we do not experiment with such sophisticated architectures and instead opt for simple ones that manage to achieve the task of cardiac arrhythmia diagnosis.

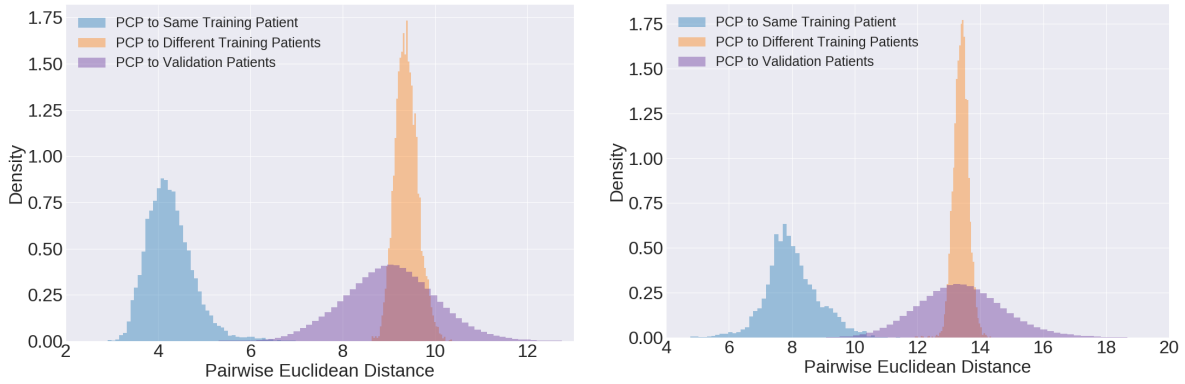
We showed that patient cardiac prototypes are *distinct* and *discriminative* along the dimension of disease class. Although promising, these findings do not directly address our initial claim that prototypes are *patient-specific*. In other words, it could

still be argued that patient cardiac prototypes are merely reflecting variations in the data along the dimension of disease class and nothing more. The main repercussion of such a claim is that the linear parameters generated by the hypernetwork would be *class-specific* instead of *patient-specific*. To explore this possibility, we pose our second question: *are patient cardiac prototypes truly patient-specific, as we have claimed?*

We address this question quantitatively as follows. We calculate the distance between each PCP-and-representation pair. Note that each representation of an instance, whether in the training or held-out set, is also associated with a patient annotation. Therefore, we could stratify these distance values based on whether the PCP-and-representation pair corresponded to the same, or a different, patient annotation. We thus refer to the distance values in these two groups as **PCP to Same Training Patient** and **PCP to Different Training Patient** and present their distributions in Fig. 6.3 for the three datasets, Chapman, PhysioNet 2020, and PTB-XL. Intuitively, evidence in support of patient-specific PCPs would manifest in the form of patient cardiac prototypes that were more similar to representations with the same patient annotation than to those with a different patient annotation.

We show that, regardless of the dataset used for training, PCPs are indeed patient-specific. This can be seen by the smaller distance values exhibited by the group PCP to Same Training Patient than those exhibited by the group PCP to Different Training Patient. For example, in Fig. 6.3a, these two distributions have an average value of 4 and 9, respectively. This suggests that, in this case, PCPs are twice as similar to representations of the same patient than to representations of a different patient.

Recall that during the inference stage, we implement a retrieval mechanism whereby representations of *unseen* instances are used to retrieve the single PCP to which they are closest. For this retrieval mechanism to work, the distance between such representations and PCPs have to be reasonable and on the same order of magnitude. To evaluate this, we calculate this distance from **PCP to Validation Patients** and also plot them in Fig. 6.3. We show that these distance values are



(a) Chapman

(b) PhysioNet 2020

Figure 6.3: **Distribution of pairwise Euclidean distance from the learned PCPs to various sets of representations from three distinct datasets. PCP to Same Training Patient** reflects the distance between PCPs and representations of instances in the training set associated with the same patient annotation. **PCP to Different Training Patient** reflects the distance between PCPs and representations in the training set associated with a different patient annotation. **PCP to Validation Patients** reflects the distance between PCPs and representations in the validation set. We show that PCPs are patient-specific, as exemplified by the separability of the two distributions, PCP to Same Training Patient and PCP to Different Training Patient.

indeed reasonable. This is supported by the observation that they are on the same order of magnitude as the distance values exhibited by the group PCP to Different Training Patients. It is worthwhile to note that the minimal overlap between the distributions PCP to Same Training Patient and PCP to Validation Patients reaffirms that the patients in the training and validation sets of the dataset do *not* overlap, which was by design.

6.4.2 *Diagnosis with Different Retrieval Mechanisms*

To make predictions during the inference stage, our framework involved the retrieval of the *single* PCP that is closest to the representation of an unseen instance. This retrieval mechanism is a critical component of our framework. To see this, note that the retrieval of an inappropriate PCP would have ramifications on the linear parameters that are generated by the hypernetwork, and, in turn, the disease diagnosis. To quantify the importance of retrieval, we explore four variants of the retrieval

mechanisms that differ in the extent to which they are patient-specific, as outlined next.

The first, which we refer to as **Mean**, involves taking the average of all PCPs, $\bar{\boldsymbol{p}} = \sum_{j=1}^{\Omega_{train}} \boldsymbol{p}_j$, regardless of the instance in the held-out set. Since PCPs and the hypernetwork are deterministic during the inference stage, this approach implies that the generated linear parameters are effectively reduced to a constant (not patient-specific). The second mechanism, which we refer to as **Similarity-Weighted Mean**, involves calculating a linear combination of the PCPs, weighted according to their similarity to the representation, $\boldsymbol{h}_i$, of an instance. Formally, $\bar{\boldsymbol{p}} = \sum_{j=1}^{\Omega_{train}} s(\boldsymbol{p}_j, \boldsymbol{h}_i) \cdot \boldsymbol{p}_j$. In effect, this approach exploits patient information to down-weight, or up-weight, the contribution of each PCP. The third mechanism, which we refer to as **Nearest**, involves the vanilla approach of retrieving the *single* PCP closest to the representation, $\boldsymbol{h}_i$. In effect, this approach retrieves the PCP of the patient in the training set that is deemed most similar to the representation of an instance associated with a different patient in the held-out set. We hypothesize that by restricting the framework to only select a single PCP, we were preventing the representation from exploiting potentially useful information contained in additional PCPs. For example, these additional PCPs could reflect patients with attributes (e.g., sex, age, and treatment outcome) that are shared with, and thus potentially useful for, the patient in the held-out set for whom the prediction is being made. To account for this information, our fourth variant of the retrieval mechanism, which we refer to as **Nearest 10**, involves taking the average of the ten PCPs closest to the representation, $\boldsymbol{h}_i$.

In Fig. 6.4, we present the AUC of our framework when deploying these variants of the retrieval mechanism as a function of the embedding dimension, E, of the PCPs. We show that incorporating imprecise patient information, in the form of the Mean approach, hinders the generalization performance of the network. For example, Mean achieved AUC < 0.65 across the embedding dimensions. This is significantly lower than the generalization performance achieved by other variants of the retrieval mech-

anism. However, after having incorporated some patient information, as reflected by the Similarity − Weighted Mean approach, we show that performance improves significantly. For example, at $E = 128$, Mean and Similarity − Weighted Mean achieved AUC ≈ 0.58 and 0.73, respectively. This suggests that the PCP-derived similarity coefficients in the latter approach were beneficial, thus lending support to the involvement of PCPs in the diagnosis pipeline.

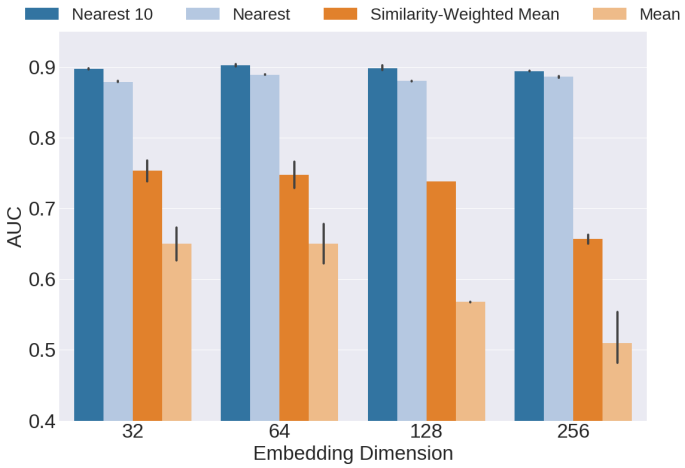


Figure 6.4: **Effect of variants of the PCP retrieval mechanism on the AUC of the test set in Chapman dataset as a function of the embedding dimension,** E**.** Results are averaged, and the error bars reflect one standard deviation, across five seeds. We show that Nearest 10 outperformed its counterparts and is less affected by changes to the embedding dimension.

We continue to test the limits of the retrieval mechanism and limit it to retrieving a *single* PCP. We show that individual PCPs are very precise and relevant for diagnosis during the inference stage. This can be seen by the strong generalization performance achieved by the Nearest approach. For example, at $E = 64$, Similarity − Weighted Mean and Nearest achieve AUC ≈ 0.75 and 0.89, respectively. Furthermore, and as expected, incorporating additional information from a subset of patients, as reflected by Nearest 10, further improves performance, albeit in a more marginal way.

6.4.3 *Interpretable Error Analysis*

In addition to making patient-specific diagnoses, our framework and the associated patient cardiac prototypes allow for a higher degree of interpretability when such diagnoses are made. To emphasize this point, recall that during inference, the representation of an unseen instance *retrieves* the PCP to which it is closest. This PCP is, in turn, fed into a hypernetwork to generate parameters (for a linear classification head) that directly influence the final diagnosis. Therefore, each diagnosis can now be traced back to the PCP that was retrieved. This PCP (and its associated patient data) can be further inspected to help researchers better understand the diagnosis made by the network. For example, in the event of a correct diagnosis, inspection of the patient data associated with the retrieved PCP can act as a sanity check that the network is depending on clinically-relevant patients when making a diagnosis. On the other hand, and in the event of an incorrect diagnosis, inspection of the patient data associated with the retrieved PCP can lend insight into both *why* and *how* a network failed for this particular instance.

To gain a better understanding of these claims, we present in Fig. 6.5, the 12-lead ECG data in addition to patient data associated with an unseen instance in the validation set and that associated with the PCP retrieved during inference. Whereas the top row illustrates a network prediction which is correct, hence the green box, the bottom row illustrates an incorrect prediction, hence the red box.

In Fig. 6.5a, we show that when making a correct prediction for an unseen patient in the validation set, our framework depends on relevant PCPs. This is evident by the observation that the patient data associated with the retrieved PCP is highly relevant and similar to that associated with the unseen instance in the validation set. For example, both exhibit a ground-truth label of sinus bradycardia, reflect patients of a similar age (60's), and outline similar cardiac-specific statistics (e.g., ventricular rate

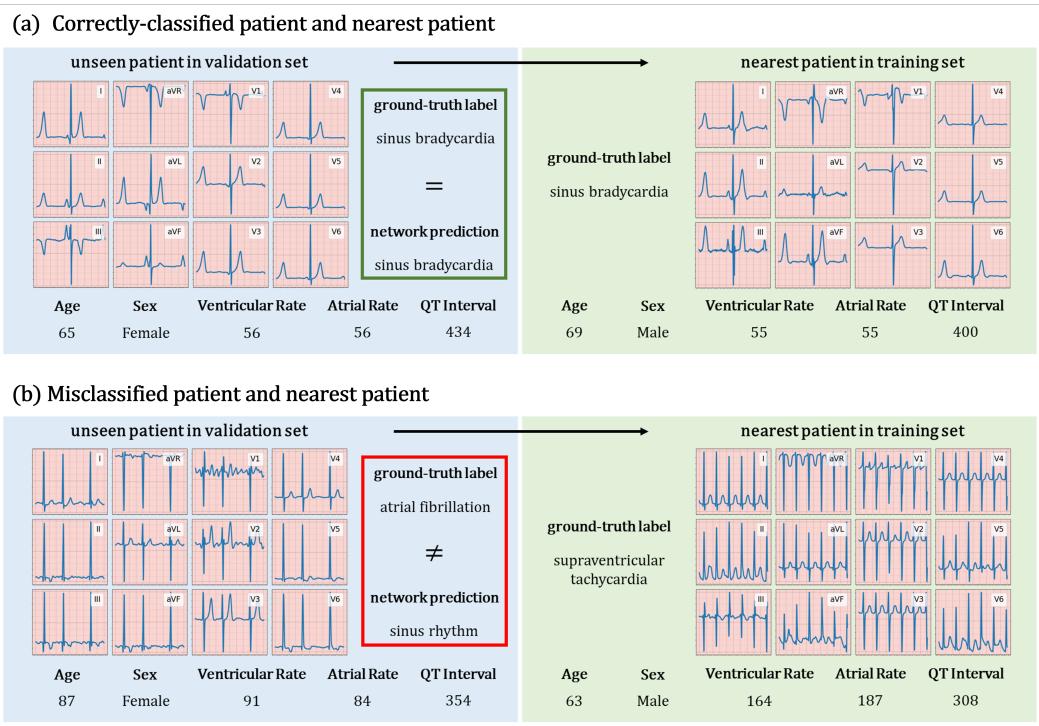


Figure 6.5: **Predictions made by the network can be tracked back to the PCP that was retrieved during inference.** We show the 12-lead ECG and patient data associated with a **(a)** correctly-classified patient and **(b)** misclassified patient from the validation set and that associated with the nearest PCP retrieved during inference. This process of tracing predictions back to the PCP retrieved during inference allows for improved network interpretability, and thus allows researchers to determine *why* a certain diagnosis was made.

and atrial rate ≈ 55). The ability to conduct such an analysis allows researchers to be more confident in the diagnoses that are being made by the network.

In Fig. 6.5b, we show that when making an incorrect prediction for an unseen patient in the validation set, our framework could be depending on irrelevant PCPs. This is evident by the observation that patient data associated with the retrieved PCP is distinct from that associated with the unseen instance in the validation set. For example, they exhibit the distinct ground-truth cardiac arrhythmia labels of supraventricular tachycardia and atrial fibrillation, respectively. Moreover, these patients differ in their age (63 vs. 87), sex, and other cardiac-specific statistics (e.g., ventricular rate = 164 and 91, respectively). As a result, we now have a better understanding of *why* the network failed to make the correct diagnosis. Inspecting such errors at the population level may also allow researchers to capture trends in

the errors generated by a network. Overall, these findings demonstrate that our framework and the associated PCPs can allow for more interpretable error analysis. This, in turn, could contribute to workarounds that are more targeted.

6.4.4 *Dataset Distillation with Patient Cardiac Prototypes*

Our deep learning framework, as described so far, receives raw cardiac signals as input and learns patient cardiac prototypes as intermediaries. Previous sections also showed that such prototypes are distinct, class-discriminative, and patient-specific. In light of this, and the observation that there are a fewer number of prototypes than instances, we hypothesize that PCPs could be exploited as a core-set. A core-set typically comprises a compact subset of the labelled, training data which, when trained on by a network, allows for strong generalization performance. Therefore, we pose the question, *to what extent can a network trained exclusively on PCPs parallel the performance of one trained on instances in the larger, labelled dataset?*

To address this question, we first train our framework, as per usual, to learn the prototypes. We then fit a machine learning model (e.g., support vector machine, random forest) to these prototypes and evaluate its performance using representations of instances in the held-out set of data. In other words, a model is *trained* on PCPs and *evaluated* on representations of instances. Note that this is possible because we used the same dimensionality, E, for the PCPs and the representations. We compare our framework to several state-of-the-art core-set construction methods, which we refer to as **Lucic** (Lucic et al., 2016), **Lightweight** (Bachem et al., 2018), and **Archetypal** (Mair and Brefeld, 2019). In short, these methods can identify a core-set of any dataset that they are provided with. As a result, we deploy these methods independently on either raw instances or representations, $\boldsymbol{h}$, learned via our framework. As outlined earlier, a machine learning model is trained on the obtained core-set and evaluated on the held-out set of data.

In Table 6.2, we present the generalization performance of models that have been trained on various core-sets and evaluated on either raw instances or representations of such instances in a held-out set of data. We show that core-sets of raw instances, generated by baseline core-set construction methods, do not provide a sufficient training signal to allow machine learning models to achieve strong generalization performance. For example, on the Chapman dataset, Lucic, Lightweight, and Archetypal achieved AUC $=$ 56.8, 56.6 and 54.8, respectively. We attribute this performance to the low class separability of the input features. However, the baseline methods continue to perform poorly even when provided with the opportunity to construct core-sets from representations learned via our framework. Recall that these representations are more separable along the disease class dimension (see Fig. 6.2). For example, on the Chapman dataset, Lucic, Lightweight, and Archetypal achieve AUC $=$ 57.8, 58.9 and 58.1, respectively. In contrast, PCPs constitute a more effective core-set, achieving AUC $=$ 88.7. These findings suggest that PCPs are effective dataset distillers.

Table 6.2: **AUC of machine learning model trained on various core-sets and evaluated on held-out set of data.** For the Chapman and PTB-XL datasets, we train a support vector machine. For CPSC, we train a random forest since it comprises a multi-label classification task. For the baseline methods, the core-set size was chosen to equal total number of PCPs. Mean and standard deviation (SD) are shown across five seeds. Note that since the raw instances of PTB-XL are 12-lead ECG signals, they could not be used with an SVM. We show that PCPs outperform baseline state-of-the-art core-set construction methods.

Core-set Method	Chapman	PhysioNet 2020	PTB-XL
Raw Instances			
Lucic (Lucic et al., 2016)	56.8 (0.8)	50.1 (0.1)	-
Lightweight (Bachem et al., 2018)	56.6 (0.4)	50.1 (0.1)	-
Archetypal (Mair and Brefeld, 2019)	54.8 (0.3)	50.1 (0.1)	-
Representations			
Lucic (Lucic et al., 2016)	57.8 (17.5)	50.6 (1.2)	51.6 (4.5)
Lightweight (Bachem et al., 2018)	58.9 (16.8)	50.5 (1.2)	52.4 (3.6)
Archetypal (Mair and Brefeld, 2019)	58.1 (16.8)	50.5 (1.2)	51.0 (5.0)
PCPs	**88.7** (0.5)	**52.8** (0.1)	**63.5** (0.7)

Having shown the effectiveness of PCPs as dataset distillers, we also wanted to investigate the extent to which further distillation was possible. Specifically, we randomly choose a fraction, $F \in [0.05, 0.1, 0.2, 0.5]$, of the PCPs and trained a machine

learning model on that subset instead. We continued to evaluate on the same held-out set of data, as before. In Fig. 6.6, we illustrate the generalization performance of models trained on these subsets alongside the performance of a model trained on all instances in the larger, original dataset (horizontal, dashed line). Note that original dataset is several folds larger than the total number of PCPs.

In Fig. 6.6, we provide further evidence that PCPs are effective dataset distillers. For example, when training on 100% of the PCPs ($\Omega_{train} = 6,387$), an SVM model achieve similar performance (AUC $\approx$ 0.89) to one trained on all instances in the training set ($N = 76,614$). In other words, similar generalization performance is achieved *despite* a *12-fold* reduction in the number of training instances provided to the model. We also show that more extreme distillation does not significantly hinder performance. For example, an SVM model trained with only 5% of the available PCPs ($\Omega_{train} = 319$) achieved AUC $\approx$ 0.82. Expressed differently, this corresponds to a 7% drop in performance despite a *240-fold* reduction (relative to training on all instances) in the number of training instances provided to the model. Such a finding reaffirms the potential of PCPs at dataset distillers. We hypothesize that this behaviour arises due to our patient-centric contrastive learning framework. PCPs, encouraged to be similar to representations of instances associated with the same patient, are able to capture the most pertinent information in patient data and avoid that which is least useful for solving the task.

6.4.5 *Patient Retrieval with Patient Cardiac Prototypes*

In addition to providing medical researchers with the ability to perform patient-specific diagnoses and to potentially reduce the size of the data used to train models, we also explore whether PCPs can provide physicians and medical educators with a tool for patient retrieval. Patient retrieval typically involves identifying similar (and dissimilar) patients within a clinical database. Therefore, we posed the ambitious question of, *can patient cardiac prototypes be exploited to retrieve similar and dissimilar*

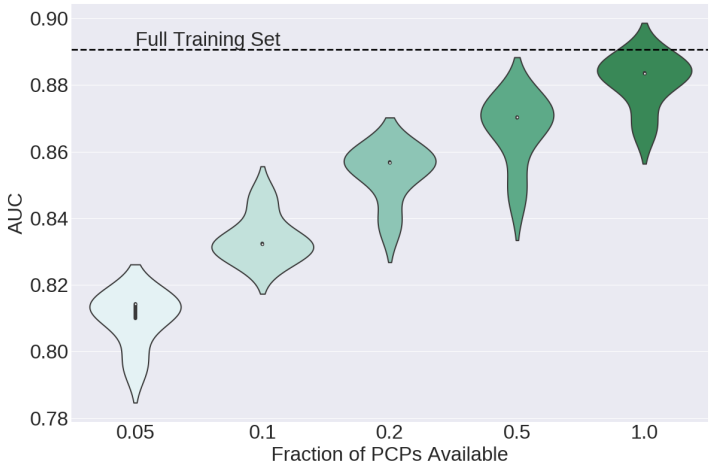


Figure 6.6: **AUC of SVM model trained on a fraction, F, of the PCPs and evaluated on a held-out set of data.** Results are averaged across five random seeds. The horizontal dashed line depicts the performance of an SVM trained on all *instances* ($N = 76644$) in the training set. We show that PCPs are effective dataset distillers; despite a 12-fold reduction in the number of training instances ($F = 1$, $\Omega_{train} = 6,387$), the SVM achieved similar performance (AUC ≈ 0.89) to one trained on all instances.

patient both within and across distinct clinical datasets? Note the challenging task of patient retrieval *across* datasets.

We first address this question to identify *similar* patients both within and across datasets. While both of these retrieval settings are non-trivial, the latter is, arguably, more challenging due to data variations present across datasets. When we searched within a dataset, we first calculate the Euclidean distance, $d(\boldsymbol{p}_j, \boldsymbol{h}_i) \in \mathbb{R}$ between the j-th PCP, $\boldsymbol{p}_j$, and the representation, $\boldsymbol{h}_i$, of the i-th instance unseen during training (e.g., those in the validation set). At this point, we have at our disposal distances between patient and representations. However, to obtain distances between patients and other patients, we average the distance values across representations, $\boldsymbol{h}_i$, associated with the same patient annotation. This process is then repeated for all PCPs. In contrast, when we searched across datasets, we simply calculate the Euclidean distance between the PCPs of these respective datasets. This immediately provides us with patient-patient distance values.

Evaluating the relevance of patients retrieved by a patient retrieval system is non-trivial. This is because the similarity (and dissimilarity) of patients from a clinical

perspective is nebulous. For example, patients can be deemed similar based on attributes such as sex and age, their medical history (e.g., cancer survivor), or drug treatment pathways. Unfortunately, publicly-available datasets of cardiac signals do not contain this information. Such datasets, do, however, comprise cardiac arrhythmia disease labels. Therefore, each pair of patients is assigned a ground-truth relevance score ($s = 1$ relevant, $s = 0$ irrelevant) according to whether they shared the same cardiac arrhythmia label. We then identify pairs of patients as being relevant if their Euclidean distance, $d < d_E$, was less than some threshold distance, d_E. If, however, we were interested in retrieving dissimilar patients, then we would simply need to redefine the ground-truth relevance score and identify pairs of patients with $d > d_E$ as being relevant (notice the swap in the sign).

Using the above formulation, we calculate the precision and negative predictive value of our patient retrieval system. For the precision, we are looking to quantify how many of the retrieved patients (those identified as relevant) were actually relevant. In contrast, for the NPV, we are looking to quantify how many of the retrieved patients (those identified as irrelevant) were actually irrelevant. In Fig. 6.7, we depict both the precision and NPV of the PCP-derived patient retrieval system.

We find that PCP-derived similarity values are able to identify patients with matching cardiac arrhythmia labels. For example, $> 90\%$ of the pairs of patients that were deemed very similar to one another (i.e., $d_E < 6.2$) exhibit a perfect cardiac arrhythmia label match. As we increase the threshold distance, $d_E \rightarrow 8.5$, we see that Precision $\rightarrow 0.3$. Such a decay is expected of a reasonable similarity metric where patients that are deemed dissimilar do not match according to their cardiac arrhythmia labels. Moreover, based on an acceptable level of precision, (e.g., 0.90), we can identify an appropriate threshold distance (e.g., $d_E \approx 6.2$) below which patients have a high probability of being similar. Building on this, we define a region of similarity where patients identified as being similar are highly likely of actually being so. It is also worthwhile to note that the specific threshold shown coincides with the

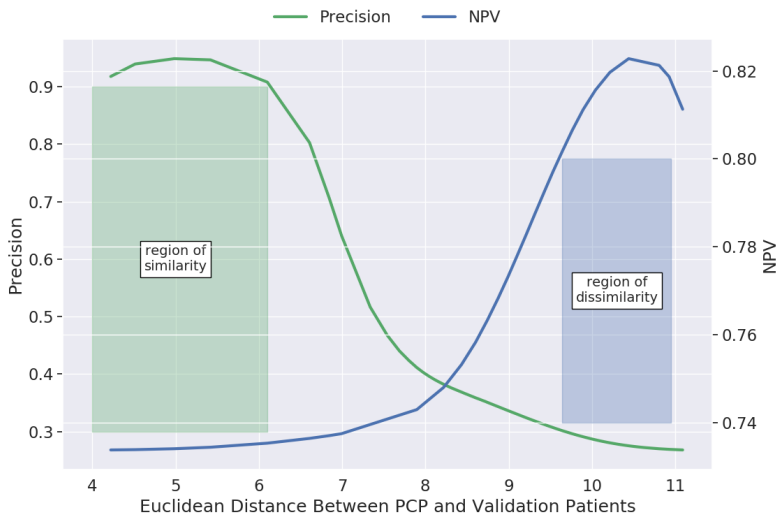


Figure 6.7: **Precision and negative predictive value of a system that identifies a pair of patients in the Chapman dataset as being similar or dissimilar based on whether their pairwise distance does not exceed, or exceeds, some threshold Euclidean distance, respectively.** For a given level of precision (e.g., 0.90), we can identify an appropriate threshold distance between patients (e.g., $d < d_E = 6.2$) and thus delineate a region of similarity. Similarly, for a given level of NPV (e.g., 0.80), we can identify an appropriate threshold distance between patients (e.g., $d > d_E = 9.7$) and thus delineate a region of dissimilarity.

region of minimal distribution overlap we observed in Fig. 6.3a. This suggests that a simple threshold on the Euclidean distance can be derived from those distributions.

We also find that PCP-derived similarity values have less control over identifying *dissimilar* patients than similar patients. This can be seen, in Fig. 6.7, by the relatively small range of the negative predictive values ($0.74 - 0.82$) across the domain of Euclidean distances. We attribute this finding to the higher proportion of dissimilar patients in the dataset; it is more likely that a pair of patients are dissimilar than they are similar. Nonetheless, the NPV curve exhibits the desired shape where as the Euclidean distance threshold increased, the likelihood of retrieving patients that are dissimilar to one another also increased. Building on this, we define a region of dissimilarity where patients identified as being dissimilar were highly likely of actually being so.

Having showed that patient cardiac prototypes can be reliably deployed for patient retrieval in a quantitative manner, we attempted to do so qualitatively. In Fig. 6.8a

(top), we present the distribution of the pairwise Euclidean distance values, d, between PCPs and representations in the held-out set of the Chapman dataset. After having converted these values to reflect patient-patient distances, we generated a pairwise distance matrix, a subset of which is presented in Fig. 6.8a (centre). Note that darker colours indicate patients that are more similar to one another. For illustration purposes, we identified the pair of patients which our framework determined to be most similar to one another (darkest cell) and illustrate their respective 12-lead electrocardiogram data.

We find that PCPs were able to sufficiently distinguish between unseen patients and thus act as reasonable patient-similarity tools. In the matrix of Fig. 6.8a, we see that there existed a large range of distance values for any chosen PCP (row). In other words, PCPs were closer to some representations than to others, implying that a chosen PCP was not trivially equidistant to all other representations. However, distinguishing between patients is not sufficient for a patient-similarity tool. We show that PCPs can also correctly retrieve relevant patients. In Fig. 6.8a (bottom), we show that the two patients identified as being most similar to one another, using our method, do indeed share many similarities. For example, their respective 12-lead ECG data are both associated with the cardiac arrhythmia label of sinus rhythm. Furthermore, similarities are observed when comparing the cardiac-specific statistics such as ventricular rate (84 in both cases) and atrial rate (84 in both cases). We hypothesize that this behaviour arises due to the ability of PCPs to efficiently summarize the cardiac state of a patient, a finding which reaffirms the potential of PCPs as tools for patient retrieval. It could be argued that validating the similarity of patients based exclusively on the cardiac arrhythmia reflected by their respective electrocardiograms is somewhat unconvincing. While we acknowledge this limitation, which is a function of the information currently present in publicly-available cardiac datasets, we believe this is an integral first step, and a sanity check, if you will, of the validity of the tools that we are using to quantify patient similarity. To address this limitation, we have repeated the aforementioned

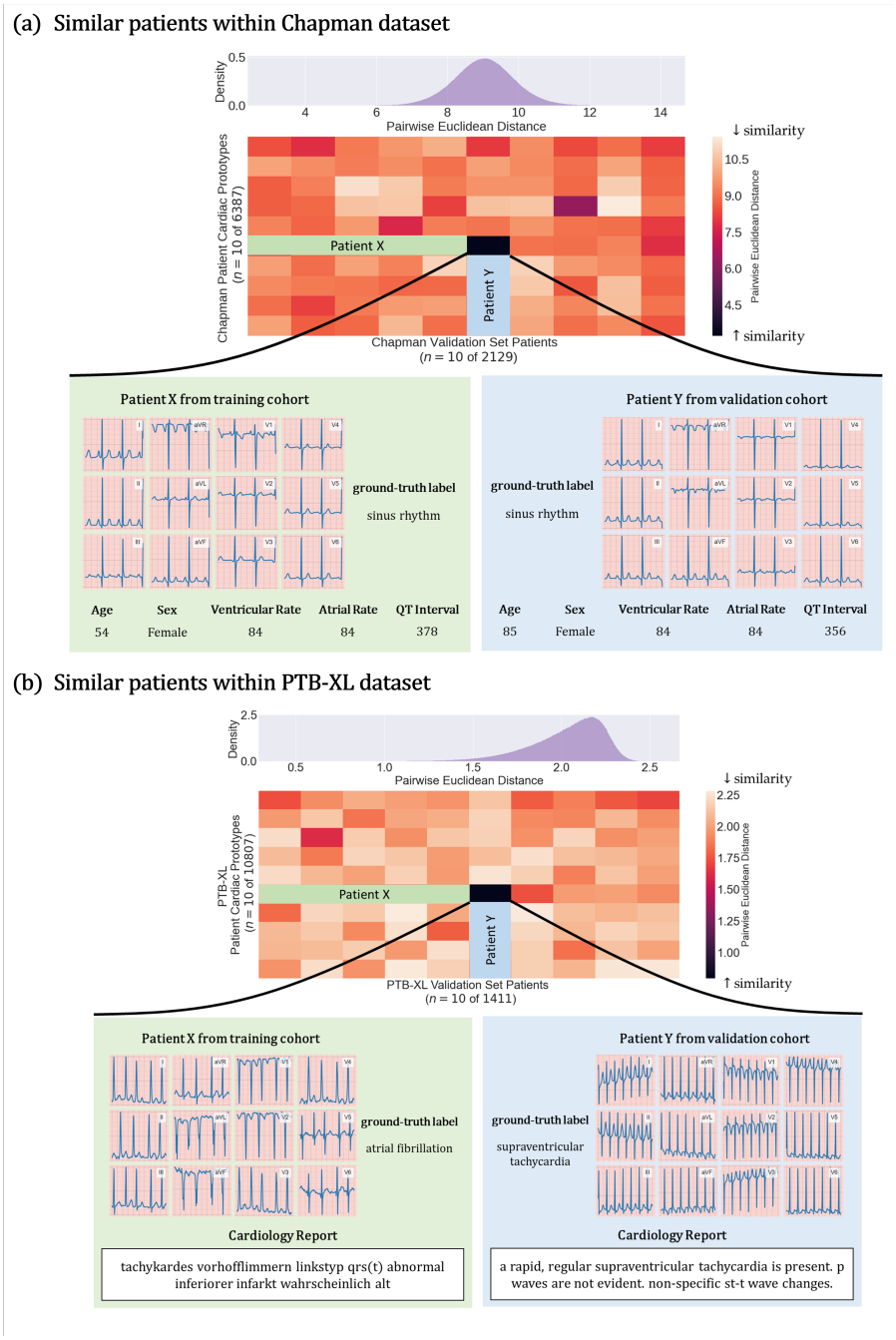


Figure 6.8: **Exploiting PCPs to discover similar patients within the (a) Chapman and (b) PTB-XL datasets.** We show a distribution of pairwise Euclidean distances between PCPs and representations in the validation set. We also present a subset of these pairwise distances in a matrix reflecting patient-patient distance values. We identify the most similar patient pair (↓ Euclidean distance) and retrieve their corresponding 12-lead electrocardiogram recordings, and where available, additional patient information or cardiology reports. We show that PCPs can reliably identify similar patients. This is evident by the high degree of similarity exhibited by the pair of patients. For example, in (a), both patients exhibit a cardiac arrhythmia label of sinus rhythm and reflect similar cardiac-specific statistics such as ventricular rate. In (b), the cardiac arrhythmia labels are similar and the cardiology reports reflect a similar pathology.

evaluations with the PTB-XL dataset, the only one which also contains cardiology reports associated with the electrocardiograms. We exploit PCPs to retrieve the most similar pair of patients, and, in Fig. 6.8b, present their corresponding 12-lead electrocardiogram signals *in addition to* their associated cardiology reports. Please note that such reports within the PTB-XL dataset are written in either German or English. To remain faithful to the original reports and to avoid potentially incorrect translations, we present these reports in their original language as found in the dataset. Similar to the conclusions we arrived at earlier with respect to the other datasets, we show that PCPs can reliably retrieve similar patients. In this case, the two patients are tagged with similar high-level cardiac arrhythmia labels, supraventricular tachycardia and atrial fibrillation, the latter being a specific example of the former. The cardiology reports also reflect similar information, with the report for patient X stating 'tachykardes verhofflimmern' (atrial fibrillation, tachycardia) and that for Patient Y stating 'rapid, regular supraventricular tachycardia'. These findings bode well for the use of PCPs as a patient similarity quantification tool. Our findings also extend to the setting in which similar patients are identified across *distinct* datasets. These results can be found in Supplementary Note 3.

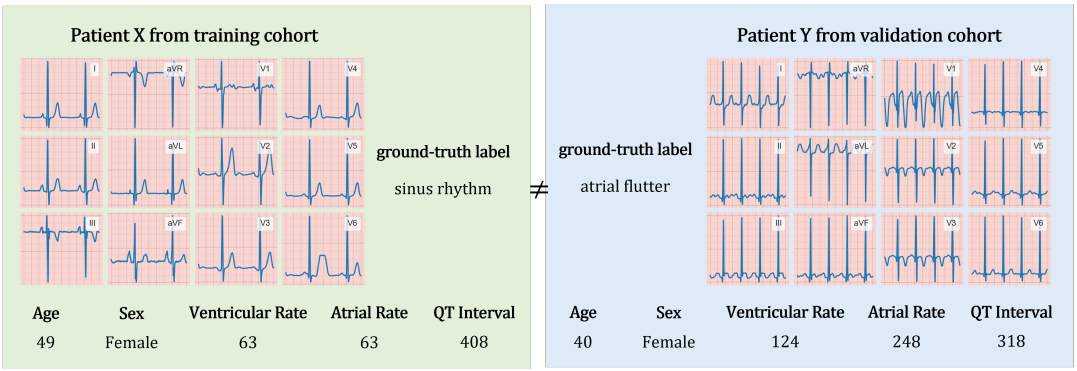


Figure 6.9: **12-Lead ECG segments corresponding to a pair of patients in the Chapman dataset identified as being dissimilar based on the PCPs.** Dissimilarity is defined as a high Euclidean distance between patient cardiac prototypes (PCPs) and representations of instances in the validation set. We show that PCPs can reliably retrieve dissimilar patients. This is evident by the distinct set of features exhibited by the pair of patients. For example, they exhibit a different cardiac arrhythmia label (sinus rhythm vs. atrial flutter) and distinct cardiac-specific statistics (e.g., ventricular rate = 63 vs. 124)

We also qualitatively evaluated our framework's ability to retrieve patients that are dissimilar from one another. To do so, we followed the previously-outlined process. However, instead of identifying the pair of patients with the lowest Euclidean distance, we identified that with the highest Euclidean distance (least similar). We present the 12-lead electrocardiogram data of these patients in the setting where patients are within the same dataset (Fig. 6.9). Results for identifying dissimilar patients *across* distinct datasets can be found in Supplementary Note 3.

We show that PCPs can also reliably retrieve *dissimilar* patients. This is evident by the observation that the retrieved patients exhibit dissimilar features. For example, the patients exhibit distinct ground-truth cardiac arrhythmia labels (sinus rhythm vs. atrial flutter). Whereas the former indicates a normal condition, the latter is relatively abnormal and may require intervention. Furthermore, the cardiac-specific statistics of these patients also differ. For example, the ventricular rates are 63 and 124, respectively, and the atrial rates are 63 and 248, respectively. Such a finding complements those in earlier sections and reaffirms the claim that PCPs can be a reliable tool for retrieving similar and dissimilar patients.

Part III

DOING MORE WITH MORE

In Part III, we continue to relax the characteristics of Part II which revolved around abundant, unlabelled data and limited, labelled data. Specifically, we now allow algorithms to indulge in the luxury of data. We illustrate this paradigm in Fig. 6.10.

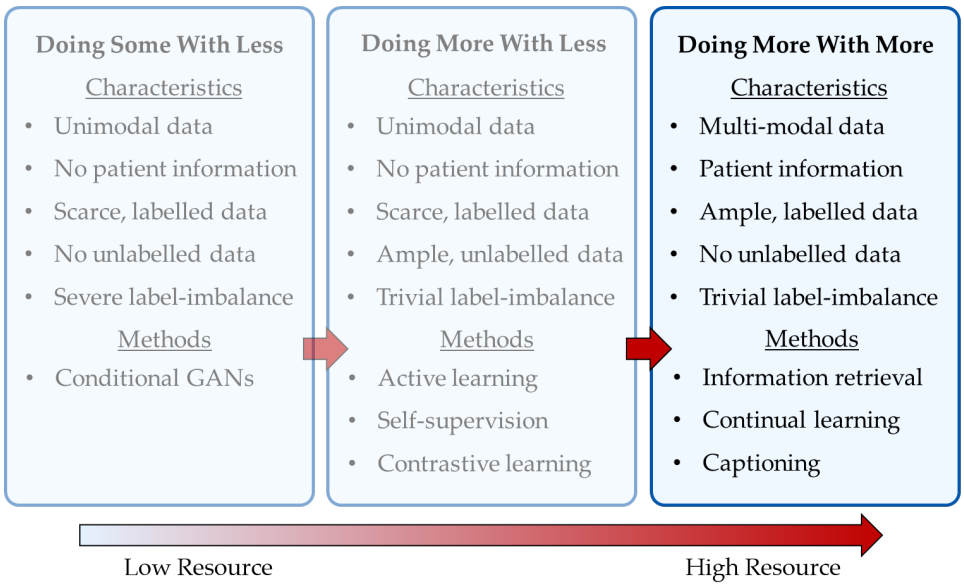


Figure 6.10: **Part III of the thesis focuses on the paradigm of 'doing more with more'.** This paradigm is characterized by abundant labelled data, patient information, and multi-modal data. In such an environment, we propose methods revolving around information retrieval, continual learning, and captioning.

Throughout Part III, we place less of an emphasis on methods that adapt to low-resource regimes, and instead address topics of clinical utility without the shackles of data-efficiency. To better identify a suitable set of topics, we assume that a) abundant, labelled data are available for exploitation. This could be due to, for example, the presence of an existing algorithm (such as those we designed earlier in the thesis) that automatically classifies instances. b) Data are available from multiple healthcare institutions. c) Data of multiple modalities are available (e.g., cardiac signals and clinical text) and include patient attribute information, such as sex and age. In light of these assumptions, we look to design clinical deep learning algorithms with the goal of 'doing more with more'; more data, more modalities, and more auxiliary information.

7

CLUSTERING AND RETRIEVAL OF CARDIAC SIGNALS

قمة الأدبِ أن تنصت إلى شخصٍ يُحدِثكَ في أمرٍ أنت تعرِفهُ جيداً وهو يجهله
إبن خلدون —

The epitome of etiquette is to listen to someone talk to you about a topic that you know well and that he is less familiar with
— Ibn Khaldoun

WITHIN the healthcare industry, evidence-based medicine has taken precedence as the gold-standard approach to care (Sackett et al., 1996). To abide by this, physicians and researchers alike have grown accustomed to manually searching for relevant instances in, and extracting information from, databases (Shivade et al., 2014). For example, clinicians extract a disease diagnosis from patient data, researchers involved in clinical trials search for and recruit patients satisfying specific inclusion criteria (Murthy et al., 2004), and educators retrieve relevant information as part of the continuing medical education scheme (Pourmand et al., 2015). This manual search-and-extract process, however, is hampered by the rapid growth of large-scale clinical databases and the increased prevalence of unlabelled instances; those for which patient attribute information is unavailable. As a result, we are interested in tackling the following research question.

Research Question

How can we design clinical algorithms that search for relevant instances in, and extract patient attribute information from, databases with unlabelled data?

In this chapter, we address the research question while exploiting large-scale electrocardiogram databases comprising patient attribute information. Our contributions are twofold. First, we propose a supervised contrastive learning framework, CROCS, in which we attract representations of cardiac signals associated with a unique set of patient attributes to embeddings, entitled clinical prototypes. Such attribute-specific prototypes, which create 'islands' of similar representations (Hinton, 2021), allow for *both* the clustering and retrieval of cardiac signals based on *multiple* patient attributes. Second, We show that CROCS outperforms the state-of-the-art method, DTC (Han et al., 2019), in the clustering setting and retrieves relevant cardiac signals from a large database. At the same time, clinical prototypes adopt a semantically meaningful arrangement and thus confer a high degree of interpretability.

7.1 RELATED WORK

Clinical representation learning and clustering. Learning meaningful representations of clinical data is an ongoing research endeavour. Recent research has focused on learning representations from electronic health records (EHRs) (Gee et al., 2019; Liu et al., 2019a; Li et al., 2020c; Biswal et al., 2020; Darabi et al., 2020) and via auto-encoders, which are then clustered using existing methods, such as k-means (Huang et al., 2019c; Landi et al., 2020). As for time-series data, auto-encoders are learned with (Ma et al., 2019) or without (Madiraju et al., 2018) an auxiliary clustering objective, salient features (shapelets) are identified in an unsupervised manner (Grabocka et al., 2014; Zhang et al., 2018), and patient-specific representations are learned via contrastive learning (Kiyasseh et al., 2020b). Li et al. (2020a) learn prototypes, or representative embeddings, via the ProtoNCE loss and cluster instances using k-means. Similar to our work is that of Gee et al. (2019) where prototypes are learned for the clustering of time-series signals. Their prototypes, however, cannot cluster instances based on multiple patient attributes and do not extend to the retrieval setting.

Clinical information retrieval (IR). Retrieving clinical data from a large database has been a longstanding goal of researchers within healthcare (Hersh and Greenes, 1990). Such research has involved the retrieval of clinical documents (Gurulingappa et al., 2016; Wang et al., 2017; Rhine, 2017; Wallace et al., 2016) where, for example, D'Avolio et al. (2010) map text queries to an ontology known as SNOMED, before retrieving relevant clinical documents. Recent research has focused on the retrieval of biomedical images (Saritha et al., 2019; Chittajallu et al., 2019), and EHR data (Goodwin and Harabagiu, 2018; Wang et al., 2019a) to discover patient cohorts in a clinical database (Chamberlin et al., 2019). Goodwin and Harabagiu (2016) implement an unsupervised patient cohort retrieval system by exploiting clinical text and time-series data. These approaches, however, do not explore cardiac signals, cannot account for multiple patient attributes, and are unable to also cluster instances. To the best of our knowledge, we are the first to design a learning framework that allows for *both* the clustering and retrieval of cardiac signals based on multiple patient attributes.

7.2 BACKGROUND

7.2.1 *Supervised Clustering*

We learn a function, $f_\theta : \boldsymbol{x} \in \mathbb{R}^D \rightarrow \boldsymbol{v} \in \mathbb{R}^E$, parameterized by $\boldsymbol{\theta}$, that maps a D-dimensional instance, $\boldsymbol{x}$, to an E-dimensional representation, $\boldsymbol{v}$. We also have a labelled dataset, $\mathcal{D} = \{\boldsymbol{x}_i, A_i\}_{i=1}^{N_l}$, where each instance, $\boldsymbol{x}_i$, is associated with a set of discrete patient attributes, $A_i = \{a_c^i, a_s^i, a_a^i\}$ where a_c = disease class, a_s = sex and a_a = age. Supervised clustering can involve learning $M \ll N_l$ centroids with each representing a unique set of attributes, $\{a_c^j, a_s^j, a_a^j\} \in \{A_j\}_{j=1}^{M}$, and grouping similar instances together. Given unlabelled instances, $\{\boldsymbol{x}_u\}_{u=1}^{N}$, the centroid closest to each representation, $\boldsymbol{v}_u = f_\theta(\boldsymbol{x}_u)$, is used to infer the latter's attributes. In this work, we learn cluster centroids which are more formally introduced in Sec. 7.3.1.

7.2.2 *Information Retrieval*

IR involves searching through a large, unlabelled dataset, $\{\boldsymbol{x}_u\}_{u=1}^{N}$, and retrieving a relevant instance, $\boldsymbol{x}_u$. However, relevance, defined based on whether an instance satisfies some criteria, is difficult to ascertain when instances are *unlabelled*. Typically, a query in the form of an embedding which represents a desired set of attributes, A_j, retrieves its closest (and most relevant) representation, $\boldsymbol{v}_u = f_\theta(\boldsymbol{x}_u)$, and infers the latter's attributes. In this work, we learn a set of query embeddings. As will become apparent in Sec. 7.4, these embeddings can also be treated as centroids, like those outlined in supervised clustering, and will thus serve a dual purpose.

7.3 METHODS

7.3.1 *Attribute-Specific Clinical Prototypes*

We aim to learn embeddings, referred to as clinical prototypes, that can be exploited for *both* the clustering and retrieval of cardiac signals based on multiple patient attributes. In the clustering setting, the goal is to annotate *unlabelled* instances with a set of patient attributes. To that end, we exploit clinical prototypes as centroids of clusters to which such unlabelled instances are assigned (see Fig. 7.1 left). In the retrieval setting, the goal is to retrieve *unlabelled* instances based on a set of patient attributes. To that end, we exploit each clinical prototype as a query to search through an *unlabelled* database and retrieve instances to which it is most similar (see Fig. 7.1 right).

In designing clinical prototypes, we take inspiration from the field of natural language processing (NLP) where a learnable word embedding represents a unique word. In our case, each prototype represents a unique combination of discrete patient attributes. Formally, given the attributes, α_c, α_s, and α_a, we would have $M = |\alpha_c| \times |\alpha_s| \times |\alpha_a|$ such unique combinations denoted by $A = \{\alpha_c^j, \alpha_s^j, \alpha_a^j\}_{j=1}^{M}$. We associate each combination, $A_j \in A$, with a learnable prototype, $\boldsymbol{p}_{A_j} \in \mathbb{R}^E$, for a set

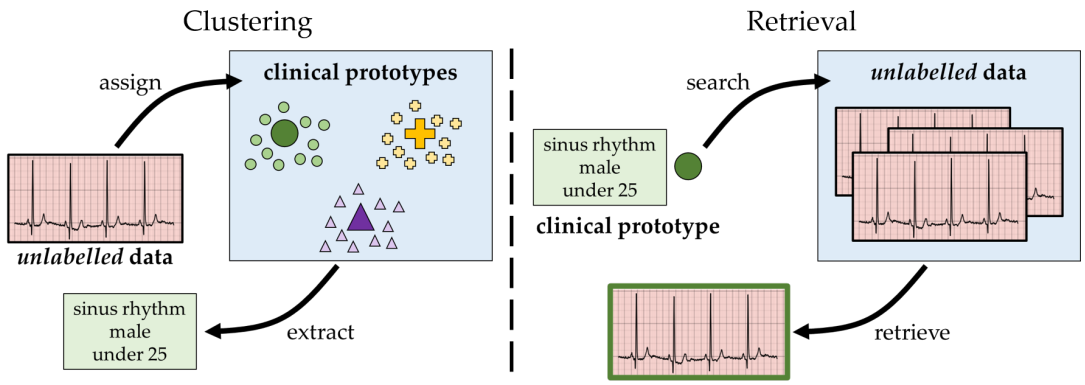


Figure 7.1: **Clinical prototypes are exploited for attribute-specific clustering and retrieval of cardiac signals.** For clustering, we exploit prototypes as centroids of clusters to which *unlabelled* instances are assigned. Such an assignment is associated with a set of attributes, such as disease class, sex, and age. For retrieval, we exploit each prototype as a query, associated with a set of attributes, to search through an *unlabelled* database and retrieve instances to which it is most similar.

of M prototypes, $P = \{\boldsymbol{p}_{A_j}\}_{j=1}^{M}$. Note that this framework extends to any number of attributes. In the next section, we outline how to learn these clinical prototypes.

7.3.2 *Learning Attribute-Specific Clinical Prototypes*

Clinical prototypes will serve a dual purpose of attribute-specific clustering and retrieval. As such, prototypes will need to be in proximity to a subset of representations of instances associated with a specific set of patient attributes. To achieve this proximity, we leverage the contrastive learning framework which involves a sequence of attractions and repulsions, as explained next.

Hard assignment. We encourage the representation, $\boldsymbol{v}_i = f_\theta(\boldsymbol{x}_i)$, of an instance, $\boldsymbol{x}_i$, associated with a set of attributes, $A_i \in A$, to be similar to the *single* clinical prototype, $\boldsymbol{p}_{A_j}$, that shares the exact same set of attributes (i.e., $A_i = A_j$), and dissimilar to the remaining clinical prototypes, $\{\boldsymbol{p}_{A_k}\}_{k \neq j}$. To achieve this, we optimize $\mathcal{L}_{NCE-hard}$ for a mini-batch of B instances (7.1). Intuitively, it heavily penalizes the learner if less probability mass is placed on the similarity of $\boldsymbol{v}_i$ and $\boldsymbol{p}_{A_j}$ than on the similarity of

other representation-and-prototype pairs. We choose to quantify the cosine similarity of pairs, $s(\boldsymbol{v}_i, \boldsymbol{p}_{A_j})$, alongside a temperature parameter, τ_s, as was done in Chapter 5.

$$\mathcal{L}_{NCE-hard} = -\frac{1}{B}\sum_{i=1}^{B}\log\left(\frac{e^{s(\boldsymbol{v}_i,\boldsymbol{p}_{A_j})}}{\sum_k^M e^{s(\boldsymbol{v}_i,\boldsymbol{p}_{A_k})}}\right) \qquad s(\boldsymbol{v}_i, \boldsymbol{p}_{A_j}) = \frac{\boldsymbol{v}_i.\boldsymbol{p}_{A_j}}{\|\boldsymbol{v}_i\|\|\boldsymbol{p}_{A_j}\|}.\frac{1}{\tau_s} \tag{7.1}$$

We refer to this many-to-one mapping from representations to clinical prototype as a hard assignment. Such an assignment, however, implies that a prototype is unlikely to extract potentially useful information from a representation whose attributes are not *exactly* the same as those of the prototype. We quantify this limitation in Sec. 7.5.4 and propose an alternative assignment next.

Soft assignment. To overcome the limitation of a hard assignment, we encourage the representation, $\boldsymbol{v}_i$, to be similar to a *subset* of clinical prototypes, $L \subset P$ (see Fig. 7.2). We must take caution, however, to avoid erroneously attracting representations to clinical prototypes from a *different* class. Doing so would reduce class-specific margins and thus hinder the downstream clustering and retrieval tasks.

Uniform attraction. We chose the subset to include prototypes, $L = \{\boldsymbol{p}_{A_l}\}_{l=1}^{|L|}$, that share the same disease class, α_c^i, with the representation, $\boldsymbol{v}_i$, implying that

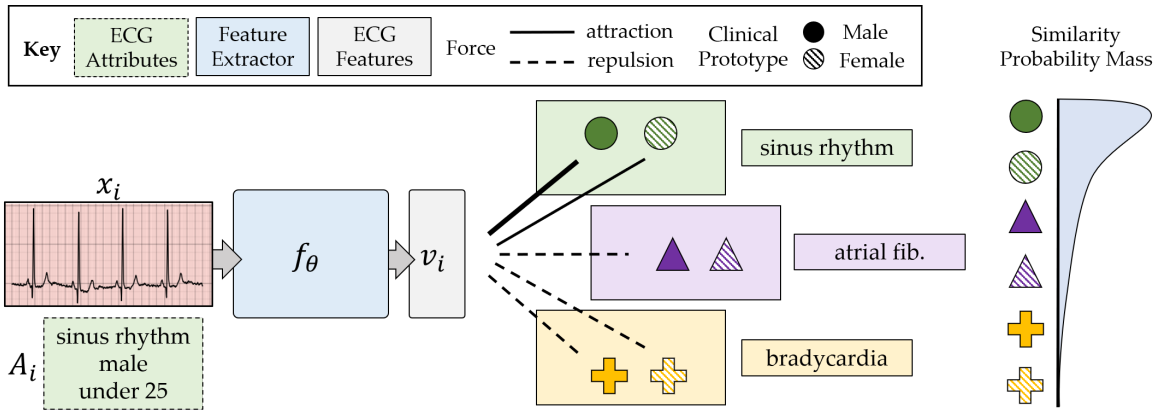


Figure 7.2: **Clinical prototypes are learned via a supervised contrastive learning framework referred to as CROCS.** The representation, $\boldsymbol{v}_i$, of an instance, $\boldsymbol{x}_i$, associated with a set of attributes, A_i, is strongly attracted to the clinical prototype which represents the same attributes, weakly attracted to others within the same disease class (colour), and repelled from those representing different classes. These attractions result in the shown similarity probability mass function.

$A_l \in \{\alpha_c^i, \alpha_s^l, \alpha_a^l\}_{l=1}^{|L|} \subset A$. Note that the clinical prototypes in the subset, L, continue to represent attributes that vary along the dimensions of patient sex and age (α_s, α_a). Therefore, attracting $\boldsymbol{v}_i$ to prototypes in L *uniformly* will likely cause the latter to become minimally distinguishable across sex and age. This is an undesired outcome in light of our goal of learning *attribute-specific* prototypes. We avoid this behaviour by modulating the degree of attraction between $\boldsymbol{v}_i$ and all prototypes in the set, P, as outlined next.

Modulated attraction. The attractive force between $\boldsymbol{v}_i$ and $\boldsymbol{p}_{A_j}$ is reflected by the corresponding $\mathcal{L}_{NCE-hard}$ term (7.1). By placing less probability mass on $s(\boldsymbol{v}_i, \boldsymbol{p}_{A_j})$ (i.e., less similarity) than on $s(\boldsymbol{v}_i, \boldsymbol{p}_{A_k})$ $\forall k, k \neq j$, the learner incurs a higher loss and thus attracts the pair. We extend this logic to all prototypes to obtain M $\mathcal{L}_{NCE-hard}$ terms per representation. To modulate these attractions, we introduce a weight, $w_{ij} \in \{w_{ij}\}_{j=1}^{M}$, as a coefficient of the j-th loss term (7.2). Each weight, w_{ij}, quantifies the degree of matching between attributes of the representation, $A_i = \{\alpha_c^i, \alpha_s^i, \alpha_a^i\}$, and those of the clinical prototype, $A_j = \{\alpha_c^j, \alpha_s^j, \alpha_a^j\}$, as reflected by $q(A_i, A_j) \in \mathbb{R}$. We define $\mathbb{1}$ as the indicator function and τ_ω as a temperature parameter that determines how soft the representation-and-prototype attraction is. For example, as $\tau_\omega \to \infty$, this approach reverts to the uniform attraction setup. The intuition is that a stronger attraction ($\uparrow w_{ij}$) should exist for a representation-and-prototype pair that shares more attributes. We also avoid the erroneous attraction of pairs from different classes (i.e., $\alpha_c^i \neq \alpha_c^j$) by setting $\omega_{ij} = 0$. When visualizing the UMAP projection (McInnes et al., 2018) of prototypes learned with a uniform attraction ($\tau_\omega = \infty$) (Fig. 7.3 left) and those learned with a modulated attraction ($\tau_\omega \neq \infty$) (Fig. 7.3 centre), we show that the latter become more linearly separable across sex.

$$\mathcal{L}_{NCE-soft} = -\frac{1}{B}\sum_{i=1}^{B}\left[\sum_{j=1}^{M}\omega_{ij}\log\left(\frac{e^{s(\boldsymbol{v}_i,\boldsymbol{p}_{A_j})}}{\sum_k^M e^{s(\boldsymbol{v}_i,\boldsymbol{p}_{A_k})}}\right)\right] \tag{7.2}$$

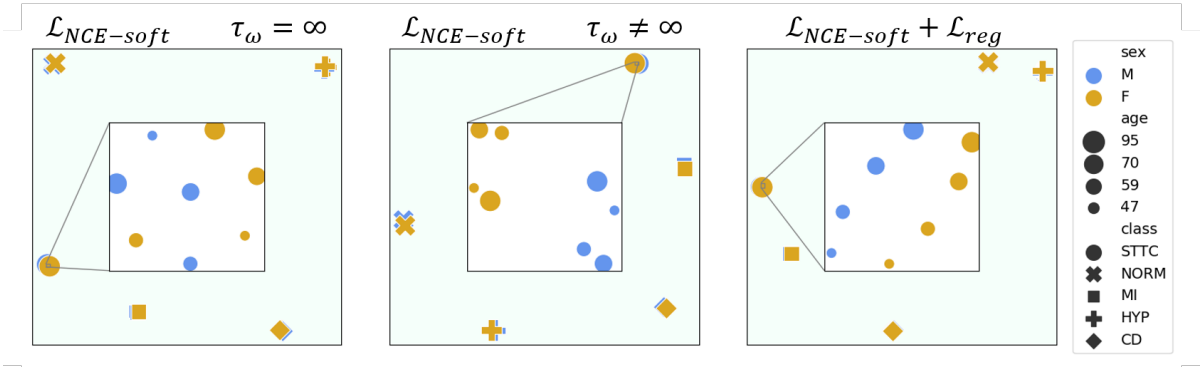


Figure 7.3: **Projection of set of clinical prototypes, P, learned with variants of the CROCS framework.** With $\mathcal{L}_{NCE-soft}$ $\tau_\omega = \infty$, representations are attracted to class-specific prototypes *uniformly*. With $\mathcal{L}_{NCE-soft}$ $\tau_\omega \neq \infty$, prototypes become more linearly separable across sex. Our full framework, $\mathcal{L}_{NCE-soft} + \mathcal{L}_{reg}$, leads to prototypes that adopt a semantically meaningful arrangement. We investigate the marginal impact of these design choices on performance in Sec. 7.5.4.

$$\omega_{ij} = \begin{cases} \frac{e^{q(A_i,A_j)}}{\sum_l^{|L|} e^{q(A_i,A_j)}} & \text{if } \alpha_c^i = \alpha_c^j \\ 0 & \text{otherwise} \end{cases}$$

$$q(A_i, A_j) = \frac{1}{\tau_\omega}.[\mathbb{1}(\alpha_c^i = \alpha_c^j) + \mathbb{1}(\alpha_s^i = \alpha_s^j) + \mathbb{1}(\alpha_a^i = \alpha_a^j)]$$

Arrangement of clinical prototypes. Clinical prototypes would confer a high degree of interpretability if they also captured the semantic relationships between attributes. Concretely, prototypes representing similar attribute sets (e.g., adjacent age groups) should be similar to one another. This is analogous to the high similarity of word embeddings representing semantically similar words (Smeaton, 1999). To capture these semantic relationships, we encourage class-specific prototypes to maintain some desired distance between one another. As such, each pair of M clinical prototypes, $\boldsymbol{p}_{A_j}$, $\boldsymbol{p}_{A_k}$ $\forall j, k \in [1, M]$ is associated with an empirical and ground-truth (desired) distance. For the former, we normalize the prototypes (L_2 norm) and calculate their Euclidean distance, $\hat{d}_{jk} = \|\boldsymbol{p}_{A_j} - \boldsymbol{p}_{A_k}\|_2$ $\forall j, k \in [1, M]$. For the latter, we define the ground-truth distance as $d_{jk} = \beta \times d_H \in \mathbb{R}$, where $d_H(A_j, A_k) \in \mathbb{Z}^+$ is the Hamming distance between a pair of discrete attribute sets. Intuitively, the Hamming distance counts the number of attribute mismatches and $\beta \in \mathbb{R}$ penalizes each mis-

match. Therefore, we can generate an *empirical* set, $\{\hat{d}_{jk}\}_{j,k=1}^{M}$ and a *ground-truth* set, $\{d_{jk}\}_{j,k=1}^{M}$, of distance values. By minimizing the mean-squared error between these two sets, we learn clinical prototypes that adopt a semantically meaningful arrangement (see Fig. 7.3 right). Since we are only interested in adopting this arrangement for prototypes of the same class (i.e., $\alpha_c^j = \alpha_c^k$), we incorporate the regularization term, $\mathcal{L}_{reg}$, into the final objective function, $\mathcal{L}_{tot}$.

$$\mathcal{L}_{reg} = \sum_{j,k=1}^{M} (\hat{d}_{jk} - d_{jk})^2 \Leftrightarrow \alpha_c^j = \alpha_c^k \qquad \mathcal{L}_{tot} = \mathcal{L}_{NCE-soft} + \mathcal{L}_{reg} \tag{7.3}$$

7.4 EXPERIMENTAL DESIGN

7.4.1 *Data and Pre-processing*

To evaluate our method, we leverage two different datasets, each of which consists of cardiac time-series waveforms alongside their annotations including patient disease class (cardiac arrhythmia label), sex, and age. We split each of the aforementioned waveforms into non-overlapping segments comprising 2500 samples. In Table 7.1, we present a summary of these datasets.

Table 7.1: **Summary of the datasets used for evaluation.** We also show additional pre-processing information. Please click on the dataset's name for more information.

Dataset	Abbreviation	Modality	Normalization	Exclusion Criteria
Chapman	$\mathcal{D}_1$	ECG	✗	-
PTB-XL	$\mathcal{D}_2$	ECG	✓	-

7.4.2 *Description of Clustering Setting*

During inference, we treat the clinical prototypes, $\{\boldsymbol{p}_{A_j}\}_{j=1}^{M}$, as a set of cluster centroids. We calculate the Euclidean distance between the i-th representation and each of the M prototypes, identify the closest prototype, $\boldsymbol{p}_{A_k}$, and assign the representation to A_k which we now denote by $\hat{A}_i = \{\hat{\alpha}_c^i, \hat{\alpha}_s^i, \hat{\alpha}_a^i\}$. Repeating this process for N unseen

instances results in a set of assigned attribute values, $\vec{\hat{\alpha}} = \{\hat{\alpha}^i\}_{i=1}^N$, for a particular attribute, $\hat{\alpha} \in A$ (e.g., disease class). Such unseen instances would typically be *unlabelled*. For evaluation, however, we assume access to the ground-truth attribute values, $\vec{\alpha} = \{\alpha^i\}_{i=1}^N$, with which we calculate the accuracy, $\text{Acc}(\hat{\alpha})$, and the adjusted mutual information, $\text{AMI}(\hat{\alpha}) \in [0, 1]$, between $\vec{\hat{\alpha}}$ and $\vec{\alpha}$.

$$\text{Acc}(\hat{\alpha}) = \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}(\hat{\alpha}^i = \alpha^i) \qquad \text{AMI}(\hat{\alpha}) = \frac{\left[\mathbb{MI}(\vec{\alpha}, \vec{\hat{\alpha}}) - \mathbb{E}(\mathbb{MI}(\vec{\alpha}, \vec{\hat{\alpha}}))\right]}{\mathbb{E}(\mathbb{H}(\vec{\alpha}), \mathbb{H}(\vec{\hat{\alpha}})) - \mathbb{E}(\mathbb{MI}(\vec{\alpha}, \vec{\hat{\alpha}}))} \tag{7.4}$$

where $\mathbb{MI}(\vec{\alpha}, \vec{\hat{\alpha}})$ denotes the mutual information between the ground-truth and assigned set of attribute values, and $\mathbb{H}(\vec{\alpha})$ denotes the entropy of the attribute values.

7.4.3 *Description of Retrieval Setting*

During inference, we treat the clinical prototypes, $\{\boldsymbol{p}_{A_j}\}_{j=1}^M$, as a query set. We calculate the Euclidean distance between the j-th clinical prototype and representations of N unseen instances, retrieve the K closest instances, and then assign them to $A_j = \{\alpha_c^j, \alpha_s^j, \alpha_a^j\}$. Note that such instances would typically be *unlabelled*, thus precluding a simple SQL search. For evaluation, however, we assume access to the ground-truth attributes, $\{\alpha_c^i, \alpha_s^i, \alpha_a^i\}_{i=1}^K$, with which we calculate a variant of the precision at K metric (7.5). It checks whether at least one of the retrieved instances is relevant, where relevance is based on a partial or exact match of query and instance attributes (# attribute matches). This value is then averaged across all M prototypes.

$$\text{P@}K = \frac{1}{M}\sum_{j=1}^{M} \mathbb{1}\left(\sum_{i=1}^{K} \mathbb{1}\underbrace{\left(\left[\alpha_c^i = \alpha_c^j\right] \cap \left[\alpha_s^i = \alpha_s^j\right] \cap \left[\alpha_a^i = \alpha_a^j\right]\right)}_{\text{relevance} \equiv \text{\# attribute matches} = 3} \geq 1\right) \tag{7.5}$$

7.4.4 *Baseline Methods*

We compare clinical prototypes learned via the CROCS framework (**CP CROCS**) to the following methods. For retrieval, **Deep Transfer Cluster (DTC)** (Han et al.,

2019) learns cluster prototypes by minimizing the KL divergence between a target distribution and one based on the distance between prototypes and representations. **TP CROCS** involves traditional prototypes where each prototype, $\bar{\boldsymbol{v}}_{A_j} = \frac{1}{\sum \mathbb{I}(A_i = A_j)} \sum_{i=1}^{N} \boldsymbol{v}_i \cdot \mathbb{I}(A_i = A_j)$, is simply an average of representations, $\boldsymbol{v}_i$, associated with the same set of attributes, A_j. Such representations are also learned via CROCS.

For the clustering task, we compare to several state-of-the-art clustering methods, in addition to those mentioned above. k-means identifies cluster centroids based on the input instances, $\boldsymbol{x}$ (**KM raw**), or representations, $\boldsymbol{v}$, learned via the CROCS (**KM CROCS**) or explainable prototypes (**KM EP**) (Gee et al., 2019) framework. **DeepCluster (DC)** (Caron et al., 2018) iteratively applies k-means to representations, pseudo-labels them according to their assigned cluster, and then exploits such labels for supervised training. **Deep Temporal Clustering Representation (DTCR)** (Ma et al., 2019) optimizes an objective function with a reconstruction, k-means, and classifier loss that determines whether instances are real. **Information Invariant Clustering (IIC)** (Ji et al., 2019) maximizes the mutual information between class probabilities of an instance and its perturbed counterpart. **SeLA** (Asano et al., 2020) implements Sinkhorn-Knopp to pseudo-label instances in a supervised manner. Further details can be found in Appendix C.1.1.

7.5 RESULTS

In this section, and with an eye on addressing the initial research question, we attempt to answer the following questions: (i) Are clinical prototypes attribute-specific and do they capture the semantic relationships between patient attributes? (ii) Are clinical prototypes able to accurately cluster cardiac signals according to multiple patient attributes? (iii) Are clinical prototypes able to accurately retrieve cardiac signals according to patient attributes? (iv) What is the marginal impact of the design choices of the CROCS framework?

7.5.1 *Visualizing Clinical Prototypes*

We begin by qualitatively validating the claim that clinical prototypes are attribute-specific. In other words, can prototypes be delineated along the dimensions of disease class, sex, and age? To address this, we illustrate, in Fig. 7.4, two dimensional UMAP projections of the class-specific clinical prototypes (large, coloured shapes), traditional prototypes (large, black shapes), and representations of instances in the validation set of Chapman and PTB-XL.

We show that clinical prototypes are indeed disease class-specific, as evident by the high degree of class separability of such prototypes. We also find that clinical prototypes are distinct from traditional prototypes, a distinction whose importance will become evident in later sections. Second, the consistency of the class labels of the prototypes with those of the representations is a harbinger of how prototypes may perform in the clustering and retrieval settings, as we show in the next section. These findings complement the delineation of the prototypes along the dimensions of sex and age, and their adoption of a semantically meaningful arrangement, as was shown in Section 7.3.

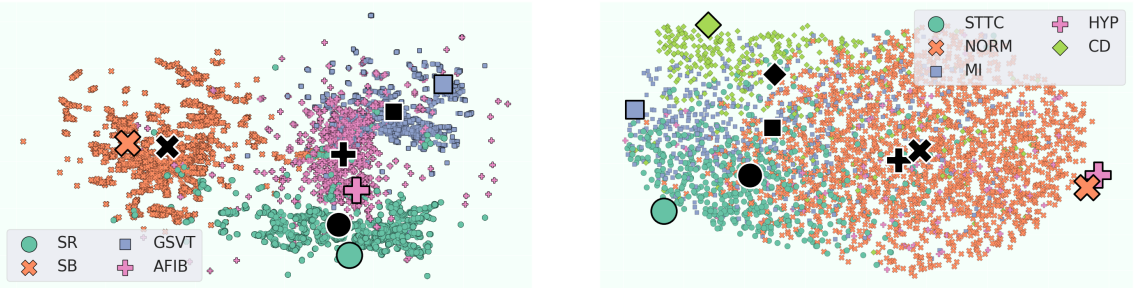


Figure 7.4: **Projection of class-specific clinical prototypes p (large, coloured shapes), traditional prototypes $\bar{v}$, (large, black shapes), and representations, v, in the validation set of (left) Chapman and (right) PTB-XL.** We show that clinical prototypes are class-specific, consistent with the class labels of representations, and distinct from traditional prototypes. This bodes well for their use as centroids for clustering (see Sec. 7.5.2) and as queries for retrieval (see Sec. 7.5.3).

7.5.2 *Deploying Clinical Prototypes in Clustering Setting*

In the clustering setting, we assign cardiac signals in a held-out dataset to a set of patient attributes associated with the cluster of the closest clinical prototype. We evaluate these assignments based on the three patient-specific attributes (disease class, sex, and age) and present the results in Table 7.2.

Table 7.2: **Clustering performance on the validation set of Chapman and PTB-XL.** Evaluation is based on (a) class and (b) sex and age attributes. Results are averaged across five random seeds. Brackets indicate standard deviation and bold reflects the top-performing method. We show that CP CROCS outperforms the remaining methods regardless of patient attribute.

(a) Cardiac arrhythmia class attribute

Method	Chapman		PTB-XL	
	Acc	AMI	Acc	AMI
SeLA (Asano et al., 2020)	21.0 (0.1)	9.2 (10.0)	10.5 (0.1)	1.6 (0.5)
DC (Caron et al., 2018)	21.0 (0.1)	3.0 (4.0)	10.5 (0.1)	4.9 (0.0)
IIC (Ji et al., 2019)	27.0 (0.2)	0.2 (0.0)	22.0 (0.7)	0.5 (0.0)
DTCR (Ma et al., 2019)	29.3 (1.3)	0.2 (0.2)	38.4 (4.2)	1.4 (0.3)
DTC (Han et al., 2019)	53.4 (15.0)	23.4 (20.0)	67.3 (1.0)	25.2 (0.7)
KM raw	28.4 (1.2)	0.3 (0.0)	-	-
KM EP (Gee et al., 2019)	65.6 (4.0)	42.8 (2.6)	48.9 (1.1)	24.2 (1.9)
KM CROCS	73.4 (7.1)	58.6 (2.8)	47.6 (3.9)	25.9 (1.1)
TP CROCS	80.3 (1.4)	65.0 (0.6)	53.6 (0.7)	29.1 (0.2)
CP CROCS	**90.3** (0.8)	**72.8** (1.6)	**76.0** (0.3)	**35.9** (0.4)

(b) Sex and age attributes

Method	Chapman		PTB-XL	
	sex	age	sex	age
DTCR (Ma et al., 2019)	51.2 (0.9)	25.9 (0.1)	51.0 (0.9)	25.1 (0.8)
DTC (Han et al., 2019)	54.8 (0.5)	26.4 (0.8)	58.6 (1.9)	25.7 (0.7)
KM EP (Gee et al., 2019)	56.1 (0.0)	31.0 (0.4)	54.1 (0.3)	29.2 (2.0)
KM CROCS	54.9 (0.8)	32.3 (0.5)	51.8 (1.2)	31.6 (1.5)
TP CROCS	54.8 (1.0)	31.1 (1.7)	69.7 (0.8)	**39.4** (0.4)
CP CROCS	**57.4** (1.2)	**38.0** (0.8)	**73.5** (0.6)	19.5 (0.2)

In Table 7.2, we find that CROCS outperforms both generic and domain-specific state-of-the-art clustering methods. For example, on Chapman, CP CROCS, KM EP, and DTC achieve Acc(class) $=$ 90.3, 65.6, and 53.4% respectively. Along the dimension of sex, and on PTB-XL, CP CROCS and DTC achieve Acc(sex) $=$ 73.5 and 58.6%, respectively. Second, we find that CROCS leads to rich representation learning that facilitates clustering. This is evident when comparing the performance of k-means applied to representations that are learned via different methods. For ex-

ample, on Chapman, KM raw, KM EP, and KM CROCS achieve Acc(class) = 28.4, 65.6, and 73.4%, respectively. We also find that clinical prototypes, when exploited as centroids, are preferable to traditional prototypes, and centroids learned via k-means. For example, on PTB-XL, KM CROCS, TP CROCS, and CP CROCS achieve Acc(class) = 47.6, 53.6, and 76.0%, respectively.

We also find that, on average, CP CROCS outperforms TP CROCS across datasets (Chapman and PTB-XL) and attributes (disease class, sex, and age) in all but one case (PTB-XL, age). Here, it appears that there exists a trade-off between performance of the age and sex attributes. Performing better on the sex attribute comes at the expense of performance on the age attribute. This could be due to the degree of attraction that is enforced between representations and clinical prototypes that reflect a similar set of patient attributes. Overall, though, discriminating sex and age exclusively based on the ECG remains a difficult task (particularly with small datasets). For context, previous researchers have attempted to do so with some success however with access to data from around 500K patients (Attia et al., 2019a).

These findings, which hold across datasets and evaluation metrics, point to the overall utility of the CROCS framework and clinical prototypes for attribute-specific clustering.

7.5.3 *Deploying Clinical Prototypes in the Retrieval Setting*

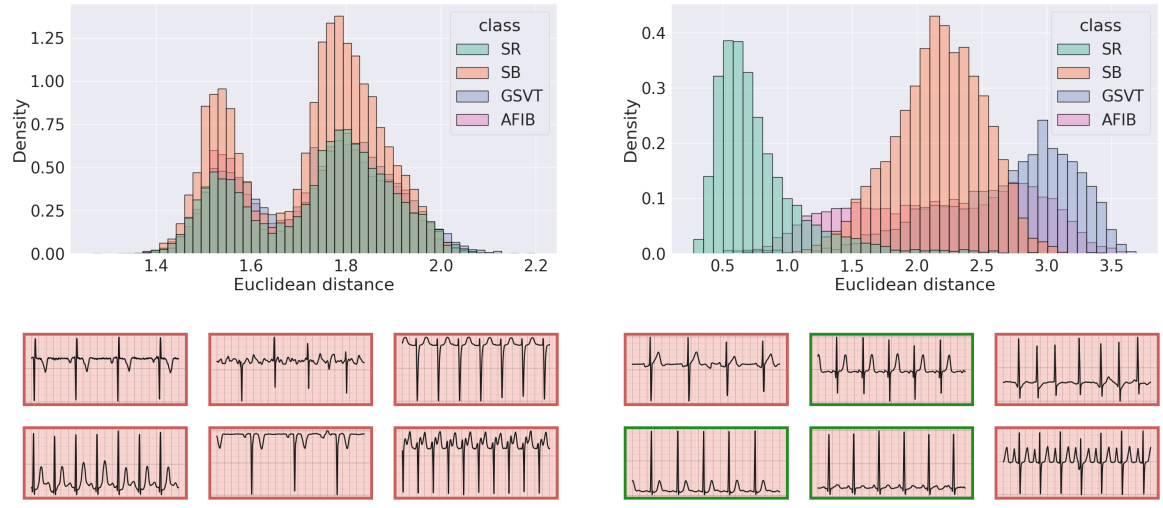
Up until now, we have shown that CROCS leads to accurate clustering. In this section, we show that CROCS can also be independently exploited for retrieval. Specifically, a query retrieves the closest $K = [1, 5, 10]$ previously unseen cardiac signals, and assigns them to its associated set of patient attributes. In Table 7.3, we evaluate these assignments based on both partial and exact matches of the attributes (# attribute matches) represented by the query and retrieved cardiac signals.

In Table 7.3, we find that CROCS outperforms the baseline retrieval method, DTC. For example, on Chapman, at $K = 1$, and when # attribute matches ≥ 1, CP CROCS

Table 7.3: **Precision of K retrieved representations, v, in the validation set of Chapman and PTB-XL, that are closest to the query.** Results are shown for partial and exact matches of the attributes (# attribute matches) represented by the query and retrieved cardiac signals, and are averaged across five random seeds. Brackets indicate standard deviation and bold reflects the top-performing method. The strong performance of CP CROCS provides evidence in support of our CROCS framework.

# attribute matches	Query	Chapman			PTB-XL		
		$K=1$	5	10	1	5	10
≥ 1	DTC (Han et al., 2019)	71.9 (0.0)	100.0 (0.0)	100.0 (0.0)	70.0 (0.0)	90.0 (8.4)	100.0 (0.0)
	TP CROCS	91.9 (3.2)	97.5 (2.3)	100.0 (0.0)	**99.0** (2.0)	100.0 (0.0)	100.0 (0.0)
	CP CROCS	**95.6** (6.1)	100.0 (0.0)	100.0 (0.0)	92.5 (0.0)	100.0 (0.0)	100.0 (0.0)
≥ 2	DTC (Han et al., 2019)	25.0 (0.0)	71.9 (9.7)	90.0 (7.0)	22.5 (0.0)	52.5 (16.0)	80.5 (4.0)
	TP CROCS	55.0 (3.8)	79.4 (7.6)	90.0 (0.1)	**71.5** (2.0)	94.5 (1.0)	**100.0** (0.0)
	CP CROCS	**61.3** (10.0)	**86.3** (10.9)	**93.8** (7.9)	63.0 (1.9)	**96.5** (2.0)	99.5 (1.0)
$= 3$	DTC (Han et al., 2019)	3.1 (0.0)	15.0 (3.1)	23.8 (0.1)	2.5 (0.0)	9.5 (4.3)	16.5 (0.2)
	TP CROCS	10.6 (1.5)	23.8 (6.1)	36.9 (7.0)	**15.5** (1.0)	32.0 (1.0)	43.5 (0.1)
	CP CROCS	**11.3** (2.5)	**33.1** (6.1)	**46.3** (6.7)	12.5 (0.0)	**33.5** (2.0)	**43.0** (4.0)

and DTC achieve a precision of 95.6 and 71.9%, respectively. This indicates that, on average, 95.6% of the cardiac signals retrieved by the clinical prototypes are



(a) DTC query {SR, male, under 49}

(b) CP CROCS query {SR, male, under 49}

Figure 7.5: **Qualitative retrieval performance of two distinct query prototypes.** (**top row**) Euclidean distance from (a) DTC query or (b) CP CROCS query to representations, v, in the validation set of Chapman. (**bottom row**) Six closest cardiac signals to the query which is associated with a set of patient attributes {disease, sex, age}. Retrieved cardiac signals with a green border indicate those whose class attribute matches that of the query. We show that the CP CROCS query is closer to representations of the same class (SR) and thus retrieves relevant cardiac signals.

relevant. Relevance, in this case, implies that the retrieved cardiac signals share at least one attribute with the query. Such a finding points to the utility of clinical prototypes as queries in the retrieval setting. We also find that CROCS leads to rich representation learning that facilitates retrieval. This is evident by the strong performance of TP CROCS which depends directly on representations learned via our CROCS framework. For example, on PTB-XL, at $K = 1$, and when # attribute matches ≥ 1, DTC, TP CROCS, and CP CROCS achieve a precision of 70.0, 99.0, and 92.5%, respectively. In this particular case, the lower performance of CP CROCS relative to TP CROCS is hypothesized to stem from clinical prototypes acting instead as *archetypes* (extreme representative data points) (Mørup and Hansen, 2012) which may occasionally hinder retrieval along multiple attributes. Evidence of such extreme embeddings can be found in Fig. 7.4.

To qualitatively evaluate the retrieval performance, we first randomly choose a query representing a set of attributes and calculate its Euclidean distance to the representations in a validation set. We present distributions of such distance values in Fig. 7.5 (top row), for a DTC query and a CP CROCS query, coloured based on the ground-truth class of the representations (other queries shown in Appendix C.1.2). In Fig. 7.5 (bottom row), we illustrate the six cardiac signals ($K = 6$) that are closest to each query, with a green border indicating signals whose class attribute matches that of the query. We find that the CP CROCS query is closer to representations of the same class (SR) than to those of a different class. For example, in Fig. 7.5b (top row), the average distance between the CP CROCS query representing $\{\text{SR}, \text{male}, \text{under 49}\}$ and representations with and without the class attribute SR is ≈ 0.6 and > 1.5, respectively. Such separability, which is not exhibited by the DTC query, points to the improved reliability of the CP CROCS query in distinguishing between the relevance of cardiac signals. Further evidence in support of this reliability is shown in Fig. 7.5 (bottom row) where we find that a DTC and a CP CROCS query retrieve relevant

cardiac signals 0% and 50% of the time, respectively. This finding also extends to the PTB-XL dataset (Appendix C.1.2).

7.5.4 *Investigating the Marginal Impact of Design Choices*

We have shown that CROCS reliably allows for both clustering and retrieval. In this section, we conduct several ablation studies to better understand the root cause of this reliability (see Table 7.4). We find that, on average, the soft assignment of representations to prototypes is preferable to the hard assignment. For example, on PTB-XL, $\mathcal{L}_{NCE-soft}$ and $\mathcal{L}_{NCE-hard}$ achieve Acc(class) $\approx$ 76.0 and 66.5%, respectively. We also find that our full framework ($\mathcal{L}_{NCE-soft} + \mathcal{L}_{reg}$) performs better than, or on par with, other variants. For example, on Chapman, $\mathcal{L}_{NCE-hard}$ and $\mathcal{L}_{NCE-soft}$ $\tau_\omega = \infty$, and $\tau_\omega \neq \infty$ achieve AMI(class) $=$ 67.5, 68.2, and 72.1%, respectively, whereas $\mathcal{L}_{NCE-soft} + \mathcal{L}_{reg}$ achieves AMI(class) $=$ 72.8%. This is a positive outcome given that the regularization term's main purpose was simply to improve the interpretability of prototypes by allowing them to capture the semantic relationships between attributes. These findings extend to the retrieval setting (Appendix C.1.2).

Table 7.4: **Marginal impact of design choices of CROCS on clustering performance.** Evaluation is based on (a) class and (b) sex and age attributes. Results are averaged across five random seeds. Brackets indicate standard deviation and bold reflects the top-performing method. We show that our full framework ($\mathcal{L}_{NCE-soft} + \mathcal{L}_{reg}$) is preferable to other variants regardless of attribute.

(a) Cardiac arrhythmia class attribute

Method	Chapman		PTB-XL	
	Acc	AMI	Acc	AMI
$\mathcal{L}_{NCE-hard}$	86.8 (0.7)	67.5 (1.1)	66.5 (0.1)	35.0 (0.0)
$\mathcal{L}_{NCE-soft}$				
$\tau_\omega = \infty$	87.3 (0.5)	68.2 (0.6)	76.3 (0.5)	36.1 (1.0)
$\tau_\omega \neq \infty$	89.8 (1.7)	72.1 (2.8)	76.1 (0.2)	36.0 (0.4)
$+\mathcal{L}_{reg}$	**90.3** (0.8)	**72.8** (1.6)	76.0 (0.3)	35.9 (0.4)

(b) Sex and age attributes

Method	Chapman		PTB-XL	
	sex	age	sex	age
$\mathcal{L}_{NCE-hard}$	56.9 (0.2)	26.2 (0.0)	76.3 (0.7)	19.8 (0.0)
$\mathcal{L}_{NCE-soft}$				
$\tau_\omega = \infty$	55.2 (0.5)	34.7 (0.3)	50.4 (0.1)	20.8 (1.0)
$\tau_\omega \neq \infty$	56.8 (1.8)	37.4 (1.0)	74.3 (0.0)	19.2 (0.9)
$+\mathcal{L}_{reg}$	**57.4** (1.2)	**38.0** (0.8)	73.5 (0.6)	19.5 (0.2)

8

CONTINUAL LEARNING FOR CARDIAC ARRHYTHMIA DIAGNOSIS

Tell me and I forget, teach me and I may remember, involve me and I learn.

— Benjamin Franklin

CLINICAL data are commonly assumed to be independent and identically distributed (i.i.d.). However, this assumption, which we made implicitly throughout this thesis, does not often hold. This violation of the i.i.d. assumption is known to hamper the ability of algorithms to learn and perform well on a held-out evaluation set (Krueger et al., 2020). Consider a scenario in which data are extracted from a single healthcare institution during the summer and winter months. An algorithm that learns from data extracted during the *summer* months is expected to make reliable predictions when deployed on data from the same season. However, deploying it on data from the *winter* months could lead to erroneous predictions and thus hinder its clinical utility. Such discrepancy in performance can be partially explained by the different distributions of the data in the summer and winter months. This can be driven by seasonal demographic and disease profiles, temporal changes that are particularly prevalent within healthcare.

The dynamic and chaotic environment that characterizes healthcare necessitates the availability of algorithms that are dynamically reliable; those that can adapt to potential covariate shift without catastrophically forgetting how to perform tasks from the past. Such dynamic reliability obviates the need to re-train algorithms on data or tasks to which it has been exposed in the past, thus improving its data-

efficiency. Secondly, we believe that algorithms which perform consistently well across a multitude of tasks are more trustworthy, a desirable trait sought by medical professionals (Spiegelhalter, 2020). Based on these observations, we are interested in tackling the following question.

Research Question

How can we design clinical algorithms that remain robust to data distribution changes over space, time, and modalities?

In this chapter, we address the outlined research question in the context of diagnosing cardiac arrhythmias based on the electrocardiogram. To begin, we exploit and adapt the continual learning paradigm which is characterized by the sequential streaming of data from multiple sources (Thrun, 1998). Specifically, we modify two essential components of methods that use a replay buffer; the storage of data encountered in the past, and the acquisition of data from a buffer, at some point in the future. Our contributions are threefold. First, we propose an importance-based buffer storage mechanism and validate its interpretation qualitatively. Second, we take inspiration from our work in Chapter 4 and propose an uncertainty-based buffer acquisition mechanism. Third, we show that our proposed framework, CLOPS, outperforms the state-of-the-art replay-based methods, GEM (Lopez-Paz and Ranzato, 2017) and MIR (Aljundi et al., 2019a), on three diverse datasets and in three continual learning scenarios.

8.1 Related Work

Continual learning (CL). In recent years, a multitude of continual learning approaches have resurfaced (Parisi et al., 2019). Those similar to ours comprise memory-based methods such as iCaRL (Rebuffi et al., 2017), CLEAR (Rolnick et al., 2019), GEM (Lopez-Paz and Ranzato, 2017), and aGEM (Chaudhry et al., 2018). In contrast to our work, the latter two methods naively populate their replay buffer with the last m examples observed for a particular task. Isele and Cosgun (2018) and Aljundi

et al. (2019b) employ a more sophisticated buffer-storage strategy where a quadratic programming problem is solved in the absence of task boundaries. Aljundi et al. (2019a) introduce MIR whereby instances are stored using reservoir sampling and sampled according to whether they incur the greatest change in loss if parameters were to be updated on the subsequent task. This approach is computationally expensive, requiring multiple forward and backward passes per batch. The application of CL in the medical domain is limited to that of Lenga et al. (2020) wherein existing methodologies are implemented on chest X-ray datasets. In contrast to previous research that independently investigates buffer-storage and acquisition strategies, we focus on a *dual* storage and acquisition strategy.

8.2 BACKGROUND

8.2.1 *Continual Learning*

We assume access to a set of T tasks, $\{\mathcal{D}_k\}_{k=1}^{T}$, where $\mathcal{D}_k = \{\boldsymbol{x_{ik}}, y_{ik}\}_{i=1}^{N}$ is a dataset comprising N instances, $\boldsymbol{x_{ik}}$, and corresponding labels, y_{ik}, from a particular task, $k \in [1 \dots T]$. Let us consider a learner, $f_\omega : \boldsymbol{x} \in \mathbb{R}^D \rightarrow \hat{y} \in \mathbb{R}^C$, parameterized by $\boldsymbol{\omega}$, that maps a D-dimensional input, $\boldsymbol{x}$, to a C-dimensional output, $\hat{y}$, where C is the number of classes.

In continual learning, a learner is exposed to tasks in a sequential manner once previously-tackled tasks are mastered. We formulate our tasks based on a modification of the three-tier categorization proposed by Van Looveren and Klaise (2019). In our learning scenarios (see Fig. 8.1), a network is sequentially tasked with solving (a) a binary classification problem in response to data from mutually-exclusive pairs of classes **Class Incremental Learning (Class-IL)**, (b) a multi-class classification problem in response to data collected at different times of the year (e.g., winter and summer) **Time Incremental Learning (Time-IL)**, and (c) a multi-class classification problem in response to inputs with a different modality **Domain Incremental Learning (Domain-IL)**. In the aforementioned cases, task identities are *absent* during both training and

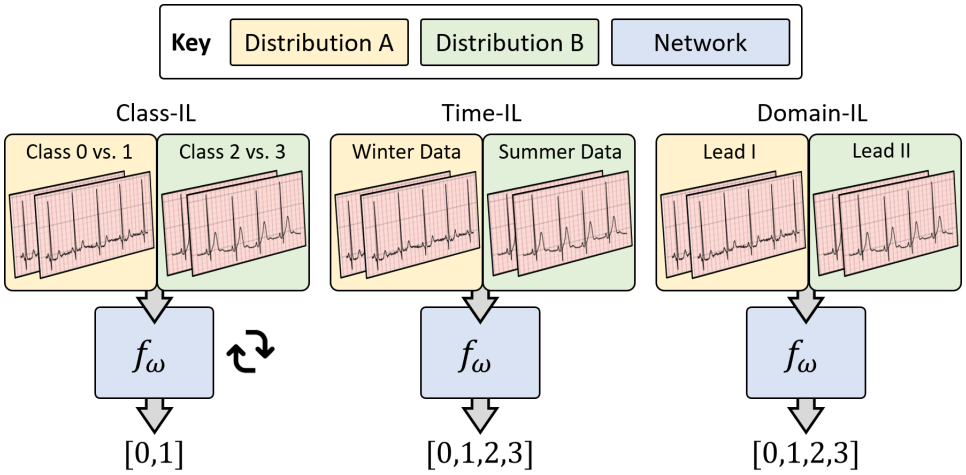


Figure 8.1: **Three continual learning scenarios relevant to the clinical domain.** A network is sequentially exposed to tasks **(Class-IL)** with mutually-exclusive pairs of classes, **(Time-IL)** with data collected at different times of the year, and **(Domain-IL)** with data from different input modalities.

testing and neural architectures are single-headed. The absence of task identities implies that when performing a forward-pass or inference, the network does not have information about the task to which the data belong. This setting is considered to be more realistic and challenging than that with task information.

8.3 METHODS

The two ideas underlying our proposal are the storage of instances into *and* the acquisition of instances from a buffer such that catastrophic forgetting, also known as destructive interference, is mitigated. We describe these in more detail below.

8.3.1 *Importance-Guided Buffer Storage*

We aim to populate a buffer, $\mathcal{D}_B$, of finite size, $\mathcal{M}$, with instances from the current task that are considered important. To quantify importance, we learn parameters, entitled task-instance parameters, β_{ik}, associated with each instance, $\boldsymbol{x_{ik}}$, in each task, k. These parameters play a dual role, as explained next.

Loss-weighting mechanism. For the current task, k, and its associated data, $\mathcal{D}_k$, we incorporate β as a coefficient of the loss, $\mathcal{L}_{ik}$, incurred for each instance, $\boldsymbol{x_{ik}} \in \mathcal{D}_k$.

For a mini-batch of size, B, that consists of B_k instances from the current task, the objective function is shown in (8.1). We can learn the values of β_{ik} via gradient descent, with some learning rate, η.

$$\mathcal{L} = \frac{1}{B_k} \sum_{i=1}^{B_k} \beta_{ik} \mathcal{L}_{ik} \qquad \beta_{ik} \leftarrow \beta_{ik} - \eta \frac{\partial \mathcal{L}}{\partial \beta_{ik}} \tag{8.1}$$

Note that $\frac{\partial \mathcal{L}}{\partial \beta_{ik}} = \mathcal{L}_{ik} > 0$. This suggests that instances that are hard to classify ($\uparrow \mathcal{L}_{ik}$) will exhibit $\downarrow \beta_{ik}$. From this perspective, β_{ik} can be viewed as a proxy for instance difficulty. However, as presented, $\beta_{ik} \to 0$ as training progresses, an observation we confirmed empirically. Since β_{ik} is the coefficient of the loss, $\mathcal{L}_{ik}$, this implies that the network will quickly be unable to learn from the data. To avoid this behaviour, we initialize $\beta_{ik} = 1$ in order to emulate a standard loss function and introduce a regularization term to penalize its undesirable and rapid decay toward zero. As a result, our modified objective function is:

$$\mathcal{L}_{\text{current}} = \frac{1}{B_k} \sum_{i=1}^{B_k} \beta_{ik} \mathcal{L}_{ik} + \lambda (\beta_{ik} - 1)^2 \tag{8.2}$$

When the network has completed at least one task ($k > 1$), we replay instances from previous tasks by using a replay buffer (described in depth in Sec. 8.3.2). In the process, B_j instances are replayed from all previous tasks, $j \in [1 \dots k-1]$, and incur a loss of $\mathcal{L}_{ij}$. Since each training mini-batch of size, B, is shared between instances in the current task, of which there are B_k, and those from past tasks, the replayed loss is defined as shown in (8.3). Note that we do not introduce weight coefficients for replayed instances, in contrast to what we perform to instances from the current task (see Appendix C.2.2).

$$\mathcal{L}_{\text{replay}} = \frac{1}{B - B_k} \sum_{j=1}^{k-1} \sum_{i}^{B_j} \mathcal{L}_{ij} \qquad \mathcal{L} = \mathcal{L}_{\text{current}} + \mathcal{L}_{\text{replay}} \tag{8.3}$$

Buffer-storage mechanism. As a proxy for instance difficulty, we leverage β to store instances into the buffer. To describe the intuition behind this process, we illustrate, in Fig. 8.2, the trajectory of β_{1k} and β_{2k} associated with two instances, $\boldsymbol{x}_{1k}$ and $\boldsymbol{x}_{2k}$, while training on the current task, k, for $\tau = 20$ epochs. In selecting instances for storage into the buffer, we can 1) retrieve their corresponding β values at the *conclusion* of the task, i.e., at $\beta(t = 20)$, 2) rank all instances based on these β values, and 3) acquire the top b fraction of instances. This approach, however, can lead to *erroneous* estimates of the relative difficulty of instances, as explained next.

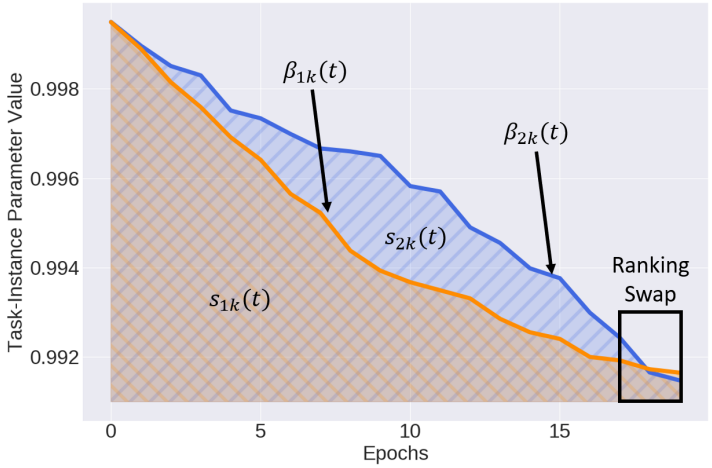


Figure 8.2: **Intuition behind ranking and storing instances based on the area under the trajectory of β values.** We show the trajectory of β_{1k} and β_{2k} on task k. Ranking instances based on $\beta(t = 20)$ leads to erroneous estimates of their relative difficulty. We propose to rank instances based on the area under the trajectory of β, denoted as s_{ik}.

In Fig. 8.2, we see that $\beta_{2k}(t) > \beta_{1k}(t)$ for the majority of the training process, $0 < t < 20$, indicating that $\boldsymbol{x_{2k}}$ had been easier to classify by the network than $\boldsymbol{x_{1k}}$. However, the swap in the ranking of these β values that occurs towards the end of training implies that solely looking at $\beta(t = 20)$ would have *erroneously* made us believe that $\boldsymbol{x_{1k}}$ was easier to classify by the network than $\boldsymbol{x_{2k}}$. A similar phenomenon has been observed by Saxena et al. (2019). Therefore, solely depending on $\beta(t = 20)$ would have eroded its reliability as a proxy of instance difficulty.

To maintain the reliability of this proxy, we propose to *track* the β values during training, record its value every epoch, t, until the final epoch, τ, for the task at

hand, and calculate the area under these tracked values. The area is calculated via the trapezoidal rule as shown in (8.4). We explored several variants of the storage function and found the proposed form to work best (see Appendix C.2.2). At $t = \tau$, we rank the instances in descending order of s_{ik} (easy to hard) as we found this preferable to the opposite order (see Appendix C.2.2), select the top b fraction, and store them into the buffer, of which each task is allotted a fixed portion. The higher the value of the storage fraction, b, the more likely it is that the buffer will contain representative instances and thus mitigate forgetting, however this comes at an increased storage cost.

$$s_{ik} = \int_0^{\tau} \beta_{ik}(t)dt \approx \sum_{t=0}^{\tau} \left(\frac{\beta_{ik}(t + \Delta t) + \beta_{ik}(t)}{2} \right) \Delta t \tag{8.4}$$

8.3.2 *Uncertainty-Based Buffer Acquisition*

The acquisition of instances that a learner is uncertain about is likely to benefit training (Zhu, 2005). This is the premise of uncertainty-based acquisition functions such as BALD (Houlsby et al., 2011; Gal and Ghahramani, 2016). We now outline how to exploit this premise for buffer acquisition.

At epoch number, τ_{MC}, referred to as Monte Carlo (MC) epochs, each of the M instances in the buffer, $\mathcal{D}_B = \{\boldsymbol{x}_B, y_B\}$, is passed through the network and exposed to a stochastic binary dropout mask to generate a probability distribution over the C classes, $p(y|\boldsymbol{x}_B, \boldsymbol{\omega}) \in \mathbb{R}^C$. This is repeated T times to form a matrix, $\boldsymbol{G} \in \mathbb{R}^{M \times T \times C}$. An acquisition function, such as BALD_{MCD}, is thus a function $\mathcal{F} : \boldsymbol{G} \in \mathbb{R}^{M \times T \times C} \rightarrow \boldsymbol{\alpha} \in \mathbb{R}^M$, that maps from the matrix, $\boldsymbol{G}$, to a list of scalars, $\boldsymbol{\alpha}$, that is subsequently used to rank datapoints (described next).

$$\text{BALD}_{\text{MCD}} = \text{JSD}(p_1, p_2, \ldots, p_T) = \mathbb{H}\left(p(y|\boldsymbol{x})\right) - \mathbb{E}_{p(\boldsymbol{\omega}|D_{train})}\left[\mathbb{H}\left(p(y|\boldsymbol{x}, \hat{\boldsymbol{\omega}})\right)\right] \tag{8.5}$$

where $\mathbb{H}(p(y|\boldsymbol{x}))$ represents the entropy of the network outputs averaged across the MC samples, and $\hat{\boldsymbol{\omega}} \sim p(\boldsymbol{\omega}|D_{train})$ as in Gal and Ghahramani (2016). At sample epochs, τ_S, we rank instances in descending order of BALD_{MCD} and acquire the top a fraction from each task in the buffer. A higher value of this acquisition fraction, a, implies more instances are acquired. Although this may not guarantee improvement in performance, it does guarantee increased training overhead. Nonetheless, the intuition is that by acquiring instances from previous tasks to which a network is most confused, it can be nudged to avoid destructive interference in a data-efficient manner. We outline the entire training procedure in Algorithms 8.1-8.4 below.

Algorithm 8.1 CLOPS

Input: MC epochs τ_{MC}, sample epochs τ_S, MC samples T, storage fraction b, acquisition fraction a, task data $\mathcal{D}_k$, buffer $\mathcal{D}_B$, training epochs per task τ

for $(\boldsymbol{x}, y) \sim \mathcal{D}_k$ **do**
  calculate $\mathcal{L}$ using (8.3)
  update β_{ik} ▷ update task-instance parameters
  if epoch = τ **then**
    $\mathcal{D}_B$ = StoreInBuffer($\mathcal{D}_B$, β_{ik}, b, $\mathcal{D}_k$)
  if epoch in τ_{MC} **then**
    $\boldsymbol{G}$ = MonteCarloSamples($\mathcal{D}_B$)
  if epoch in τ_S **then**
    $\mathcal{D}_k$ = AcquireFromBuffer($\mathcal{D}_B$, $\boldsymbol{G}$, a, $\mathcal{D}_k$)

Algorithm 8.2 StoreInBuffer

Input: buffer $\mathcal{D}_B$, task-instance parameters β_{ik}, storage fraction b, task data $\mathcal{D}_k$

calculate s_{ik} using (8.4)
SortDescending(s_{ik})
$(\boldsymbol{x}_b, y_b) \subset \mathcal{D}_k$ ▷ obtain instances with highest s_{ik}
$\mathcal{D}_B \leftarrow \mathcal{D}_B \cup (\boldsymbol{x}_b, y_b)$

Algorithm 8.3 MonteCarloSamples

Input: buffer $\mathcal{D}_B$

for $(\boldsymbol{x}, y) \sim \mathcal{D}_B$ **do**
  for MC sample in T **do**
    obtain $p(y|\boldsymbol{x}, \hat{\boldsymbol{\omega}})$ and store in $\boldsymbol{G} \in \mathbb{R}^{M \times T \times C}$

Algorithm 8.4 AcquireFromBuffer

Input: buffer $\mathcal{D}_B$, MC posterior distributions $\boldsymbol{G}$, acquisition fraction a, task data $\mathcal{D}_k$

1: calculate $\boldsymbol{\alpha}$ using (8.5)
2: SortDescending($\boldsymbol{\alpha}$)
3: $(\boldsymbol{x}_a, y_a) \subset \mathcal{D}_B$ ▷ acquire instances with highest α
4: $\mathcal{D}_k \leftarrow \mathcal{D}_k \cup (\boldsymbol{x}_a, y_a)$

8.4 EXPERIMENTAL DESIGN

8.4.1 *Data and Pre-processing*

To evaluate our method, we leverage three different datasets, each of which consists of cardiac time-series waveforms alongside cardiac arrhythmia labels. We split each of the aforementioned waveforms into non-overlapping segments comprising 2500 samples. In Table 8.1, we present a summary of these datasets.

Table 8.1: **Summary of the datasets used for evaluation.** We also show additional pre-processing information. Please click on the dataset's name for more information.

Dataset	Abbreviation	Modality	Normalization	Exclusion Criteria
Cardiology	$\mathcal{D}_1$	ECG	✗	sinus bradycardia cases
Chapman	$\mathcal{D}_2$	ECG	✗	-
PhysioNet 2020	$\mathcal{D}_3$	ECG	✓	-

8.4.2 *Continual Learning Scenarios*

We simulated three environments with changing dynamics in which the network was sequentially tasked with performing cardiac arrhythmia classification.

In the class incremental learning (Class-IL) scenario, the network solved a binary classification problem in response to data from mutually-exclusive pairs of cardiac arrhythmia classes. In our context, we split the Cardiology dataset (Hannun et al., 2019) based on the following class-pairs $[0, 1]$, $[2, 3]$, $[4, 5]$, $[6, 7]$, $[8, 9]$, and $[10, 11]$. We chose the Cardiology dataset for this scenario to avoid complications that may have arisen due to multi-label segments in other datasets, such as PhysioNet 2020. This scenario allowed us to evaluate the sensitivity of a network to new classes.

In the time incremental learning (Time-IL) scenario, the network solved a multi-class classification problem in response to data collected at different times of the year (e.g., winter and summer). In our context, we split the Chapman dataset (Zheng et al., 2020) into three tasks; Term 1, Term 2, and Term 3 corresponding to mutually-exclusive dates of the year during which patient data were collected. We chose the Chapman dataset for this scenario as it was the only one that reported the recording date of the ECG signals. This scenario allowed us to evaluate the effect of temporal non-stationarity on system's performance.

In the domain incremental learning (Domain-IL) scenario, the network solved a multi-class classification problem in response to inputs with a different modality. In our context, we split the PhysioNet 2020 dataset (Wagner et al., 2020) according to the 12 leads of an ECG, which can be considered as 12 different projections of the same electrical signal generated by the heart. Although the Chapman dataset also has 12 leads, we chose the PhysioNet 2020 dataset for this scenario in order to evaluate our methods on a range of diverse ECG signals. This scenario allowed us to evaluate the robustness of a system to changes in the input distribution.

Each of continual learning scenarios are broken down into tasks which consist of their own training, validation, and test sets. We present, in Table 8.2, the number of instances available in these sets.

Table 8.2: **Number of instances in the training, validation, and test sets of tasks in CL scenarios.** The three continual learning scenarios are Class-IL, Time-IL, and Domain-IL.

Task Name	0-1	2-3	4-5	6-7	8-9	10-11
Training	781	227	463	2118	309	179
Validation	126	141	118	587	83	82
Test	285	77	130	703	89	102

(a) Class-IL, $\mathcal{D}_1$

Task Name	Term 1	Term 2	Term 3
Training	37596	12534	12552
Validation	20586	6858	6864
Test	18424	6143	6139

(b) Time-IL, $\mathcal{D}_2$

Task Name	Lead I	Lead II	Lead III	...	Lead V6
Training	11598	11598	11598	...	11598
Validation	3238	3238	3238	...	3238
Test	4041	4041	4041	...	4041

(c) Domain-IL, $\mathcal{D}_3$

8.4.3 *Baseline Methods*

We compare our proposed method to the following: **Multi-task Learning (MTL)** (Caruana, 1993) is a strategy whereby all datasets are assumed to be available at the same time and thus can be simultaneously used for training. Although this assumption may not hold in clinical settings due to the nature of data collection, privacy or memory constraints, it is nonetheless a strong baseline. **Fine-tuning** is a strategy that involves updating all parameters when training on subsequent tasks as they arrive without explicitly dealing with catastrophic forgetting. We also adapt two replay-based methods for our scenarios. **GEM** (Lopez-Paz and Ranzato, 2017) solves a quadratic programming problem to generate parameter gradients that do not increase the loss incurred by replayed instances. **MIR** (Aljundi et al., 2019a) replays instances from a buffer that incur the greatest change in loss given a parameter pseudo-update. Details on how these methods were adapted are found in Appendix C.2.1.

8.4.4 *Evaluation Metrics*

To evaluate our methods, we exploit metrics suggested by Lopez-Paz and Ranzato (2017) such as average AUC and Backward Transfer (BWT). We also propose two additional evaluation metrics that provide us with a more fine-grained analysis of learning strategies.

t-Step backward transfer. To determine how performance changes 't-steps into the future', we propose BWT_t which evaluates the performance of the network on a previously-seen task, after having trained on t-tasks after it.

$$\text{BWT}_t = \frac{1}{T-t}\sum_{j=1}^{T-t} \text{R}_j^{j+t} - \text{R}_j^j \tag{8.6}$$

Lambda backward transfer. We extend BWT_t to all time-steps, t, to generate BWT_λ. As a result, we can identify improvements in methodology at the task-level.

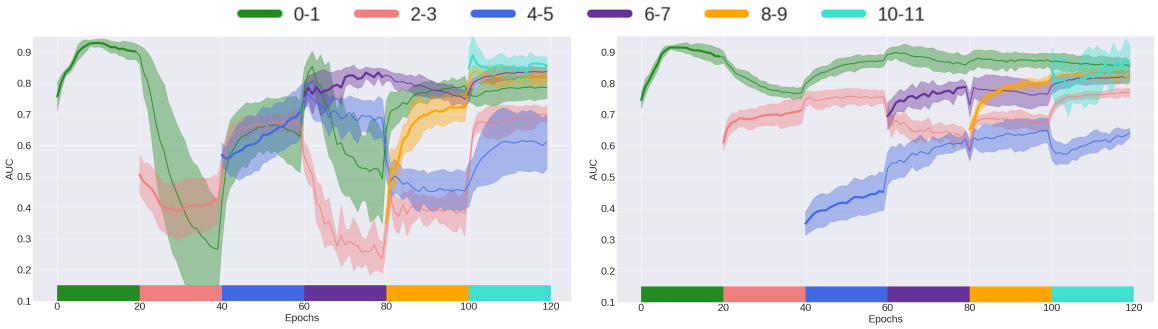
$$\text{BWT}_\lambda = \frac{1}{T-1}\sum_{j=1}^{T-1}\left[\frac{1}{T-j}\sum_{t=1}^{T-j}\text{R}_j^{j+t} - \text{R}_j^j\right] \tag{8.7}$$

8.5 RESULTS

In this section, and with an eye on addressing the initial research question, we attempt to answer the following questions: (i) How does our framework perform in the class incremental learning scenario; when novel disease classes are presented to the algorithm? (ii) How does our framework perform in the time incremental learning scenario; when data with seasonal changes are presented to the algorithm? (iii) How does our framework perform in the domain incremental learning scenario; when data from multiple domains are presented to the algorithm? (iv) Can we leverage task-instance parameters to quantify the similarity between tasks?

8.5.1 *Class Incremental Learning*

Destructive interference is notorious amongst neural networks. In this section, we quantify such interference when learners are exposed to tasks involving novel classes. In Fig. 8.3a, we illustrate the AUC achieved on sequential binary classification tasks. We find that destructive interference is prevalent. For example, the network quickly forgets how to perform task $[0-1]$ once exposed to data from task $[2-3]$. This can be seen by the AUC $\approx 0.92 \rightarrow 0.30$. The final performance of the network for that particular task (AUC ≈ 0.78) is also lower than that maximally-achieved. In Fig. 8.3b, we show that CLOPS alleviates this interference. This can be seen by the absence of significant drops in AUC and higher final performance for all tasks relative to the fine-tuning strategy.



(a) Fine-tuning (b) CLOPS

Figure 8.3: **Mean validation AUC in the Class-IL scenario** of the a) fine-tuning and b) CLOPS strategy ($b = 0.25$ and $a = 0.50$). Coloured blocks indicate tasks on which the learner is currently being trained. The shaded area represents one standard deviation across five seeds.

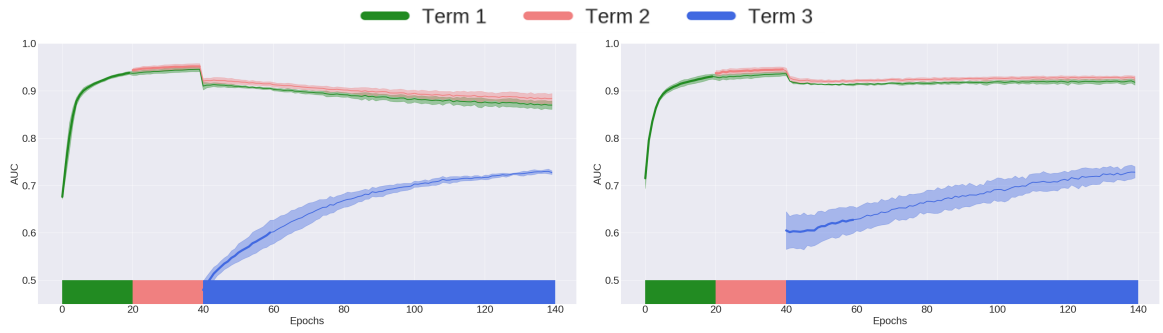
Table 8.3: **Performance of CL strategies in the Class-IL scenario.** Storage and acquisition fractions are $b = 0.25$ and $a = 0.50$, respectively. Mean and standard deviation are shown across five seeds.

Method	Average AUC	BWT	BWT_t	BWT_λ
MTL	0.701 ± 0.014	-	-	-
Fine-tuning	0.770 ± 0.020	0.037 ± 0.037	-0.076 ± 0.064	-0.176 ± 0.080
Replay-based Methods				
GEM	0.544 ± 0.031	-0.024 ± 0.028	-0.046 ± 0.017	-0.175 ± 0.021
MIR	0.753 ± 0.014	0.009 ± 0.018	0.001 ± 0.025	-0.046 ± 0.022
CLOPS	**0.796 ± 0.013**	**0.053 ± 0.023**	**0.018 ± 0.010**	**0.008 ± 0.016**

In Table 8.3, we compare the performance of the CL strategies in the Class-IL scenario. We find that CLOPS outperforms MTL (AUC = 0.796 vs. 0.701), which is a seemingly non-intuitive finding. We hypothesize that this finding is due to positive weight transfer brought about by a curriculum wherein sequential tasks of different levels of difficulty can improve generalization performance (Bengio et al., 2009). We explore this hypothesis further in Sec. 8.5.5. We also find that CLOPS outperforms State-of-the-art methods, GEM and MIR, in terms of generalization performance and exhibits constructive interference. For example, CLOPS and MIR achieve an AUC = 0.796 and 0.753, respectively. Moreover, BWT = 0.053 and 0.009 for these two methods, respectively. Such a finding underscores the ability of CLOPS to deal with tasks involving novel classes. We also show that CLOPS is robust to task order (see Appendix C.2.2).

8.5.2 *Time Incremental Learning*

Environmental changes within healthcare can introduce seasonal shift into datasets. In this section, we quantify the effect of such a shift on learners. In Fig. 8.4a, we illustrate the AUC achieved on tasks with seasonally-shifted data.



(a) Fine-tuning (b) CLOPS

Figure 8.4: **Mean validation AUC in the Time-IL scenario** of the (a) fine-tuning and (b) CLOPS strategy. Coloured blocks indicate tasks on which the learner is currently being trained. The shaded area represents one standard deviation from the mean across five seeds.

Table 8.4: **Performance of CL strategies in the Time-IL scenario.** Storage and acquisition fractions are $b = 0.25$ and $a = 0.50$, respectively. Mean and standard deviation values are shown across five seeds. In some settings, the quadratic program (QP) of GEM could not be solved.

Method	Average AUC	BWT	BWT_t	BWT_λ
MTL	**0.971 ± 0.006**	-	-	-
Fine-tuning	0.824 ± 0.004	-0.020 ± 0.005	-0.007 ± 0.003	0.010 ± 0.001
Replay-based Methods				
GEM	*QP problem could not be solved*			
MIR	0.856 ± 0.010	-0.007 ± 0.006	-0.003 ± 0.004	0.001 ± 0.004
CLOPS	0.834 ± 0.014	-0.018 ± 0.004	-0.007 ± 0.003	0.007 ± 0.003

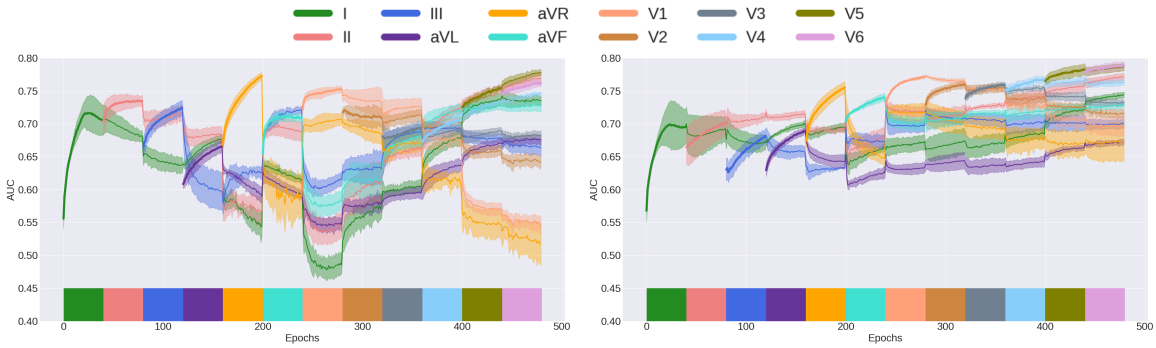
In this scenario, we find that CLOPS is capable of achieving forward weight transfer (FWT). For example, in Figs. 8.4a and 8.4b, CLOPS achieves an AUC ≈ 0.62 after one epoch of training on task Term 3, a value that the fine-tuning strategy only achieves after 20 epochs, signalling a 20-fold reduction in training time. We attribute this FWT to the loss-weighting role played by the task-instance parameters. By placing greater emphasis on more useful instances, the generalization performance

of the network is improved. We also find that CLOPS exhibits reduced catastrophic forgetting relative to fine-tuning. For example, performance on tasks Term 1 and Term 2 is maintained at AUC > 0.90 when training on task Term 3. We do not observe this for the fine-tuning setup.

8.5.3 *Domain Incremental Learning*

So far, we have shown the potential of CLOPS to alleviate destructive interference and allow for forward weight transfer. In this section, and in Table 8.5, we illustrate the performance of the CL strategies in the Domain-IL scenario. We show that CLOPS outperforms State-of-the-art methods. For example, CLOPS and MIR achieve an AUC $= 0.731$ and 0.716, respectively. CLOPS is also better at mitigating destructive interference, as shown by BWT $= -0.011$ and -0.022, respectively. We provide an explanation for such performance by conducting ablation studies in the next section.

To gain a better understanding of the learning dynamics of CLOPS in the domain-IL scenario, we plot the validation AUC in Fig. 8.5. Here, significant destructive interference occurs in the fine-tuning strategy. This is shown by large drops in the AUC of one task when subsequent tasks are trained on. For instance, when training on Lead V1 (starting at epoch 240), performance on Lead I (green) drops from an AUC $= 0.725 \rightarrow 0.475$, completely erasing any progress that had been made on that lead. We hypothesize that this is due to the different representations of instances belonging to Lead V1 and Lead I. Anatomically speaking, these two leads are projections of the same electrical signal in the heart that are oriented approximately 180 degrees to one another. This detrimental behaviour is absent for almost all leads when CLOPS is implemented, as shown in Fig. 8.5b. A notable exception is Lead aVR at epoch 200 where performance drops from an AUC $\approx 0.75 \rightarrow 0.65$.



(a) Fine-tuning (b) CLOPS

Figure 8.5: **Mean validation AUC in the Domain-IL scenario** of a) fine-tuning and b) CLOPS ($b = 0.25$ and $a = 0.50$) strategy. Each task belongs to the same dataset yet different input modality. Coloured blocks indicate tasks on which the learner is currently being trained. The shaded area represents one standard deviation from the mean across 5 seeds.

Table 8.5: **Performance of CL strategies in the Domain-IL scenario.** Storage and acquisition fractions are $b = 0.25$ and $a = 0.50$, respectively. Mean and standard deviation are shown across five seeds.

Method	Average AUC	BWT	BWT_t	BWT_λ
MTL	0.730 ± 0.016	-	-	-
Fine-tuning	0.687 ± 0.007	-0.041 ± 0.008	-0.047 ± 0.004	-0.070 ± 0.007
Replay-based Methods				
GEM	0.502 ± 0.012	-0.025 ± 0.008	**0.004 ± 0.010**	-0.046 ± 0.021
MIR	0.716 ± 0.011	-0.022 ± 0.011	-0.013 ± 0.004	-0.019 ± 0.006
CLOPS	**0.731 ± 0.001**	**-0.011 ± 0.002**	-0.020 ± 0.004	-0.019 ± 0.009

8.5.4 *Effect of Task-Instance Parameters and Acquisition Function*

To better understand the root cause of CLOPS' benefits, we conduct additional studies investigating the marginal effect of our storage and acquisition mechanisms on performance. These mechanisms are dependent upon the *amount* of data that were stored and acquired, and as such, we conducted these studies while varying the fraction of data that are stored in the buffer and that are acquired from the buffer. These are denoted by the storage fraction, b, and acquisition fraction, a. In the *random storage* study, we dispense with our storage mechanism and instead randomly stored ECG signals in the buffer. In the *random acquisition* study, we dispense with our acquisition mechanism and instead randomly acquire ECG signals from the buffer.

Lastly, in the *random storage and acquisition* study, we store ECG signals in, and acquire them from, the buffer randomly. We present the resulting AUC of these experiments in Fig. 8.6.

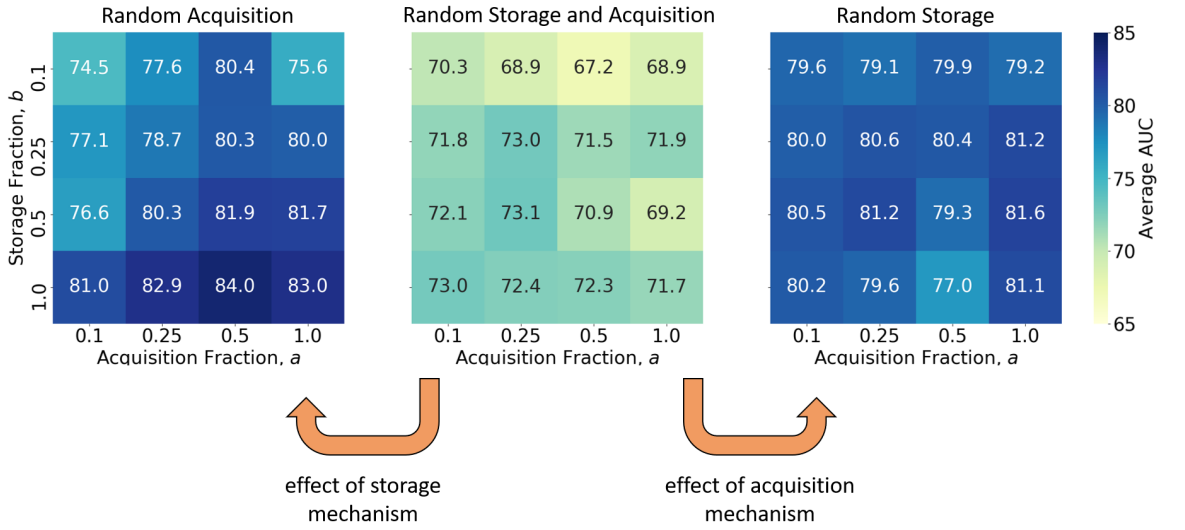


Figure 8.6: **Marginal benefit of storage and acquisition mechanisms on performance of CLOPS.** We show three different learning strategies in the Class-IL scenario. (**Random storage and acquisition**) stores instances into, and acquires them from, the buffer randomly. (**Random acquisition**) stores instances into the buffer using our importance-based strategy and acquires them from the buffer randomly. (**Random storage**) stores instances into the buffer randomly and acquires them from the buffer using our uncertainty-based strategy. The results are shown as a function of storage fractions, b, and acquisition fractions, a and are an average across five seeds. Improvement in performance of the *random acquisition* and *random storage* learning strategies relative to the *random storage and acquisition strategy* points to the benefit of our storage and acquisition mechanisms, respectively.

Our storage mechanism contribute drastically to the generalization performance of our network. Specifically, the incorporation of the storage mechanism increases the AUC of the network regardless of the amount of data that are stored and acquired from the buffer. For example, when we only store 10% of the ECG signals in the buffer ($b = 0.1$) and acquire 50% of the ECG signals from the buffer ($a = 0.5$), we observe an improvement in the AUC $= 67.2 \rightarrow 80.4$, reflecting a 13.2% improvement. Such a finding points to how indispensable our storage mechanism is.

When we independently evaluate the acquisition mechanism, we show that it also contributes drastically to the generalization performance of our network. Specifically,

the incorporation of the acquisition mechanism increases the AUC of the network regardless of the amount of data that are stored and acquired from the buffer. For example, when we only store 10% of the ECG signals in the buffer ($b = 0.1$) and acquired 10% of the ECG signals from the buffer ($a = 0.1$), we observ an improvement in the AUC $= 70.3 \rightarrow 79.6$, reflecting a 9.3% improvement. Such a finding, particularly with such a small storage and acquisition fraction of ECG signals, points to how robust our acquisition mechanism can be to scarce data environments. Although we presented results illustrating the generalization performance, we arrive at similar conclusions when we evaluated the degree to which these mechanisms alleviate catastrophic forgetting.

8.5.5 *Validation of Interpretation of Task-Instance Parameters*

We claimed that our framework was learning to identify important ECG signals for their eventual storage in a buffer. We then mathematically showed the equivalency of this importance with the difficulty with which the network diagnosed the cardiac arrhythmia of ECG signals. To validate this claim empirically, we explore and visualize the importance parameters that were learned. In Fig. 8.7, we illustrate the distribution of these parameters for all ECG signals and across all tasks that the network was sequentially exposed to.

We find that the network perceived various tasks to differ in their level of difficulty. For example, the network struggles more to solve the cardiac arrhythmia task $[6-7]$ relative to the task $[8-9]$. This is supported by the observation that the the parameter values of the distribution of task $[6-7]$ are lower than those of task $[8-9]$. To validate that these distributions were indeed indicative of the difficulty with which ECG signals were diagnosed, we also identify the two ECG signals associated with the lowest and highest importance parameter values and present them alongside the distributions. Based on our setup, these two ECG signals should correspond to the most and least difficult signals to diagnose, respectively. We find that our

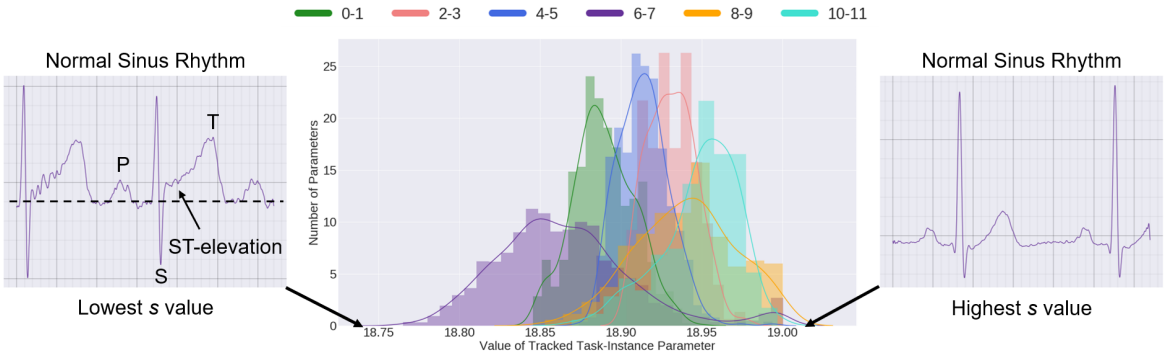


Figure 8.7: **Distribution of the s values in the Class-IL scenario corresponding to** CLOPS **($b = 0.25$ and $a = 0.50$).** Each colour corresponds to a different task. The ECG recording with the lowest s value is labelled as normal despite the presence of ST-elevation, a feature common in heart attacks. This could indicate that the recording is in fact abnormal and is incorrectly labelled or that the recording is normal and that minor feature changes have thrown off the network.

expectations were indeed corroborated by basic ECG domain expertise. For example, both of these signals had a ground-truth, cardiologist-derived label of normal sinus rhythm. However, the ECG signal deemed most difficult by the network exhibited morphological aberrations, such as ST-elevation, a typical feature in certain cardiac abnormalities. Such a feature could have confused the network and hindered its ability to diagnose the ECG signal correctly. We provide additional qualitative evidence in Appendix C.2.2. Such a finding reaffirms our interpretation of the importance parameters as a proxy for the difficulty with which an ECG signal is diagnosed. As a result, we have a tool that allows us to peer into the network and better understand its inner workings.

In addition to validating our interpretation of the storage parameters qualitatively, we set out to do so more quantitatively. We take inspiration from the curriculum learning literature (Bengio et al., 2009) which has shown that the order with which data are presented to a learning system can impact the system's generalization capabilities. Specifically, we exploit the storage parameters, s, to design several curricula based on the notion of task difficulty and similarity, as explained next. First, we fit a Gaussian distribution, $\mathcal{N}(\mu_k, \sigma_k^2)$, to each of the six distributions shown in Fig. 8.7. Using this

information, we define the difficulty of task, k, as $d_k = \frac{1}{\mu_k}$ and the similarity, $S(j,k)$, between task j and k based on the Hellinger distance (8.8).

$$S(j,k) = 1 - \underbrace{\sqrt{1 - \sqrt{\frac{2\sigma_j\sigma_k}{\sigma_j^2\sigma_k^2}} e^{-\frac{1}{4}\frac{(\mu_j-\mu_k)^2}{\sigma_j^2\sigma_k^2}}}}_{\mathcal{D}_H = \text{Hellinger Distance}} \tag{8.8}$$

In Fig. 8.8, we illustrate the resulting pairwise task similarity matrix for tasks in the Class-IL scenario. For this particular example, we find that task $[8-9]$ is most similar to task $[10-11]$. Conversely, task $[0-1]$ and $[10-11]$ are identified as being least similar to one another, based on our definition of similarity. Insight from such a similarity matrix, although preliminary, has a twofold effect. First, when coupled with clinical domain expertise, it has the potential to supplement clinical knowledge by potentially identifying differences between medical conditions and patient cohorts, depending on the chosen task definition. Second, by allowing researchers to identify which tasks are believed to be different by the learner, it could facilitate transfer learning across tasks, domain adaptation, and even curriculum learning. To illustrate this point, we experimented with the latter.

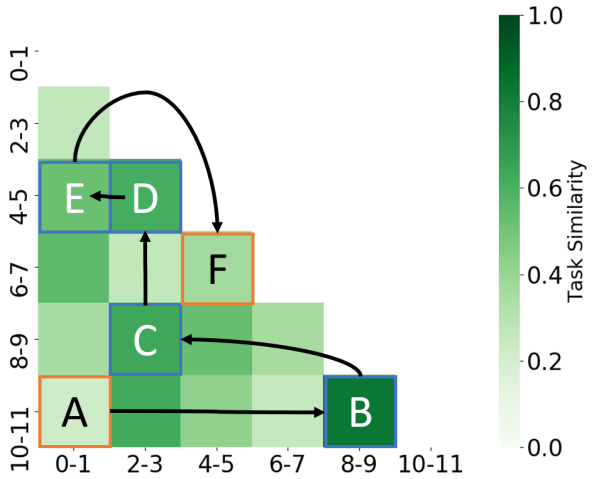


Figure 8.8: **Similarity of tasks in the Class-IL scenario overlaid with the chain of tasks used for curriculum learning.** The curriculum begins with the task identified as being the easiest ($[10-11]$), which is chained to similar tasks by following the arrows, and concludes with the task identified as being the most difficult ($[6-7]$). The effect of such a curriculum on the learning process can be found in Table 8.6.

We design a curriculum by first selecting the easiest task ($\downarrow d_{\mathcal{T}}$, task $[10-11]$), based on Fig. 8.7, and then creating a chain of tasks that are similar to one another, based on Fig. 8.8. This chaining process is illustrated by the arrows in Fig. 8.8. Conversely, for an anti-curriculum, we repeat the process except that we started with the hardest task ($\uparrow d_{\mathcal{T}}$, task $[6-7]$). In Table 8.6, we show the performance of the CLOPS as a result of these various curricula.

Table 8.6: **Effect of various curricula on the performance of CLOPS in the Class-IL scenario.** The storage and acquisition fractions are $b = 0.25$ and $a = 0.50$, respectively. The mean and standard deviation (SD) are shown across five seeds. Bold results reflect the curricula leading to the best performance. The constructive interference exhibited by CLOPS following a curriculum based on our task-instance parameters supports our interpretation of such parameters as a proxy for instance difficulty.

Task Order	Average AUC	BWT	BWT_t	BWT_λ
Random	**0.796 ± 0.013**	0.053 ± 0.023	0.018 ± 0.010	0.008 ± 0.016
Curriculum	0.744 ± 0.009	**0.087 ± 0.011**	**0.038 ± 0.021**	**0.076 ± 0.037**
Anti-curriculum	0.783 ± 0.022	0.058 ± 0.016	-0.013 ± 0.013	-0.003 ± 0.014

We find that CLOPS achieves the highest constructive interference when trained with a curriculum (easy $\rightarrow$ hard) as opposed to when trained with an anti-curriculum or randomly-ordered tasks. For example, with a curriculum, BWT $= 0.087$ whereas BWT $= 0.058$ and 0.053 with an anti-curriculum and randomly-ordered tasks, respectively. We hypothesize that transitioning from easy to hard tasks along a chain of similar tasks allowed CLOPS to efficiently maintain knowledge from one task to the next. Such a finding, which aligns well with the broader expectations of curriculum learning, further supports the intuition that our storage parameters act as a proxy for the difficulty of instances. However, we also find that such improved constructive interference comes at the cost of generalization performance. This was evident by the AUC $= 0.744$ and 0.796 achieved by CLOPS when trained with a curriculum and randomly-ordered tasks, respectively. We hypothesize that maintaining knowledge from the past hindered the learner's ability to perform as well on the current task. In other words, in this particular set of curriculum learning scenarios, we find that there may exist a trade-off between "remembering the past" and "facing the future".

We hypothesize that this trade-off exists due to the limited capacity of replay-based methods more generally. Specifically, tasks must compete for the same set of scarce resources (parameters), and this is likely to lead to interference of the parameters.

As alluded to earlier, the literature on quantifying task similarity is rich. In our context, we decided to quantify the similar of tasks by exploiting task-instance parameters. Such task similarity quantification can have a multitude of downstream applications. We had presented one example in which we used it for the design of a learning curriculum. Another application can include exploiting task similarities to regularize a multi-task objective function optimized over a graph (Nassif et al., 2020). In this case, the task similarity-inspired regularization term can be used to promote the relationship between tasks. Doing so can lead to the learning of smoother parameters which contribute to improved generalization performance.

9

MULTILINGUAL CAPTIONING OF CARDIAC SIGNALS

أَنا من هناك. أَنا من هنا ولستُ هناك، ولستُ هنا لِيَ اسمانِ يلتقيان ويفترقان

ولي لُغتان، نسيت بأيَّهما كنتُ أَحلُمُ

محمود درويش —

I am from there. I am from here and I am not there, I am not here. I have two names meeting and parting and I have two languages, I have forgotten in which I used to dream

— Mahmoud Darwish

As we approach the resource-rich end of the resource spectrum, we take the liberty of exploiting a multitude of modalities. In doing so, we aim to demonstrate the potential impact of multi-modal data on streamlining certain clinical workflows. While continuing to focus on the cardiac arrhythmia diagnosis workflow, we direct our attention to the process that follows the diagnosis itself, as described next.

After making a diagnosis, an expert cardiologist is often required to generate an accompanying textual report, and share it with fellow physicians (or patients) to complement a patient's medical records during a hospital visit. For example, an ECG is of maximal value when (and is legally required to be) accompanied by a textual report (Richley and Walters, 2020). Such reports, however, can be time-consuming to generate and detract physicians from caring for patients. They also exhibit a high degree of ambiguity and intra/inter-physician variability which erodes

patient-physician communication (Hibbard et al., 2001; Keselman and Smith, 2012). Automating the generation of ECG reports can streamline the clinical workflow for cardiologists, allow for more consistent ECG interpretations (Willems et al., 1991), and potentially reduce the 'inadvertent oversight' of medical conditions (Brailer et al., 1997). Based on these observations, we are interested in tackling the following question.

Research Question

How can we design clinical algorithms that generate accurate and plausible clinical reports that summarize cardiac signals?

In this chapter, we address the outlined research question in the context of generating ECG reports in response the electrocardiogram signal. We refer to this task as cardiac signal captioning and take it upon ourselves to generate ECG reports in multiple languages. Our contributions are threefold. First, in light of the absence of publicly-available datasets comprising ECG signals and multilingual reports, we translate reports paired with cardiac signals into seven different languages (the first of its kind) and make them available via open-source to facilitate research into multilingual cardiac captioning. Second, equipped with such a rich dataset, we design a cardiac signal captioning system that generates clinical textual reports, in various languages, in response to cardiac signals. Last, in the absence of sufficient labelled data, deep neural networks can benefit from a pre-training procedure in which parameters are first learned on an arbitrary task. We propose such a task, entitled RTLP, in the form of discriminative multilingual pre-training. In this setting, words from clinical reports are randomly replaced with those from other languages and the network is tasked with predicting the language of all words.

9.1 RELATED WORK

Visual and language representation learning. Representation learning is an integral component of textual and visual systems. In the former, generative language rep-

resentation learning tasks such as masked language modelling (MLM) have proven effective (Devlin et al., 2018; Liu et al., 2019b). These generative methods have also been extended to the multilingual case (Conneau and Lample, 2019; Conneau et al., 2019; Liu et al., 2020). Most similar to our work is ELECTRA (Clark et al., 2020), wherein tokens (or words) are replaced with those from a generative model, and a network is tasked with discriminating between the original and the replaced tokens. This approach simultaneously learns an MLM and discriminative network, deeming it computationally expensive and dependent on large datasets. In contrast, our work is not dependent on an additional MLM network and explicitly deals with the multilingual setting. MARGE (Lewis et al., 2020) retrieves documents, in potentially different languages, and attempts to reconstruct a target document. Instead of focusing on a single modality, others propose to learn textual and visual representations jointly (Sun et al., 2019; Lu et al., 2019; Zhang et al., 2020c). To the best of our knowledge, we are the first to propose a discriminative multilingual language representation learning method in the context of cardiac signals.

Multilingual representation learning. Pre-training and fine-tuning networks on multiple languages has been shown to benefit natural language processing (NLP) tasks (Conneau et al., 2019; Pratap et al., 2020; Conneau et al., 2020; Artetxe et al., 2020). For example, Conneau et al. (2019) and Artetxe et al. (2020) show that multilingual pre-training is more advantageous than its monolingual counterpart when solving downstream NLP tasks. In some cases, this has been attributed to commonalities across languages such as word order, characters, and semantic structure. These findings hav been partially fueled by monolingual datasets which have been machine-translated to multiple languages, such as XNLI (Conneau et al., 2018). In this work, we translate ECG reports into multiple languages and leverage that for pre-training and fine-tuning purposes. Most similar to our multilingual pre-training setup is that of (Huang et al., 2019a). Our work differs in that it explores more languages, defines a different pre-training task, and leverages that for cardiac captioning.

Image captioning (IC) in healthcare. Image captioning, a formal description of which is relegated to Section 9.2, focuses on generating a summary caption of an image. In biomedical IC, most research has focused on chest X-rays. For example, Hasan et al. (2018) propose to incorporate clinical concept prediction to improve the captioning performance. Kisilev et al. (2016); Zeng et al. (2020) introduce a multi-task objective to simultaneously perform bounding box regression and captioning. Liu et al. (2019a) condition their captioning system on the medical topic to be discussed and Wang et al. (2018b) propose a multi-level attention model that attends to both the image and the text. More recently, methods such as ClinicalBert (Huang et al., 2019b), BioBert (Lee et al., 2020), and BioELMo (Jin et al., 2019) were shown to learn rich representations of clinical text. These representations are also beneficial to biomedical applications (Yoon et al., 2019). To the best of our knowledge, we are the first to explore multilingual captioning in the context of cardiac signals.

9.2 BACKGROUND

9.2.1 *Cardiac Signal Captioning*

We begin by assuming access to a dataset, $\mathcal{D} = \{\boldsymbol{x}_i, \text{cap}_i\}_{i=1}^{N}$ comprising N cardiac signals, $\boldsymbol{x} \in \mathbb{R}^D$, and their associated captions, $\text{cap} = \{w_s\}_{s=1}^{S}$, which consist of S words, w_s. The goal of cardiac signal captioning is to generate a caption (report) that reliably summarizes the physiological state of a patient as manifested in a cardiac signal. To extract features from the signal, an encoder, $f_\theta : \boldsymbol{x} \in \mathbb{R}^D \rightarrow \{\boldsymbol{v}_t \in \mathbb{R}^M\}_{t=1}^{T}$, parameterized by $\boldsymbol{\theta}$, maps a D-dimensional instance, $\boldsymbol{x}$, to a set of T representations, $\{\boldsymbol{v}_t \in \mathbb{R}^M\}_{t=1}^{T}$, each of which is M-dimensional (see Fig. 9.1 left).

To convert the captions into a usable format, we first convert each word, w_s, in a caption, $\text{cap} = \{w_s\}_{s=1}^{S}$, to a token, u_s, to form a sequence of tokens $\{u_s\}_{s=1}^{S}$. Such tokenization can involve lower-casing and stemming words, in addition to removing punctuation. After deriving tokens from all captions in a training set, we form a fixed vocabulary, $V = \{u_i\}_{i=1}^{C}$, of the C unique tokens where $|V| = C$. We then define

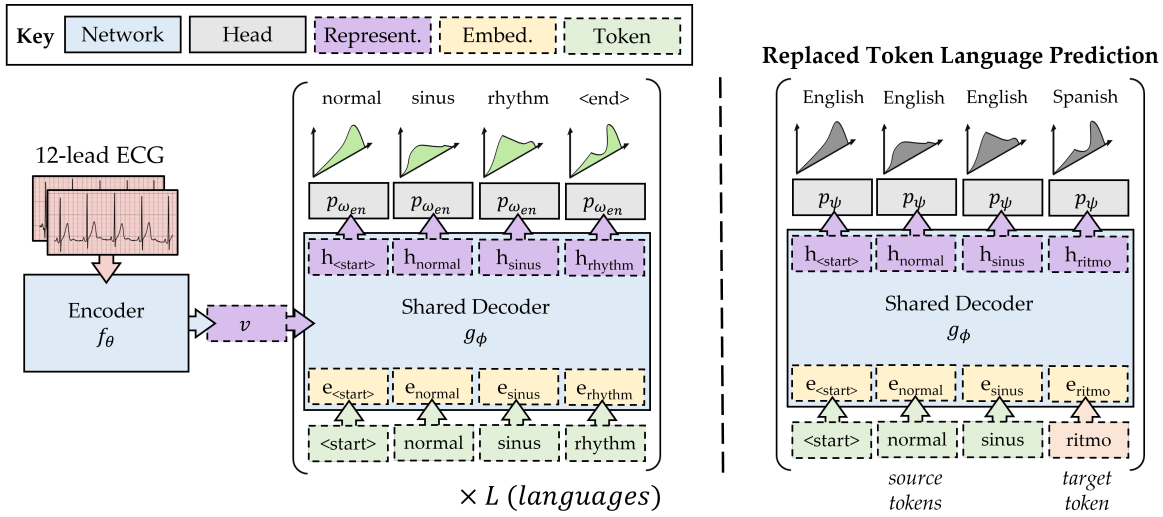


Figure 9.1: **(Left) Multilingual cardiac signal captioning pipeline.** We feed a 12-lead ECG into an encoder, f_θ, to extract representations, $\boldsymbol{v}$. These are fed, alongside embeddings, $\boldsymbol{e}$, of tokens from a particular language, to a decoder, g_ϕ, to generate token representations, $\boldsymbol{h}$. We feed $\boldsymbol{h}$ into a language-specific head, p_{ω_l}, to generate a caption in a specific language. **(Right) Replaced token language prediction framework**. We randomly replace source tokens with those from a target language and task the network with classifying the language of all tokens. In doing so, we encourage the network to capture relationships between representations of tokens from different languages.

an embedding matrix (lookup table), $\boldsymbol{E} \in \mathbb{R}^{C \times M} : u \rightarrow \boldsymbol{e} \in \mathbb{R}^M$, which maps each token, u, to an M-dimensional token embedding, $\boldsymbol{e}$. These embeddings are typically randomly-initialized and learned in an end-to-end manner via gradient descent. As such, every caption can now be represented as a sequence of token embeddings, $\{\boldsymbol{e}_s\}_{s=1}^S$.

To extract features from language, a decoder, $g_\phi : \{\boldsymbol{v}_k\}_{k=1}^K, \{\boldsymbol{e}_s\}_{s=1}^S \rightarrow \{\boldsymbol{h}_s \in \mathbb{R}^M\}_{s=1}^S$, parameterized by $\boldsymbol{\phi}$, attends to all *encoder* representations, $\{\boldsymbol{v}_k\}_{k=1}^K$, while mapping a token embedding, $\boldsymbol{e}_s$, at step, s, to an M-dimensional token representation, $\boldsymbol{h}_s$. Each token representation, $\boldsymbol{h}_s$, in the sequence of representations, $\{\boldsymbol{h}_s\}_{s=1}^S$, is then fed into a linear classification head, $p_{\boldsymbol{\omega}} : \boldsymbol{h}_s \in \mathbb{R}^M \rightarrow \boldsymbol{y}_s \in \mathbb{R}^C$, to output a probability distribution, $\boldsymbol{y}_s$, over the C tokens in the vocabulary. This forms a sequence of outputs, $\{\boldsymbol{y}_s\}_{s=1}^S$. At each step, s, in the sequence, the goal is to maximize the likelihood of generating the token of the next step, u_{s+1}. Therefore, by identifying the most probable output token at each step, we can form a sentence of words (i.e., a caption).

9.3.1 *Multilingual Captioning of Cardiac Signals*

From a practical perspective, a monolingual captioning framework would appear to be sufficient for physicians communicating in a single language. However, the motivation for a *multilingual* captioning framework, in which reports are simultaneously generated in multiple languages, is threefold. First, a multilingual framework would obviate the cumbersome process of having to train a distinct model for different languages (Conneau et al., 2020). Second, recent work has demonstrated the benefits of incorporating additional languages into the learning process (Artetxe et al., 2020). Lastly, a multilingual framework exhibits greater flexibility than its monolingual counterpart, as it can always be collapsed, during inference, to generate captions from a single language.

To enable multilingual captioning, we first assume access to a set of L language-specific datasets, $\{\mathcal{D}_l\}_l^L$ where $\mathcal{D}_l = \{\boldsymbol{x}_i, \text{cap}_i^l\}_{i=1}^N$ comprises N cardiac signals, $\boldsymbol{x}$, and captions in a specific language, $l \in \mathbb{L} = \{\text{en}, \text{es}, \cdots\}$, where en and es represent English and Spanish, respectively. Note that the cardiac signals are *shared* across the datasets. To generate captions in multiple languages, we follow the same encoder-decoder approach mentioned in the previous section with one exception. We now replace the single classification head with L language-specific heads to account for the distinct vocabularies of the L languages. In doing so, we exploit recent observations that demonstrated the utility of having a network with parameters that are both language-specific and shared across languages (Zhang et al., 2020a).

Formally, for each language, l, we have a linear classification head, $p_{\boldsymbol{\omega}_l} : \boldsymbol{h}_s^l \in \mathbb{R}^M \rightarrow \boldsymbol{y}_s^l \in \mathbb{R}^C$, with $\boldsymbol{\omega}_l \in \{\boldsymbol{\omega}_{\text{en}}, \boldsymbol{\omega}_{\text{es}}, \cdots\}$ reflecting language-specific parameters. Each head maps the token representation, $\boldsymbol{h}_s^l$, at each step, s, in the sequence to a probability distribution, $\boldsymbol{y}_s^l$, over C tokens where $C \in \{|V_{\text{en}|}, |V_{\text{es}|}, \cdots\}$ reflects the size of a language-specific token vocabulary. When doing this for each step in the

sequence, we arrive at a set of probability distributions, $\{\boldsymbol{y}_s^l\}_{s=1}^S$. As with traditional language models, at each step in the sequence, $s \in [1, S]$, we maximize the likelihood of observing the next token in the sequence, u_{s+1}^l, from a particular language, $l \in \mathbb{L}$. Therefore, for a mini-batch of B captions in a single language, we would optimize the categorical cross-entropy loss at each step, s. To extend this to L languages, we load L mini-batches and optimize the following *multi-task* categorical cross-entropy loss.

$$\mathcal{L}_{\text{multilingual}} = -\frac{1}{LBS} \sum_{l \in \mathbb{L}}^{L} \sum_{i=1}^{B} \sum_{s=1}^{S} \log p_{\omega_l}(y_{i,s}^l = u_{i,s+1}^l) \tag{9.1}$$

9.3.2 *Replaced Token Language Prediction*

To facilitate achieving the downstream task of multilingual cardiac signal captioning, we design a discriminative multilingual pre-training task. At a high-level, this task involves randomly selecting tokens in a sequence, replacing them with semantically-similar tokens from a different language, and tasking a network with classifying the language of all tokens (see Fig. 9.1 right).

The intuition behind our framework is that a network exposed to semantically-similar tokens from distinct languages which share the same context (neighbouring tokens) can learn that such tokens are indeed similar to one another. We hypothesize that encouraging this behaviour can lead to the learning of token representations that are beneficial particularly for the downstream task of *multilingual* captioning since these tokens are also likely to arise in the same generated report, albeit in a different language. As such, when multilingual reports are generated, they might be more likely to contain the appropriate tokens. From hereon forward, we refer to this method as replaced token language prediction (RTLP) and describe its mechanics next.

SOURCE TOKEN SELECTION Given tokens, $\{u_s^{l_{src}}\}_{s=1}^S$, in a sequence of length, S, from a source language, $l_{src} \in \mathbb{L}$, we first sample K distinct steps, $\{s_k\}_{k=1}^K$ where

$s_k \in [1, S]$ from a uniform distribution, $\mathcal{U}$. We then replace the corresponding source tokens, $\{u_{s_k}^{l_{src}}\}_{k=1}^{K}$, with those from a target language, $\{u_{s_k}^{l_{tgt}}\}_{k=1}^{K}$, as explained next.

TARGET LANGUAGE AND TOKEN SELECTION For each source token, $u_{s_k}^{l_{src}}$, we sample a target language, $l_{tgt} \sim \mathcal{U}(\mathbb{L}')$, uniformly at random from the set of remaining languages $\mathbb{L}' = \mathbb{L} \setminus l_{src}$ where $|\mathbb{L}'| = L - 1$. Given l_{tgt}, we now sample a target token, $u_{s_k}^{l_{tgt}} \sim \mathcal{U}(V_{l_{tgt}})$, uniformly at random from the language-specific vocabulary of tokens, $V_{l_{tgt}}$. Such random sampling, however, can lead to the selection of a target token that is semantically different from the source token. As such, the network, when discriminating between source and target tokens, may learn a detrimental shortcut that is based on semantics and not language.

To avoid this behaviour, we instead adopt a strategy whereby that target token is likely to be semantically similar to (e.g., a noisy translation of) the source token. Formally, we quantify the cosine similarity, sim_j, between the source token embedding, $\boldsymbol{e}_{s_k}^{l_{src}} \in \mathbb{R}^M$, and the embedding, $\boldsymbol{e}_j^{l_{tgt}} \in \mathbb{R}^M$ of each token, $u_j^{l_{tgt}} \in V_{l_{tgt}}$. We then take the softmax of these similarities to form a categorical distribution, q, with elements, q_j, and sample $u_{s_k}^{l_{tgt}}$ from this categorical distribution, as shown below. As the token embeddings become more meaningful, the sampled target token is more likely to be semantically similar to the source token.

$$u_{s_k}^{l_{tgt}} \sim q \quad , \quad \text{where} \quad q_j = \frac{\exp^{(\text{sim}_j)}}{\sum_m^{|V_{l_{tgt}}|} \exp^{(\text{sim}_m)}} \quad , \quad \text{sim}_j = \frac{\boldsymbol{e}_{s_k}^{l_{src}} \cdot \boldsymbol{e}_j^{l_{tgt}}}{|\boldsymbol{e}_{s_k}^{l_{src}}||\boldsymbol{e}_j^{l_{tgt}}|} \tag{9.2}$$

OBJECTIVE FUNCTION Equipped with a sequence comprising tokens in source and target languages, we define a classification head, $p_{\boldsymbol{\psi}} : \boldsymbol{h} \rightarrow \boldsymbol{y} \in \mathbb{R}^L$, parameterized by $\boldsymbol{\psi}$, that maps each token representation, $\boldsymbol{h}$, to a probability distribution over L languages. Formally, given a mini-batch comprising B captions of length, S, and in the source language, l, we optimize the categorical cross-entropy loss for both the tokens in the source language ($r = 0$) and those that are replaced ($r = 1$). To extend this to

L source languages, we load L mini-batches and optimize the following *multi-task* categorical cross-entropy loss where $\mathbb{1}$ is the indicator function.

$$\mathcal{L}_{\text{RTLP}} = -\frac{1}{LBS}\sum_{l\in\mathbb{L}}^{L}\sum_{i=1}^{B}\sum_{s=1}^{S}\mathbb{1}_{r=0}\cdot\log p_\psi(y_{i,s}=l_{src}|u_{i,s}^{l_{src}}) + \mathbb{1}_{r=1}\cdot\log p_\psi(y_{i,s}=l_{tgt}|u_{i,s}^{l_{tgt}}) \tag{9.3}$$

9.4 EXPERIMENTAL DESIGN

9.4.1 *Data and Pre-processing*

We focus on a dataset that consists of the electrocardiogram alongside a paired textual report. To that end, we leverage the publicly-available **PTB-XL** dataset (Wagner et al., 2020) which comprises 12-lead ECG recordings from 18,885 patients alongside ECG reports, which are predominantly in English and German. It also consists of cardiac arrhythmia labels which we group into 5 major classes (Strodthoff et al., 2020).

We remind readers of our assumption of access to datasets that consist of cardiac signals and *multilingual* medical reports. To the best of our knowledge, such datasets do not exist. For example, the PTB-XL dataset (Wagner et al., 2020) comprises cardiac signals and ECG reports predominantly in German. As a result, we set out to generate such multilingual reports. More specifically, we follow a similar strategy to that proposed by Conneau et al. (2018) and translate reports to multiple languages using the Google Translate API[1]. Further details on this process can be found in Appendix C.3.1. Although such translation can introduce artifacts, we hypothesize (and indeed show) that the net effect on downstream performance will remain advantageous.

9.4.2 *Captioning of Cardiac Signals*

REPRESENTATION LEARNING OF CARDIAC SIGNALS Supervised pre-training continues to be an effective way to learn rich, generalizable representations. To that

[1] https://pypi.org/project/googletrans/

end, we pre-train the *encoder*, f_θ, to map 12-lead ECG signals to abnormalities in the functioning of the heart (cardiac arrhythmias). This choice is motivated by our observation that ECG reports are likely to reflect the pathology of a cardiac arrhythmia. Therefore, learning representations of cardiac signals that are discriminative along the dimension of cardiac arrhythmias may benefit the downstream task of cardiac signal captioning.

REPRESENTATION LEARNING OF CLINICAL REPORTS To implement RTLP, we exploit L language-specific tokenizers from SpaCy to tokenize the ECG reports. For simplicity, we lower-case the text and remove any punctuation. By keeping track of unique tokens, we form L distinct vocabularies. Each vocabulary also includes language-specific tokens to indicate the start and end of the report, and the [PAD] and [OOV] tokens to refer to padded entries and tokens observed during inference that are not seen during training, respectively. We also introduce the [MASK] token where appropriate.

MULTILINGUAL CARDIAC SIGNAL CAPTIONING After pre-training the encoder, f_θ, and the decoder, g_ϕ, independently of one another, we exploit the learned parameters, $\{\theta, \phi\}$, and token embeddings to solve the task of cardiac signal captioning (see Fig. 9.1 left). After experimenting with several variants of our framework, we chose to *freeze* the encoder parameters and extract multiple representations per cardiac signal. These encoder representations are used in the cross-attention mechanism of the Transformer decoder. Multilingual captioning also requires a ground-truth ECG report to be available in multiple languages. Since such *paired* reports do not exist, we translate the original set of reports in English (en) to six languages [German (de), Greek (el), Spanish (es), French (fr), Italian (it), and Portuguese (pt)] using the Google Translate API, a strategy also adopted by Conneau et al. (2018). We open-source

these reports[2] and provide further details in Appendix C.3.1. Although translated reports can be imperfect ground-truth reports, we hypothesize that the net effect on downstream performance is advantageous.

9.4.3 *Evaluation of Generated Captions*

As we are mainly interested in the cardiac captioning task, we leverage three automatic metrics commonly used to evaluate image-captioning (BLEU score (Papineni et al., 2002), METEOR (Banerjee and Lavie, 2005) and ROUGE−L (Lin, 2004)). At a high-level, these metrics quantify the degree of overlap of n-grams between a ground-truth sentence and a generated sentence. An n-gram can be thought of as a combination of tokens (words) that neighbour one another. For example, a 1-gram simply consists of all individual tokens in a sentence whereas a 2-gram consists of all pairs of adjacent tokens in the sentence.

BLEU Score. The BLEU score first requires calculating the precision of n-grams for a particular value of n. This precision is defined as the number of overlapping n-grams between the generated and ground-truth sentence, $Overlap_n$, divided by the total number of n-grams in the generated sentence $Total_n$. Such a calculation is repeated for multiple values of $n \in [1, \ldots, N]$ before being averaged and weighted according to a brevity penalty, BP, which penalizes generated sentences which are shorter than the ground-truth sentence.

$$\text{BLEU} - \text{N} = BP \cdot \left(\Pi_{n=1}^{N} Precision_n \right)^{\frac{1}{N}} \qquad Precision_n = \frac{Overlap_n}{Total_n} \tag{9.4}$$

METEOR Score. The METEOR score was designed, for the most part, to explicitly account for recall in n-gram overlap calculations, an operation not included in the BLEU score. Specifically, it aligns the ground-truth and generated sentences with one another, calculates the *unigram* (1-gram) precision and recall, and derives an F-score

[2] Code and data can be found at: https://tinyurl.com/CardiacSignalCaptioning

with greater emphasis on recall than on precision. Similar to BLEU, it weights the F-score with a sentence brevity penalty, BP.

$$\text{METEOR} = \frac{10 \cdot Precision \cdot Recall}{9 \cdot Precision + Recall} \tag{9.5}$$

ROUGE Score. Although there exist multiple ROUGE scores (e.g., ROUGE-N, ROUGE-L, ROUGE-W, and ROUGE-S), we opt for the ROUGE-L score because it obviates the need to set a value of n for the n-grams. Formally, ROUGE-L calculates the F-score (geometric mean of precision and recall) of the longest common subsequence (LCS) of tokens between the ground-truth sentence with m tokens and the generated sentence with n tokens.

$$\text{ROUGE} - \text{L} = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \tag{9.6}$$

$$Precision = \frac{LCS}{n} \quad \text{and} \quad Recall = \frac{LCS}{m}$$

9.4.4 *Baseline Methods*

Our focus is on cardiac signal captioning which exploits multilingual discriminative language representation learning. As such, we compare our method to several language representation learning methods: 1) **MLM**, a masked language modelling pre-training objective (Devlin et al., 2018) where the decoder is tasked with identifying masked tokens, 2) **ELECTRA**, a replaced token detection pre-training objective (Clark et al., 2020) where the decoder is tasked with identifying whether tokens have been replaced with those from an MLM model, and 3) **MARGE**, a multilingual generative language representation learning approach (Lewis et al., 2020) where source documents in various languages are exploited to generate a similar yet distinct target document. Further details on how we adapted these methods can be found in Appendix C.3.2.

9.5 RESULTS

In this section, and with an eye on addressing the initial research question, we attempt to answer the following questions: (i) Does our pre-training paradigm allow for improved performance generalization, in both the monolingual and multilingual settings, relative to state-of-the-art discriminative and generative pre-training methods? (ii) Is our method capable of generating reliable and plausible clinical text in response to cardiac signals? (iii) How does our pre-training paradigm influence the diversity of the clinical text that is generated by the network?

9.5.1 *Quantitative Evaluation of Generated Reports*

We begin by quantitatively evaluating the ability of the pre-training methods to generate high quality multilingual reports. In Table 9.1, we present the BLEU−1, METEOR, and ROUGE−L scores of reports generated in seven different languages.

We find that RTLP performs on par with state-of-the-art generative pre-training methods. For example, on average, RTLP achieves BLEU−1 = 28.5 whereas MLM and MARGE achieve BLEU−1 = 29.4 and 28.9, respectively. This finding holds across

Table 9.1: **Multilingual cardiac signal captioning performance of pre-training methods.** Results are shown on the test set across five seeds. The standard deviation is shown in brackets. For clarity, we have highlighted our method in gray. We find that RTLP performs on par with state-of-the-art language pre-training methods, MLM and MARGE.

Language Pre-training Method	German (**de**)	Greek (**el**)	English (**en**)	Spanish (**es**)	French (**fr**)	Italian (**it**)	Portuguese (**pt**)	**Average**
BLEU-1								
MLM (Devlin et al., 2018)	25.9 (0.6)	20.5 (0.3)	31.3 (0.5)	33.2 (0.8)	29.7 (0.6)	30.3 (0.2)	34.9 (0.7)	29.4 (4.6)
ELECTRA (Clark et al., 2020)	0.1 (0.1)	0.2	0.2 (0.2)	0.3 (0.1)	0.6 (0.1)	0.5 (0.1)	0.5 (0.1)	0.3 (0.2)
MARGE (Lewis et al., 2020)	24.9 (1.0)	19.5 (0.9)	30.8 (0.5)	32.9 (0.5)	29.7 (0.6)	29.4 (0.5)	34.5 (1.0)	28.9 (4.8)
RTLP	25.4 (1.1)	19.8 (0.6)	30.0 (0.7)	33.1 (0.9)	28.3 (0.8)	30.0 (0.1)	33.5 (1.0)	28.5 (4.5)
METEOR								
MLM (Devlin et al., 2018)	36.7 (1.0)	23.6 (0.2)	37.3 (1.1)	38.6 (0.6)	33.5 (0.7)	33.9 (0.7)	38.8 (0.7)	34.6 (5.0)
ELECTRA (Clark et al., 2020)	0.3 (0.5)	0.2 (0.1)	0.2 (0.1)	0.5 (0.4)	1.1 (0.3)	0.9 (0.2)	0.5 (0.2)	0.5 (0.4)
MARGE (Lewis et al., 2020)	35.6 (1.5)	22.2 (1.0)	36.5 (0.8)	37.1 (0.6)	33.1 (1.0)	32.9 (0.9)	37.8 (1.1)	33.6 (5.1)
RTLP	36.5 (0.7)	22.6 (1.0)	36.0 (0.9)	38.5 (0.8)	32.4 (0.4)	33.7 (0.9)	37.6 (0.6)	33.9 (5.1)
ROUGE-L								
MLM (Devlin et al., 2018)	34.6 (0.8)	11.4 (1.9)	28.5 (0.2)	39.3 (1.3)	34.5 (0.8)	36.9 (0.5)	39.1 (0.6)	33.5 (9.3)
ELECTRA (Clark et al., 2020)	0.2 (0.3)	0	0.2	0.5 (0.3)	1.0 (0.2)	0.8 (0.1)	0.5 (0.1)	0.5 (0.4)
MARGE (Lewis et al., 2020)	33.2 (0.7)	11.1 (2.3)	38.1 (0.5)	39.2 (0.6)	34.4 (0.8)	36.1 (0.6)	39.0 (0.5)	33.0 (9.4)
RTLP	34.0 (1.1)	11.6 (2.3)	36.3 (0.9)	39.1 (1.2)	33.1 (0.9)	36.5 (1.0)	37.3 (0.9)	32.6 (8.9)

languages and evaluation metrics. Furthermore, we find that RTLP outperforms the state-of-the-art *discriminative* pre-training method, ELECTRA. For example, on average, RTLP and ELECTRA achieve ROUGE$-$L $= 33.4$ and 0.5, respectively. We note that although we experimented extensively with ELECTRA, we were unable to achieve satisfactory performance (please see GitHub for reproducibility). One hypothesis for this stems from the difficulty of optimizing its objective function which comprises a generative *and* discriminative term. We also find that, regardless of the pre-training method implemented, performance varies significantly across languages. For example, RTLP achieves BLEU$-1 = 19.8$ and 33.5 on Greek (el) and English (en) reports, respectively. We hypothesize that this is due to a high level of dissimilarity between the Greek vocabulary and that of the remaining languages. This, in turn, reduces the amount of knowledge transferred from the other languages to the Greek language. Overall, our findings suggest that RTLP can be an effective discriminative multilingual pre-training method. Qualitative evidence to support this claim is provided in the next section.

9.5.2 *Qualitative Evaluation of Generated Reports*

We also manually inspect the multilingual clinical reports generated by RTLP. Ideally, such reports should accurately reflect clinical information. In Fig 9.2, we present a 12-lead ECG segment alongside the multilingual ground-truth reports and those generated by RTLP and MLM.

There are three main takeaways from Fig 9.2. First, RTLP allows networks to generate reports that accurately capture the aberrant morphology (shape) of the ECG signal. For example, in Spanish (es), the ground-truth and RTLP-generated reports both explicitly mention the cardiac abnormality *'bloqueo de rama izquierda'* (left bundle branch block). Second, we find that RTLP-generated reports manage to capture critical aspects of the ECG signal that their MLM counterparts struggle with. For example, in English (en), both the ground-truth and RTLP-generated reports both

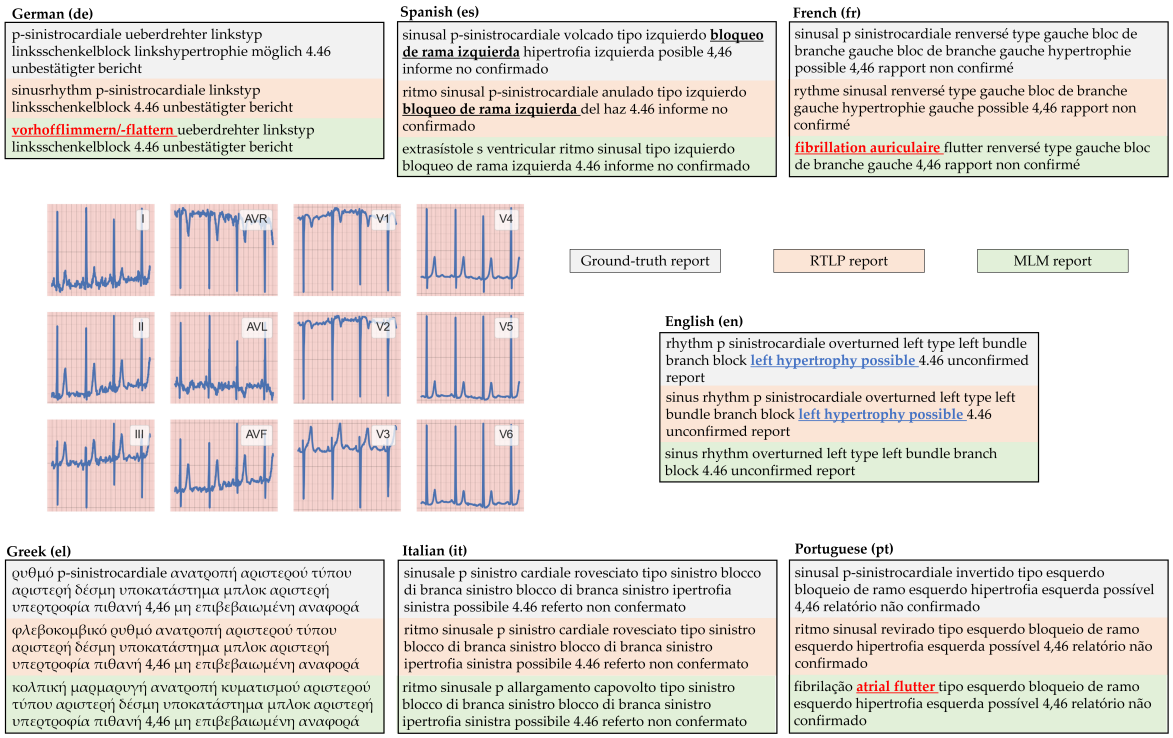


Figure 9.2: **12-lead ECG, multilingual ground-truth reports, and those generated by RTLP and MLM.** We show some phrases in **bold** which exhibit a high level of agreement in both the ground-truth report and that generated by RTLP, in blue which are captured by RTLP and *not* captured by MLM (false negatives), and in red which MLM erroneously includes (false positives). Overall, we show that RTLP can generate reports that accurately capture the pathology of the cardiac signal.

explicitly mention *'left hypertrophy possible'*, whereas this phrase is noticeably absent from the MLM-generated report. Such an absence is problematic as it might result in a physician overlooking this aspect, and thus failing to act accordingly. Lastly, we find that MLM-generated reports can include additional *erroneous* phrases which are neither present in the ground-truth report nor in the RTLP-generated report. For example, in Portuguese (pt), the MLM-generated report mentions *'atrial flutter'*, which is noticeably (and correctly) absent from the remaining reports. Similar erroneous inclusions can also be found in the German (de) and French (fr) reports. This is also problematic since physicians can be mislead by such statements, potentially resulting in unneeded medical treatments. Such findings, combined, indicate that RTLP allows networks to generate reliable and clinically accurate reports.

9.5.3 *Quantifying Diversity of Generated Multilingual Reports*

So far, we have shown that RTLP is capable of generating clinically accurate and plausible reports. It could be argued that generated reports should also exhibit a certain level of diversity in text (Li and Jurafsky, 2016). To illustrate why, consider a constrained system that simply regurgitates the same caption regardless of the input cardiac signal. Such a system would be of minimal clinical value due to its inability to distinguish between the nuances of the cardiac signals. To quantify the diversity of generated reports, we exploit the Self-BLEU metric (Zhu et al., 2018), which measures the BLEU score between all pairs of generated reports. Intuitively, the lower the value of this metric (↓ Self-BLEU), the higher the diversity of the reports (↑ diversity). In Fig. 9.3, we present the Self-BLEU score for three different pre-training methods across all languages in the multilingual setting.

There are several takeaways from Fig. 9.3. First, the degree of diversity exhibited by the generated reports differs across the languages. For example, the French (fr) reports, on average, exhibit the lowest Self-BLUE (≈ 30) and highest diversity whereas the Spanish (es) reports exhibit the least diversity (Self-BLEU ≈ 40). We hypothesize

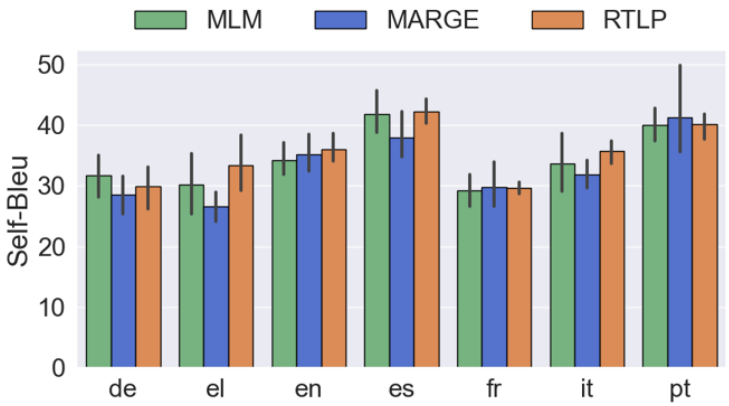


Figure 9.3: **Self-BLEU (diversity) of the reports generated by various pre-training methods.** ↑ Self-BLEU implies increased similarity between reports and thus ↓ diversity. Error bars indicate the standard deviation across five seeds. We show that the diversity of generated reports differ significantly across languages and that the reports generated by different methods exhibit a similar level of diversity, irrespective of the language.

that this could be due to the formation of distinct vocabulary sets of these respective languages. Second, we find that all three methods, MLM, MARGE, and RTLP, generate reports with a similar level of diversity. For example, when generating English (en) reports, these methods achieve Self-BLEU ≈ 35.

9.5.4 *Investigation of the Curse of Multilinguality*

Multilingual neural systems can experience the *curse of multilinguality* (Conneau et al., 2019). Concisely, this attributes the potentially poorer performance of multilingual models relative to their monolingual counterparts to interference between the various languages. Intuitively, tasking a network with generating reports in multiple languages, analogous to multi-task learning (Caruana, 1993), can be too demanding and thus hinder its ability to generate sensible reports. We explore this curse from the lens of the clinical accuracy and diversity of the generated reports. To do so, we first pre-train our networks, as per usual, and fine-tune them in the *monolingual* setting. We then compare the performance of networks and the diversity of the generated reports in the multilingual setting to those in the monolingual setting. In Fig. 9.4, we present such a comparison, illustrating BLEU-1 (top row) and Self-BLEU (bottom row) of reports generated by MARGE and RTLP.

In Fig. 9.4 (top row), we find that MARGE does not experience the curse of multilinguality when evaluated along the dimension of performance. This can be seen by the similar performance achieved by MARGE irrespective of whether it is fine-tuned in the monolingual or multilingual setting. For example, when generating Spanish (es) reports, MARGE achieves BLEU$-1 \approx 32$ in both settings. Such a finding suggests that incorporating additional languages into the fine-tuning process has little to no effect on the quality of the generated reports. In contrast, we find that RTLP benefits significantly from the inclusion of multiple languages. This can be seen by the higher generalization performance ($\uparrow$ BLEU-1) achieved by RTLP in the multilingual setting than in the monolingual setting. For example, for Spanish (es) reports, the

network in both settings achieves BLEU$-1 \approx 32$ and ≈ 18, respectively. This reflects almost a two-fold improvement. We denote this marginal benefit of multilinguality as the *blessing of multilinguality*. We hypothesize that such benefits stem from the transfer and sharing of knowledge across languages. We also note the poorer performance of RTLP in the monolingual setting relative to the multilingual setting. For English (en) reports, RTLP achieves BLEU$-1 \approx 13$ and ≈ 31, respectively. We hypothesize that this is due to the relative degree of importance placed by the model on particular languages during multilingual pre-training with RTLP. In other words, pre-training may implicitly weight languages differently. As such, the learned language-specific token representations may differ in their expressiveness. In light of this, networks fine-tuned in the monolingual setting may perform poorly.

In Fig. 9.4 (bottom row), we find that, on average, MARGE generates reports with a similar level of diversity irrespective of whether it is fine-tuned in the monolingual

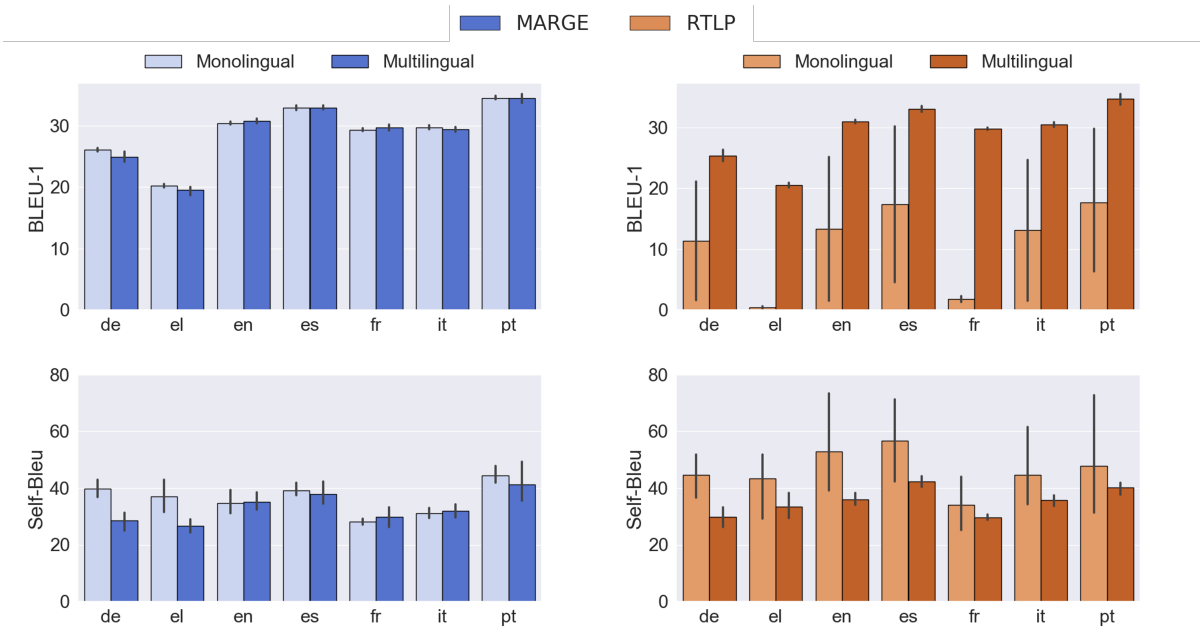


Figure 9.4: **Effect of multilinguality on the (top row) quality and (bottom row) diversity of the generated reports.** The multilingual setting involves simultaneously generating reports in all seven languages, $\mathbb{L} = \{\text{de}, \text{el}, \text{en}, \text{es}, \text{fr}, \text{it}, \text{pt}\}$. Results are shown for all reports and across five random seeds. We show that, through the lens of report quality and diversity, MARGE does not consistently experience the curse of multilinguality. In contrast, RTLP benefits significantly from incorporating multiple languages into the fine-tuning process, a phenomenon denoted by the *blessing of multilinguality*.

or multilingual setting. For example, although MARGE generates more diverse (↓ Self-BLEU) German (de) reports in the latter setting, the opposite holds for French (fr) reports. In contrast, RTLP consistently generates more diverse reports (↓ Self-BLEU) in the multilingual setting than in the monolingual setting. For example, when generating English (en) reports, RTLP achieves Self-BLEU $\approx$ 38 and $\approx$ 52, respectively. Such a finding also contributes to the *blessing of multilinguality*.

10

CONCLUSION

A wide array of research shows that people who are delusionally optimistic tend to outlive people with more realistic attitudes.

— Shankar Vedantam, Hidden Brain

In this thesis, we focused on the challenge that clinical deep learning algorithms are highly dependent on costly resources, such as abundant data, ground-truth annotations, and supervision by medical professionals. We began addressing this challenge by devising a resource spectrum along which distinct paradigms reflected a different amount of resources available to clinical deep learning algorithms. At the low-resource end of the spectrum, we designed learning frameworks with reduced dependence on abundant, labelled data, ground-truth annotations, and, more broadly, supervision. At the high-resource end of the spectrum, we illustrated the potential of clinical deep learning algorithms, when provided with abundant resources, to automate cardiac arrhythmia diagnosis and streamline clinical workflows. We discuss our findings next.

10.1 DISCUSSION OF PROPOSED RESEARCH

In this section, we revisit our proposed frameworks and scrutinize them in depth. Specifically, we discuss their clinical significance, limitations, inter-connectedness, and potential future work.

Generative Learning

In Chapter 3, we proposed several conditional generative adversarial networks in order to generate synthetic medical time-series data as a form of data augmentation.

SIGNIFICANCE. Our proposed cGANs artificially increase the amount and diversity of data that can be used to train a neural network. Data-hungry neural networks can now be trained by medical researchers and practitioners in settings with minimal access to resources (e.g., data). Therefore, existing clinical workflows stand to benefit from such networks.

LIMITATIONS. Neural networks are notoriously data hungry, and this can include cGANs. However, the original motivation behind the use of cGANs was to overcome the challenge of data paucity. Therefore, this setting is similar to that of the 'chicken-and-the-egg'. In other words, do cGANs or data come first? The importance of this question would be moot, however, if the amount of data required to train cGANs is *less* than that required to train a network to achieve a clinical task. In such a setting, the apparent data paucity may be insufficient to solve the clinical task due to the latter's complexity, yet sufficient to train a cGAN. Nonetheless, we did not explore this effect of data availability on the training of cGANs. Such an exploration might have provided practitioners with more concrete guidelines about when cGANs might add value to the learning process.

Furthermore, traditional GANs are exclusively unsupervised; they do not require annotations of any sort. In contrast, cGANs are at the mercy of instance annotations provided by expert physicians. Therefore, errors in the annotation process are likely to negatively affect the training of cGANs. Such errors can manifest in the form of label noise (incorrect labels) or as coarse categories that conceal finer nuances in the data. We did not explore the effect of label noise on our proposed frameworks nor did we explicitly account for such noise during the training of our framework. As a result, there is a possibility that our cGANs, in the event of extreme label noise, may generate synthetic data with incorrect annotations. If such data are used for data augmentation, then the learning process of a clinical network may be hampered.

INTER-CONNECTEDNESS. Recall that the motivation behind our work in Chapter 3 was data augmentation via the generation of synthetic data. Data augmentation,

however, can manifest in various forms and can also be thought of as attempting to exploit invariances in the data. From this perspective, our work in Chapter 3 bears a resemblance to that in Chapter 5 (CLOCS). For example, in Chapter 5, we exploited the notion of transformations and perturbations applied to medical time-series signals. The former involved spatial and temporal invariances present in the electrocardiogram, whereas the latter involved more subtle invariances. With some extensions, the synthetic data generated by a cGAN could have been used in the pre-training paradigm of CLOCS. For example, synthetic data could have (a) been used to form positive and negative pairs of representations or (b) informed the perturbations we applied to cardiac signals.

Active Learning

In Chapter 4, we proposed an active learning framework that exploits both labelled and unlabelled data and dynamically determines whether to annotate such data via an oracle or a pseudo-labelling process.

SIGNIFICANCE. Our active learning framework provides researchers with the opportunity to exploit abundant, unlabelled data at their disposal. In the process, our framework can help (a) accelerate the learning of networks when achieving a clinical task and (b) reduce the annotation burden placed on physicians. The former implies that researchers with fewer computational resources can still train networks to a sufficient level of accuracy. The latter implies that physicians now have more time to reconnect with patients and focus on their needs, and, in turn, improve patient satisfaction.

LIMITATIONS. Our consistency-based acquisition framework involved applying perturbations to both inputs and network parameters and observing changes in the output probability distribution. Such perturbations, if inappropriate, could hamper the progress of the overall active learning process. In our context, we only explored a single input perturbation; Gaussian additive noise. However, such noise may have been sub-optimal relative to other forms of perturbations in the context of active

learning. These perturbations could have included those introduced in Chapter 5, such as temporal or spectral masking of the time-series signal.

In addition to the acquisition framework, we designed a dynamic oracle selection method that manifested in the form of an oracle selection network. The ground-truth label for the outputs of such a network was the zero-one loss incurred by the prediction network. This setup, however, faces several limitations. First, the presence of class label noise would have propagated through the zero-one loss, resulting in an *erroneous* oracle label. For example, let us assume that the prediction network classifies an instance correctly while the corresponding ground truth label is incorrect (due to noise). In this setting, the oracle selection network would erroneously learn that, for such an instance, an oracle would need to be requested for labelling. An oracle, however, would not be needed for this instance given that the network could have correctly provided a pseudo-label. Therefore, the presence of label noise could systematically increase the dependence of our framework on an oracle. The second limitation of the zero-one loss is reflected by its inability to capture near misses in the class predictions. Concretely, an oracle should be more likely to be requested for labelling if an instance is severely misclassified than if it is misclassified by a small margin. The severity of such misclassifications can, for example, be determined by the similarity of disease classes or the repercussions of misclassifications on clinical decision-making.

INTER-CONNECTEDNESS. Active learning, at a high level, can be thought of as simultaneously exploiting both labelled and unlabelled data in order to achieve stronger generalization performance on a particular task. From this perspective, this work shares some similarities with that in Chapter 5 (CLOCS). For example, CLOCS also exploited labelled and unlabelled data, however it did so in a sequential manner. It first exploited unlabelled data for pre-training and transitioned to labelled data for fine-tuning. Despite these high-level similarities, the motivation behind the two frameworks are different. With active learning, our goal was to choose

informative unlabelled instances to be annotated and ultimately incorporated into the learning process. On the other hand, with contrastive learning, our goal was to learn rich, generalizable representations using unlabelled data that could transfer to the downstream task. Although typically applied in distinct settings, these two methods can also be used synergistically with one another. As an example, it is possible that representations learned via CLOCS can allow the active learning framework to better identify informative instances for acquisition.

FUTURE WORK. During the annotation process, our framework was designed to defer to an oracle in the event that the oracle selection network was deemed unreliable. Such a strategy would be problematic, though, if oracles were not capable of providing high-quality annotations. Consequently, one can attempt to leverage relevant *a priori* information to guide the default deferral strategy. For example, knowledge about the degree of noise inherent in an oracle's annotations can be incorporated into our framework, SoQal, by modifying the Hellinger threshold, S. Such a modification would allow users to better handle the trade-off that may exist between annotation burden and accuracy.

Our active learning framework also operated under the assumption that a *single* oracle, at most, was available throughout the learning process. Despite having experimented with various scenarios ranging from the absence of an oracle to the presence of one with a high degree of random or nearest neighbour noise, more challenging scenarios can be explored in the future. For example, within healthcare and depending on the specific domain, the presence of *multiple* oracles is a possibility, e.g., multiple radiologists or dermatologists in a laboratory setting. Moreover, these oracles may have specialty areas within their respective domains, deeming them more skilled at certain tasks than others. To further complicate this scenario, the skill-level of such oracles is likely to exhibit temporal non-stationarity, a phenomenon driven by exhaustion, personal circumstances, and so forth. Designing a dynamic active

learning framework capable of exploiting these non-stationary, skill-varying oracles would be quite valuable given the realistic nature of such a scenario.

Contrastive Learning

In Chapter 5, we proposed a contrastive learning framework, CLOCS, that exploited abundant, unlabelled data to learn rich representations of cardiac signals.

SIGNIFICANCE. Our contrastive learning framework, similar in certain ways to our active learning framework, also allows researchers to exploit abundant, unlabelled data at their disposal. Such exploitation confers several benefits including to (a) accelerate the learning of a network when solving a clinical task and (b) minimize the amount of labelled data required for training a network. These benefits suggest that, given a pre-trained network, fewer resources (both computational and data) are required to train a network. This, in turn, expands the accessibility of neural networks and puts them in the hands of researchers operating in the limited data regime. In doing so, the stakeholders of such communities are likely to benefit from the insight generated by deep learning networks.

LIMITATIONS. CLOCS was anchored around the notion of spatial, temporal, and patient invariances present in cardiac signals. Although such invariances are likely to be present in the setting we laid out in Chapter 5, they may not extend to other scenarios. Such scenarios can involve medical data that are collected over long periods of time (e.g., years) and which comprise multiple modalities (e.g., ECG and coronary angiograms). Specifically, the body's physiological state is likely to vary significantly over time (e.g., due to aging) and manifest differently with different data modalities. As a result, caution must be taken when deploying our pre-training framework to ensure that appropriate invariances are being considered. It is entirely possible that not doing so could result in the learning of representations that are detrimental to the downstream task.

On another note, recall that the CLOCS framework operated on *single-lead* ECG. Specifically, single-lead ECG was exploited for both the pre-training and fine-tuning

process. However, clinical settings more commonly generate and deal with 12-lead ECG signals. At present, and without any modifications, our framework is unable to directly work with such 12-lead ECG signals. One way to accommodate for such signals, after having pre-trained the network on single-lead ECG, is by replicating the convolutional kernels learned in the 1D setting a sufficient number of times to match the number required in the 2D setting (which accepts 12-lead ECG data).

INTER-CONNECTEDNESS. Our CLOCS framework resulted in the learning of representations of instances that happened to be patient specific. This was primarily due to our patient-specific noise contrastive estimation loss. Such a finding inspired, and shares some similarities with, our work in Chapter 6 (PCPs). Specifically, with CLOCS, we assumed that cardiac signals exhibited intra-patient invariance. In other words, instances pertaining to the same patient were assumed to belong to the same class and were thus attracted to one another. With the PCPs in Chapter 6, we continued to exploit this invariance assumption. However, we did so to learn patient-specific embeddings in an end-to-end manner. In this setting, we attracted representations belonging to the same patient to their corresponding patient cardiac prototype. Although this distinction may seem trivial at first, we showed that prototypes allow for additional applications ranging from data distillation to patient retrieval.

FUTURE WORK. We showed that CLOCS leads to the learning of patient-specific representations. This was driven by our redefinition of 'shared context' to incorporate patient information. Within the field of biology, the quantification of entity or patient similarity plays a significant role. In light of this, we believe that representations learned via CLOCS have the potential to contribute to such quantification, primarily at the patient level and perhaps with extensions to other patient attributes. In the process, researchers may be equipped with an additional tool to discover patient sub-cohorts and specific drug treatment and response profiles, insight which could ultimately improve the way in which certain diseases are treated.

Our contrastive learning work was also constrained to a single data modality, namely the electrocardiogram. Despite having shown the utility of our framework in this context, it remains limited in various ways. First, we have not explored how our framework would perform when the upstream (pre-training) and downstream (clinical diagnostic) task do not share the same modality. Conversely, a multitude of unlabelled data modalities can be available for pre-training. For example, in many hospital settings, ECG signals are typically recorded alongside the photoplethysmogram. Although integrating these modalities should be trivial under our CMLC paradigm, simply by attracting representations of temporally-aligned ECG and PPG segments to one another, further studies are required to quantify the benefit of such an approach. More generally, this challenge begs the question of how to best handle multiple modalities with the aim of transferring knowledge to a downstream task.

Patient Cardiac Prototypes

In Chapter 6, we learned patient-specific embeddings, entitled patient cardiac prototypes (PCPs), via a supervised contrastive learning framework.

SIGNIFICANCE. We showed that PCPs serve multiple purposes. First, they can be used to generate patient-specific network parameters for personalized disease diagnosis. As a result, physicians can have access to a tool that provides them with patient-specific diagnoses that go beyond the generic ones typically generated by clinical deep learning algorithms. Such personalized diagnoses could increase physicians' trust in, and their likelihood of adopting, clinical deep learning systems. Second, PCPs can act as efficient data distillers. In other words, they can be used, in lieu of the original and much larger dataset, to train networks. Therefore, when provided with PCPs (which, by design are a compact version of the original dataset), researchers can train their networks with fewer computational resources. Lastly, PCPs can be used to retrieve similar and dissimilar patients from a clinical database. Such capabilities can be exploited by medical educators and physicians to retrieve patient data for illustration purposes and decision support, respectively.

LIMITATIONS. Despite the multi-purpose use of patient cardiac prototypes, they remain relatively myopic; they do not account for spatial and temporal changes in patient data. When learning PCPs, we implicitly assumed that all representations of cardiac signals belonging to the same patient should be attracted to a single PCP. This is a valid assumption if such patient-specific representations do indeed reflect a similar underlying physiological state. As we discussed with CLOCS, this assumption may not hold if patient data span multiple years or pertain to distinct modalities. In such a setting, patient cardiac prototypes may simply be a spatial and temporal average of a patient's health state. Although this summary embedding could be of use in certain scenarios, it would conceal subtle but useful changes in the patient's physiological state.

INTER-CONNECTEDNESS. Our work on PCPs shares similarities with that in Chapter 7 (CROCS). For example, with PCPs, our emphasis was on learning patient-specific embeddings via contrastive learning. Specifically, we encouraged representations belonging to the same patient to be similar to a single embedding that reflects that patient. With CROCS, we also learn embeddings via contrastive learning. However, each of these embeddings reflect a *set of patient attributes*. Moreover, the PCPs and the embeddings learned in the CROCS framework differ in their application. With the former, patient cardiac prototypes are used to generate patient-specific parameters, distil larger datasets, and discover similar and dissimilar patients. In contrast, the clinical prototypes in CROCS are used to cluster unlabelled cardiac signals, and retrieve relevant cardiac signals from a large database based on patient attributes. We believe that the representations of cardiac signals learned via the CROCS framework can also be exploited for the retrieval of patients with similar attributes. We did not explore this since CROCS was primarily designed to deal with *unlabelled* cardiac signals. Therefore, the retrieval of similar patients would not have provided practitioners with the attributes associated with each patient.

FUTURE WORK. In our work on personalized cardiac arrhythmia diagnosis, we learned patient cardiac prototypes that, we showed, were efficient descriptors of the *cardiac* state of the patient. Such an interpretation was driven by our use of multiple, large-scale, cardiac time-series datasets. Inspired by our findings, we believe that similar prototypes can be learned from other/multiple data modalities in order to obtain a more holistic summary of the state, cardiac or otherwise, of the patient. For example, given coronary angiogram data, electrocardiograms, and clinical reports, would it be possible to learn prototypes that further propel the diagnostic performance of clinical deep learning algorithms? Addressing this question could also lend insight into a future path, described earlier, pertaining to reliable patient similarity quantification.

Our work on patient cardiac prototypes provided a proof of concept for representations that can be exploited to retrieve similar and dissimilar patients both within and across different datasets. In other words, we validated the utility of PCPs as tools for patient-similarity quantification. A field of study that could benefit from such quantification is that of graph neural networks. Such networks typically require the design of an adjacency matrix, one that quantifies the presence and weight of edges between nodes. However, designing this matrix is non-trivial, particularly when dealing with physiological data, and can become a computational burden if the adjacency matrix is assumed to be dense. Instead, by interpreting these nodes as patients, our PCP-derived similarity values can be used to initialize the weights of edges between nodes and potentially inform the sparsity of node connections. As such, graph neural networks can be thought of as being trained with a prior of some sort.

Clinical Prototypes

In Chapter 7, we learned patient attribute-specific embeddings, entitled clinical prototypes, via a contrastive learning framework which we referred to as CROCS. We

ultimately used these clinical prototypes for the clustering and retrieval of unlabelled cardiac signals.

SIGNIFICANCE. Given that our clinical prototypes can be used for both clustering and retrieval, they confer two direct benefits. First, by clustering *unlabelled* cardiac signals in a supervised manner, researchers are now able to not only identify signals which are similar to one another, but also obtain their corresponding set of patient attributes. This process can be loosely thought of as a mechanism that annotates unlabelled instances. As such, it could prove valuable to those with access to abundant, unlabelled data from which they would like to extract clinical insight. Second, by retrieving relevant cardiac signals from a large clinical database, clinical prototypes can impact several stakeholders. For example, medical educators can now retrieve *unlabelled* instances that satisfy a set of criteria and use that to educate the next generation of medical students. Furthermore, researchers involved with clinical trials can now identify relevant patients to recruit that otherwise would have been excluded from consideration. As a result, patients may have improved access to clinical trials and the latter may begin to incorporate more comprehensive and diverse patient cohorts.

LIMITATIONS. Clinical prototypes, by design, were patient attribute-specific. Specifically, a clinical prototype was learned for each combination of patient attribute values. In our context, the number of such prototypes that had to be learned remained small (≈ 40). This was primarily because we chose a select few attributes (disease, sex, age) which we discretized. Incorporating more attributes and increasing the resolution of the discretization process would have resulted in significantly more clinical prototypes to learn. Such a setting poses two challenges to our existing CROCS framework. First, there is a possibility, although we do not have evidence in support of this, that clinical prototypes may exhibit some redundancy as their number grows. Second, as with natural language processing applications that exploit word embeddings, the growth of the number of clinical prototypes will increase the

computational cost of network training. This is because the embeddings are learned in an end-to-end manner and can thus be thought of as additional parameters of a network.

INTER-CONNECTEDNESS. Our CROCS framework, at a high-level, falls within the purview of contrastive learning methods. As such, it shares similarities with our work in Chapters 5 (CLOCS) and 6 (PCPs). In all such cases, we exploit the notion of attractions and repulsions and optimize a noise contrastive estimation loss. For example, with CLOCS, we encouraged representations belonging to the same patient to be similar to one another. With PCPs, we encouraged representations belonging to the same patient to be similar to a patient-specific embedding. With CROCS, we encouraged representations of instances associated with a set of patient attributes to be similar to an attribute-specific embedding.

FUTURE WORK. Although clinical prototypes encapsulate patient meta-information such as disease class, sex, and age, they are not fully interpretable in and of themselves. Yet their interpretability can improve their adoption and provide researchers with more control over the cardiac signals that they retrieve from large, unlabelled databases. One way to do so is by exploiting advancements in the field of disentangled representation learning. For example, clinical prototypes can be disentangled into their constituent attributes somewhat similar to what was proposed in β-VAEs (Higgins et al., 2016). Such disentangled prototypes, with more control over sensitive patient attributes, can also lend themselves to be applied in the field of fair machine learning.

When learning clinical prototypes, we made the design choice of discretizing patient attributes that are naturally continuous such as age. Although this choice simplified our framework and allowed us to experiment with a tractable approach, it suffers from several drawbacks. Many patient attributes can be continuous in nature and thus clinical prototypes may stand to benefit from incorporating such attribute values. We believe this would necessitate a sufficient amount of data from each of

these continuous attribute groups in order to learn meaningful prototypes but that is left for future work. Further research is also required to identify how to *scale* CPs to a continuous and large set of attributes, a process that will allow for even more fine-grained retrieval and further increase their utility.

Continual Learning

In Chapter 8, we proposed a continual learning framework, entitled CLOPS, that allows clinical deep learning networks to mitigate catastrophic forgetting. Such catastrophic forgetting is typically brought about by the presence of data that violate the assumption of being independent and identically distributed (i.i.d.).

SIGNIFICANCE. Our CLOPS framework explicitly dealt with scenarios in which data violated the i.i.d. assumption. Given the prevalence of such scenarios within healthcare, CLOPS confers significant benefits. First, researchers can now be equipped with a framework that achieves a clinical task (e.g., disease diagnosis) consistently well. This robustness increases the trustworthiness of networks and contributes to their clinical adoption. Second, by only having to replay instances from a buffer at periodic intervals, instead of re-training the network from scratch on a larger dataset, CLOPS can reduce the network training overhead. This observation suggests that networks can be trained more quickly and at a lower cost while minimally disrupting the clinical workflows in which these systems are deployed.

LIMITATIONS. Continual learning has primarily been achieved through three approaches; architecture-based approaches, regularization-based approaches, and replay-based approaches. Architecture-based approaches typically involve expanding the capacity of a network in response to new tasks. This can manifest in the form of, for example, additional classification heads for each task. Regularization-based approaches maintain the same capacity of the network across tasks however typically involve reducing the degree to which certain parameters are updated on subsequent tasks. Lastly, replay-based methods also maintain the same capacity of the network

across tasks yet involve a replay buffer in which instances are stored and replayed on subsequent tasks.

Replay-based methods can suffer from several limitations. First, their time and space complexity can grow in a linear fashion with the number of tasks that the network is exposed to. This is because additional instances are stored into the buffer with each additional task (spatial complexity). Moreover, depending on the replay mechanism (e.g., MC Dropout), additional time is now required to process the instances in the buffer (time complexity). As such, replay-based methods may experience significant slowdowns as the number of tasks grows. Second, replay-based methods assume that instances can be trivially stored in a buffer. This assumption, however, does not always hold. For example, within healthcare, storing raw patient data in such a manner can increase the risk of HIPAA violations and compromise patient privacy. Such a violation would be mitigated with architecture- and regularization-based approaches. Lastly, replay-based methods, due to their fixed network capacity, are more likely to experience interference in their parameters as a result of new tasks compared to architecture-based methods. This is because information pertinent for solving new tasks is forced to take residence in the existing parameters, which may hold information for solving previous tasks. This challenge may become even more pronounced as the number and diversity of tasks grows over time where we have more tasks competing for the same, scarce set of resources (parameters).

Benefits can also arise with the use of replay-based methods. For example, empirically, and based on recent literature, these methods have demonstrated superior performance relative to the remaining methods. This could be due to the explicit way in which catastrophic forgetting is mitigated in this setting. By storing instances in a replay buffer, this can be thought of as identifying a core-set of data points from each task which are sufficient for the learning of that task. Therefore, by replaying these instances during subsequent tasks, we are explicitly forcing the network to not forget how to solve previous tasks. Second, replay and regularization-based

methods are more elegant than architecture-based methods. This is because the latter can be thought of as "brute-force" methods that simply increase the capacity of the network over time. The computational burden of training and performing inference with such networks can quickly become intractable and beyond practical use. Such lack of practical applicability would be particularly pronounced in healthcare where decisions during inference might need to be made in a short period of time. Furthermore, this computational burden can place continual learning algorithms beyond the reach of researchers and practitioners in low-resource settings that do not have the infrastructure required to train and deploy them. As such, the accessibility of deep learning is reduced.

CLOPS is anchored around the notion of storing instances into, and acquiring instances from, a replay buffer. Since instances are stored into the buffer after each task, this buffer grows linearly with the number of tasks. Therefore, a buffer may require large computational resources (for storage and processing) and pose a risk to patient privacy (due to the storage of raw patient data). CLOPS also assumes that patient data can be stored temporarily in a buffer for use at a future date. However, this may not be possible due to patient privacy constraints. Furthermore, the buffer acquisition mechanism involves performing multiple forward passes (Monte Carlo Dropout, MCD) for each instance in the buffer. In light of these forward passes and the growth of the buffer with the number of tasks, buffer acquisition is guaranteed to become more costly with an increased number of tasks.

INTER-CONNECTEDNESS. Our framework's buffer acquisition mechanism was directly inspired by our work in Chapter 4. There, we deployed Monte Carlo Dropout alongside acquisition functions on *unlabelled* instances in order to quantify their informativeness and acquired them before requesting an oracle for their corresponding labels. With CLOPS, we are also deploying MCD with an acquisition function. However, in this context, we do so to quantify the informativeness of labelled instances in a replay buffer, before acquiring them for training purposes. Since they are already

labelled, an oracle is not required as part of the learning process. It is also worthwhile to note that the purpose of these acquisitions differed between the active and continual learning frameworks. The former was executed to accelerate the learning process of a network. The latter (CLOPS) was executed to mitigate the catastrophic forgetting of a network.

FUTURE WORK. In our work on continual learning, we learned task-instance parameters which, among other things, quantified the importance of instances for buffer storage and were probed for the quantification of task similarity via the Hellinger distance. To validate our interpretation of task-instance parameters, we leveraged this similarity to design multiple training curricula. However, the notion of task-similarity metrics dates back to Thrun and O'Sullivan (1996); Silver and Mercer (1996) with applications ranging from designing training curricula, to defining meta-learning tasks, and identifying optimal datasets for transfer learning. To advance the aforementioned fields when deployed for healthcare, further research can focus on designing similarity metrics that are specific to the healthcare task. Such research can also improve the interpretability of datasets, networks, and their interconnections.

CLOPS was primarily designed to overcome the phenomenon of destructive interference, where networks trained on tasks in the present forget how to solve tasks previously seen. To that end, we built upon the rich literature of replay-based methods which, we believe, can be thought of as dealing with destructive interference *reactively*. Such a reactive approach suggests that there exists a window of time, regardless of how brief, during which the performance of algorithms may suffer. High-stakes healthcare scenarios may not be able to afford such degradation in performance. Therefore, transitioning to a more *proactive* approach will reduce that lag-time, ensure algorithmic performance remains high, and that patient outcomes are not sacrificed in the process. For example, further research could exploit task-similarity and predict the likelihood of catastrophic forgetting occurring at some stage in the future, and explicitly account for that.

In Chapter 9, we designed a cardiac signal captioning framework that received, as input, a 12-lead ECG signal and returned, as output, an ECG report in multiple languages.

SIGNIFICANCE. Our captioning system confers several benefits to those in the healthcare community. First, by generating complete and accurate captions, this system reduces the amount of time and effort required by a cardiologist to manually generate these reports. This, in turn, allows them to refocus their attention on patient care and ultimately improve patient satisfaction. Second, generating a clinical report can provide cardiologists with information that supplements their decision making. As such, our system can operate as a decision support tool with regards to cardiac diagnoses. Third, the ability of our system to generate reports in multiple languages suggests that it can automatically cater to clinical settings at distinct geographical locations or to physicians that speak different languages. This can break down cultural barriers, improve communication, and streamline clinical workflows.

LIMITATIONS. In order to be able to generate ECG reports in multiple languages, we assumed access to paired ground-truth reports in such languages. In other words, an ECG report in English must also have been available in French and Spanish. This assumption, however, faces several limitations. First, these reports are unlikely to be available in various settings. This is because hospital settings are likely to operate in a single language and not more. In our context, we translated reports originally in English and German into other languages in order to form our ground-truth reports. Such an approach implies that errors in the translation system are likely to propagate through to the captioning system.

On another note, textual reports within healthcare can be quite heterogeneous. They can vary in length and complexity, and pertain to different divisions with a clinical setting. For example, a radiology report can be used to describe an MRI or CT scan taken of a patient's lungs. Such a report can entail generic patient details,

specific diagnoses, and co-morbidities that the patient is suffering from. These details are much more involved than those found in relatively straightforward ECG reports, and may thus be more difficult to generate via a neural network. Our captioning system was limited to generating these ECG reports and may thus not perform as well on more complex clinical reports.

INTER-CONNECTEDNESS. When designing the captioning system, we proposed a discriminative multi-lingual pre-training framework entitled RTLP. This notion of warm-starting a network such that it learns faster and achieves stronger generalization performance was also considered in our work in Chapter 5 (CLOCS). There, we exploited temporal and spatial invariances in the ECG data alongside the noise contrastive estimation loss to learn rich representations of cardiac signals. In contrast, with captioning, we proposed a token language prediction task (RTLP) as a way to learn rich representations of words in ECG reports in different languages. With CLOCS, we showed that pre-training can allow a network to learn faster (with fewer epochs) and with less dependence on labelled data. With captioning, we showed that networks pre-trained with RTLP perform on par with those pre-trained with more computationally expensive, generative frameworks such as MLM.

FUTURE WORK. Although ECG reports provide a succinct summary of the pathology of the heart, they do, arguably, contain fewer words and exhibit more homogeneity compared to, for example, reports of coronary angiograms or cardiac MRI. The diversity of text that appears in the latter may pose a greater challenge for our captioning network. As such, further research can focus on experimenting with clinical reports that are more complex, exhibit greater diversity, and even cover a more extensive list of languages. If proven successful, our framework has the potential to impact a multitude of clinical scenarios that involve the manual write-up of reports.

Our captioning framework was focused on mapping a single cardiac data modality, namely the ECG, to clinical reports in a multitude of languages. Despite having shown the relative accuracy and plausibility of the reports generated by our framework, we

believe that it stands to benefit from the incorporation of multiple data modalities. For example, in a hospital setting, ECG data are rarely collected independently of other modalities such as coronary angiograms, PPG, and so forth. Further research can focus on designing a multi-stream encoder that extracts pertinent features from multiple modalities before feeding them into a decoder for captioning. Not only is this approach more relatable to existing workflows in clinical settings, but it may also prove vital in generating holistic and more complex reports.

10.2 Discussion of broader limitations of clinical deep learning

Causal Inference, Interpretability, and Uncertainty Quantification

Causal inference is a framework which can involve controlling for confounding factors and identifying causal relationship between a pair of variables X and Y. Such a framework (Pearl, 1998) can offer researchers several benefits such as improving the interpretability of network-based predictions and accommodating for potential distribution shifts in the data (Subbaswamy and Saria, 2020).

As it pertains to causal inference and network interpretability, variables that are identified as being causally related can be compared to the *known* underlying relationship between such variables (e.g., via domain expertise). If there is alignment here, then researchers can become more confident in the associated causal diagram. In some cases, causal inference can even allow researchers to discover *unknown* causal relationships between health factors and patient outcomes, a process referred to as causal discovery (Glymour et al., 2019; Runge et al., 2019). Such findings can better guide future clinical research and policy interventions. Furthermore, when operating on the highest rung of the 'ladder of causality' (Pearl, 2009), researchers are interested in the notion of counterfactuals. This involves identifying the outcome of alternative decisions had they been taken instead of the actual decision. The ability to identify counterfactual scenarios allows researchers to better understand the underlying mechanisms behind a disease or pharmaceutical agent, and thus

contribute to better clinical decision-making (Schulam and Saria, 2017). Preliminary research in this direction has shown promising signs (Prosperi et al., 2020).

More broadly, deep learning systems are notorious for their opaque decision-making capabilities. In other words, it is difficult for practitioners to identify *why* a model made a certain decision. Although recent research has attempted to alleviate this concern through the use of saliency methods, these remain brittle (Adebayo et al., 2018) and thus unconvincing to downstream stakeholders such as physicians (Saporta et al., 2021). As a result of this, many defer to deploying traditional machine learning models that offer a higher level of interpretability. These can include logistic regression where the learned parameters have an intuitive meaning or even a random forest which returns the importance of features used in making predictions.

Based on my experience at several healthcare organizations, the level of interpretability that is required for a system depends on the workflow in which it will be deployed and how it will be used. For example, a system deployed in a critical workflow which involve 'make-or-break' decisions may have a higher threshold for interpretability. Conversely, those deployed for screening purposes which are overseen by medical practitioners may have a lower threshold for interpretability. Tangential but similar to the concept of interpretability is uncertainty quantification. While point estimates for predictions are a great first step, they ideally should be complemented by a quantification of the uncertainty of such predictions (Begoli et al., 2019). For example, in the case of regression, predicting a value of 50 for an outcome bounded between 0 (not severe condition) and 100 (severe condition) is less actionable than a value of 50 with a variance of 50. The former would suggest a patient has a mild condition whereas the latter suggests that there is not enough confidence in the prediction for a decision to be made. To incorporate such uncertainty quantification into systems, Bayesian neural networks and variants thereof (e.g., MC Dropout) have been exploited (Bate et al., 1998). It is also vital that such uncertainty quantification is reliable. In other words, is the uncertainty exhibited by the model a reflection of

the true uncertainty (Ovadia et al., 2019)? If, for example, the model exhibits low uncertainty for a prediction, where in fact it should have exhibited high uncertainty, then this would cause users to confidently depend on such predictions when they should not. Such dependence could have a detrimental impact on patient outcomes.

Causal inference can also play a role in overcoming the challenges associated with distribution shift (Quionero-Candela et al., 2009; Castro et al., 2020). For example, researchers can pro-actively design stable prediction models which are less dependent on unreliable paths in causal diagrams (Subbaswamy et al., 2019). Such unreliable paths are those which are unlikely to hold across distinct scenarios (e.g., hospitals). To achieve this, insight from the second rung on the 'ladder of causality' is exploited to surgically modify causal diagrams. While promising, this approach assumes that researchers are aware of the unreliable paths in causal diagrams, which can be non-trivial to isolate.

Adversarial Examples, Shortcut Learning, and Fairness

Deep learning networks are known to be quite brittle in a multitude of scenarios. For example, adversarial examples can easily fool networks into thinking that they belong to a distinct class (Han et al., 2020). Recent work has demonstrated that a visually imperceptible amount of noise that is added to an instance during inference can significantly confuse the network and cause it to output a different prediction. Moreover, the phenomenon of shortcut learning is pervasive amongst deep neural networks (Geirhos et al., 2020). In this setting, networks learn the spurious correlations that are in the data when solving a particular task. These spurious correlations, which do not reflect the true relationship between the inputs and the outputs, contribute to the unreliability of network predictions 'in the wild'. For example, the background of medical images could be correlated with a particular medical outcome, and, as such, the model latches onto this unreliable feature for outcome prediction (Jabbour et al., 2020). To more formally address this challenge, researchers released the 'Wilds' database (Koh et al., 2021), which comprises ten image datasets across distinct

domains. A similar database within the field of healthcare would also add significant value.

On a similar note, researchers have demonstrated the lack of reliability of clinical deep learning systems for certain sex groups and ethnicities (Chen et al., 2021). For example, a recent deep learning system systematically assigned Black patients a lower risk score than White patients even though the former exhibited a larger number of chronic conditions (Obermeyer et al., 2019). Such a finding, in addition to many others, emphasizes the importance of error analysis; conducting a thorough analysis of the errors committed by the model. This ultimately allows researchers to better understand potential biases that either the model has learned or that exist in the dataset used for training. Doing so then allows for more targeted solutions to the problem at hand. One potential solution is to limit the scope of application of the deep learning system to a sub-population for whom the model performs well. Another solution consists of improving the curation of clinical datasets and ground-truth labels (Willemink et al., 2020). Far too often, researchers, including myself, are focused on the design of algorithms for achieving particular tasks. However, as was recently pointed out as part of the push for MLOps (Ng, 2021), greater emphasis needs to be placed on the curation and quality of clinical datasets. The way in which such curation takes place could help quantify and mitigate the degree of bias inherent in clinical datasets (Mehrabi et al., 2021).

Generalizability, Dataset Availability, and Symbiosis

One of the core challenges of machine learning is, arguably, the notion of generalizability (Finlayson et al., 2020); can models perform their assigned task well on unseen instances from populations that are distinct geographically and demographically? One way to learn such a model would be to expose it to clinical data from as diverse a population as possible. However, researchers are often limited to the clinical datasets that reside within a handful of healthcare institutions in a single geographical location. Such limitations can arise due to stringent patient privacy regulations (e.g.,

HIPAA), bureaucratic policies, and the desire to maintain a competitive advantage over others. As such, most researchers within healthcare are locked out of access to clinical datasets that are large and contain a diverse population.

While I understand the importance of patient privacy and the way in which patient information can be abused if in the wrong hands, I also believe privacy, more broadly, is an (a) unnecessary modern invention and (b) an illusion in the 21st century. Without delving into the details, my perspective is that the potential benefits of a unified and shared clinical database (e.g., similar to databases periodically released by NASA) will outweigh the potential risks of having one. Seeing as this future is far off, other options do exist in the meantime. For example, federated learning (Li et al., 2020b) can allow systems to learn from clinical datasets located at distinct medical centres (Brisimi et al., 2018). This is achieved by inverting the standard approach to machine learning whereby a single model is trained on a single unified dataset. Instead, a model sends its parameters to distinct medical centres where training is conducted locally. Updated parameters are then sent back from each of these medical centres to a unified server where they are aggregated. Although federated learning can suffer from instabilities during the learning process, additional research can promote its technical feasibility.

Beyond federated learning, other methods have recently been proposed to design models that generalize well. These include the framework of invariant risk minimization (Arjovsky et al., 2019) in which an optimal classifier, when dealing with representations of data across distinct distributions, remains the same. Moreover, the framework of distributionally-robust machine learning focuses on optimizing a 'worst-case scenario' loss function (Sagawa et al., 2019) which attempts to model variations in the data distribution or reweights the per-instance loss during training via an importance-weighting parameter (Byrd and Lipton, 2019).

Despite the aforementioned limitations, I believe that clinical deep learning systems do have a role to play in the handful of settings characterized by regular, well-defined

tasks. However, many clinical workflows typically comprise multiple tasks that take place in unison or serially. Herein lies the potential of *modular* clinical deep learning systems that occupy a particular sub-task in this entire workflow. The remaining tasks can then be achieved either by more traditional machine learning models that lend themselves to a higher degree of interpretability or by physicians in the loop (Bansal et al., 2021). As such, I envision a symbiotic relationship between clinical deep learning systems and the communities they are meant to serve.

***

The motivating factor underlying the design of clinical deep learning algorithms is their potential tangible impact on healthcare. This can be in the form of a decision support system that assists with diagnoses, an early warning score that predicts the deterioration of hospitalized patients, or even a simple drug recommendation system. Regardless of the application, these systems require and benefit from the involvement of a medical professional from idea inception to system deployment. This ensures that we are asking the appropriate research questions and addressing them in an inclusive manner such that we increase the likelihood of adoption of our designed systems. For idea inception, domain knowledge can allow researchers to better exploit invariances in the clinical data, attend to pertinent regions of the input, and design network architectures that mimic biological systems, all of which can facilitate the extraction of more useful representations. As for deployment, it is integral to minimize the potential disruption of existing clinical workflows by the system, consider its human-computer interface and how stakeholders, such as physicians and nurses, will interact with such a system, and employ safeguards that will protect both hospitals from a legal perspective and patients from a safety perspective (Gerke et al., 2020). Ultimately, narrowing the 'adoption gap', as was done in Connell et al. (2019); Oktay et al. (2020), is crucial to improve patient outcomes. We hope that our proposed frameworks contribute to this ambitious endeavour.

***

Equipped with clinical deep learning algorithms capable of doing 'more with less'; less data, fewer annotations, and less medical supervision, we can contribute to the mission of improving the accessibility of healthcare to vulnerable patients in low-resource settings.

Part IV

APPENDICES

A

DOING SOME WITH LESS

A.1 GENERATIVE ADVERSARIAL NETWORKS FOR DISEASE SEVERITY DIAGNOSIS

A.1.1 *Additional Results*

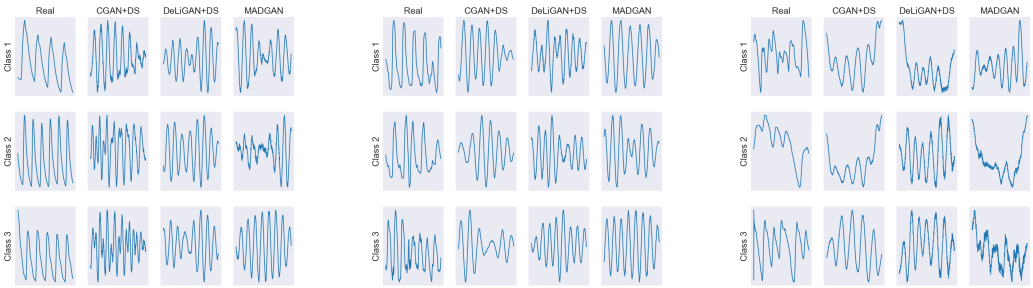
Comparison of Real Data to Synthetic Data

In this section, we compare the real PPG data to its synthetic counterpart in several ways. First, we qualitatively compare the signals themselves in Fig. A.1. Second, we perform non-linear dimensionality reduction on the real and synthetic PPG signals using t-distributed Stochastic Neighbour Embedding (t-SNE) (Maaten and Hinton, 2008). The results are shown, in Figs. A.2-A.4, for each of the four datasets (HFM, Tetanus, CVD, and Physionet) and each of the GANs implemented (CGAN+DS, DeLIGAN+DS, and MADGAN). The three colors represent the three different classes for each dataset.

In the t-SNE plots, the presence of relatively dense regions that contain synthetic datapoints (as in Fig. A.2) could be indicative of mode-collapse. This is a common challenge posed by GANs where trained generators produce outputs that are similar to one another and that do not span the entire distribution of the real data.

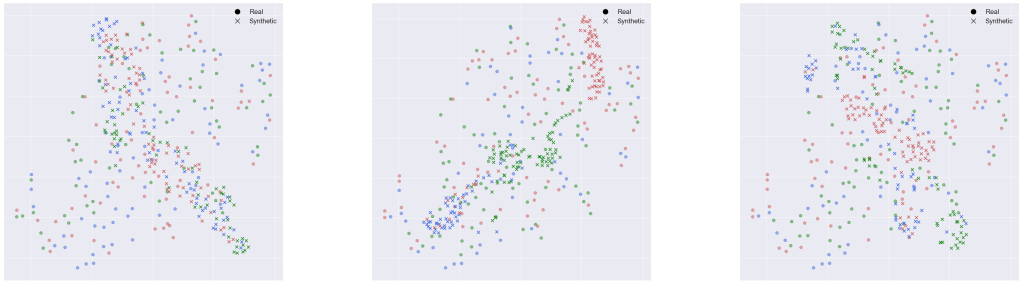
Intraclass Similarity Matrices

In this section, we illustrate the intraclass similarity kernel matrices for the datasets not presented in Chapter 3. We randomly sampled 30 synthetic datapoints generated by the three proposed conditional GANs and calculated their similarity to other



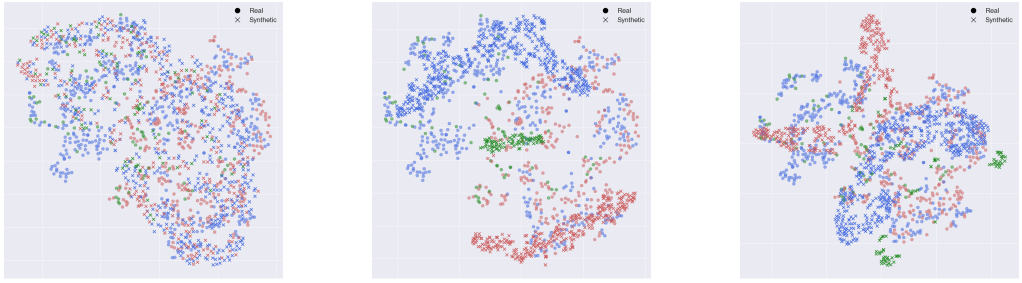
(a) Tetanus (b) CVD (c) Physionet

Figure A.1: **Randomly sampled class-specific real and synthetic PPG data generated by each of the CGAN models.** Samples are 5 seconds in duration. Note the ability of the CGANs to capture respiratory sinus arrythmia-induced amplitude modulation.



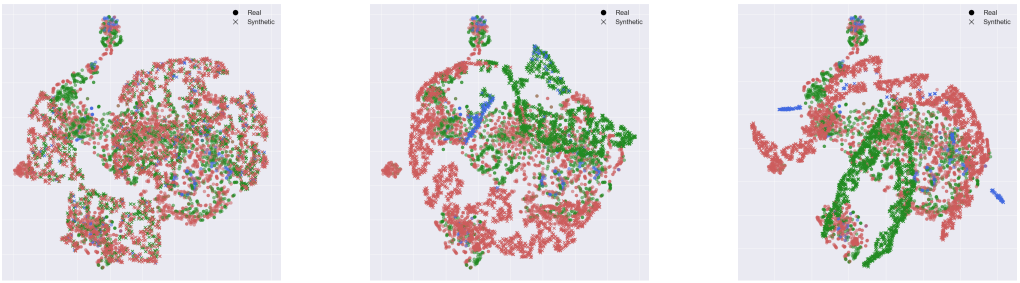
(a) CGAN+DS (b) DeLiGAN+DS (c) MADGAN

Figure A.2: **t-SNE embedding of real data from the HFM dataset and synthetic data generated by (a) CGAN+DS, (b) DeLiGAN+DS, and (c) MADGAN.** We see that the real and synthetic data cover a similar space and exhibit an adequate degree of diversity.



(a) CGAN+DS (b) DeLiGAN+DS (c) MADGAN

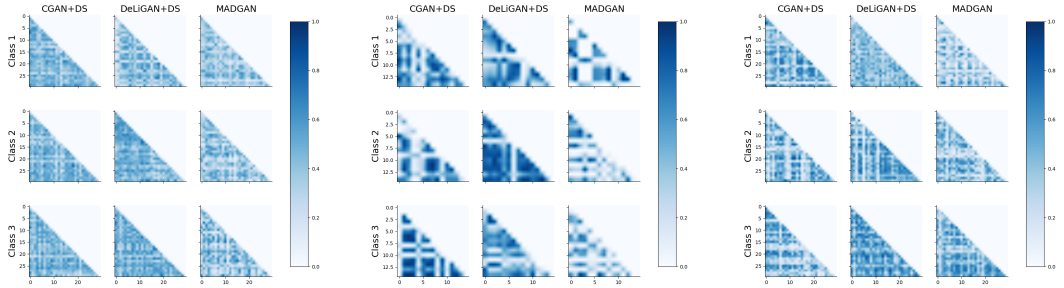
Figure A.3: **t-SNE embedding of real data from the Tetanus dataset and synthetic data generated by (a) CGAN+DS, (b) DeLiGAN+DS, and (c) MADGAN.** We see that the real and synthetic data cover a similar space and exhibit an adequate degree of diversity.



(a) CGAN+DS (b) DeLiGAN+DS (c) MADGAN

Figure A.4: **t-SNE embedding of real data from the Physionet dataset and synthetic data generated by (a) CGAN+DS, (b) DeLiGAN+DS, and (c) MADGAN.** We see that the real and synthetic data cover a similar space and exhibit an adequate degree of diversity.

synthetic datapoints from the same class. High similarity values indicate low diversity and likely point to mode-collapse at the class level.

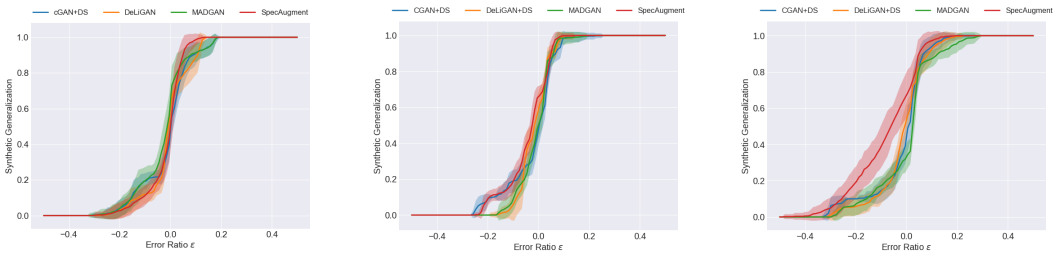


(a) Tetanus (b) CVD (c) Physionet

Figure A.5: **Exponentiated quadratic kernel matrices reflecting the intraclass similarity of synthetic datapoints from each cGAN.** We show the results for 30 randomly sampled datapoints generated by the three different cGANs (columns) for each of the three classes. Results are only shown for one seed.

Synthetic Generalization Curves

In this section, we illustrate the synthetic generalization curves (Fig A.6) for the datasets not presented in Chapter 3. When compared along this dimension, the proposed data augmentation methods do not differ significantly from one another. We quantify the average area under the synthetic generalization curve (AUSGC) for each of these curves in Table 3.4 of Chapter 3.



(a) HFM (b) Tetanus (c) Physionet

Figure A.6: **Synthetic generalization curve averaged across all 54 augmentation policies for each data augmentation method.** Curves are shown when evaluation is performed on the test set of (a) HFM, (b) Tetanus, and (c) Physionet. The shaded area represents one standard deviation from the mean.

B

DOING MORE WITH LESS

B.1 ACTIVE LEARNING FOR CARDIAC ARRHYTHMIA DIAGNOSIS

B.1.1 *Acquisition Functions*

$$\begin{aligned}\text{Variance Ratio} &= 1 - \frac{1}{T}\sum_{t=1}^{T}\left(\delta\left(\underset{c}{\text{argmax}}\, p(y=c|\boldsymbol{x},\boldsymbol{\omega_t}) = \hat{c}\right)\right) \\ \hat{c} &= \underset{c}{\text{argmax}}\left(\underset{c}{\text{argmax}}\, p(y=c|\boldsymbol{x},\boldsymbol{\omega_t})\ \forall t \in (1,T)\right)\end{aligned} \tag{B.1}$$

where $\hat{c}$ is the most common class prediction across the T MC samples and δ is the Dirac delta function that evaluates to 1 if its argument is true, and 0 otherwise.

$$\text{Entropy}, \mathbb{H} = -\sum_{c=1}^{C} p(y=c|\boldsymbol{x}) \log p(y=c|\boldsymbol{x}) \tag{B.2}$$

$$\begin{aligned}\text{BALD} &= \text{JSD}(p_1, p_2, \ldots, p_T) \\ &= \text{H}(p(y|\boldsymbol{x})) - \mathbb{E}_{p(\boldsymbol{w}|D_{train})}\left[\text{H}(p(y|\boldsymbol{x},\boldsymbol{w}))\right]\end{aligned} \tag{B.3}$$

where C is the number of classes in the task formulation and $p(y=c|\boldsymbol{x},\boldsymbol{\omega})$ is the probability assigned by a network parameterised by $\boldsymbol{\omega}$ to a particular class c when given an input $\boldsymbol{x}$.

B.1.2 *Implementation Details*

In this section, we outline the hyperparameters and network architecture used for all experiments conducted in Chapter 4. We also outline the batchsize and learning rate associated with training on each of the datasets.

Hyperparameters

For all experiments, we chose the number of MC samples $T = 20$ to balance between computational complexity and accuracy of the approximation of the version space. We acquire unlabelled instances at pre-defined epochs during training which we refer to as acquisition epochs, $\tau = 5n$, $n \in \mathbb{N}^+$. During each acquisition epoch, we acquire $b = 2\%$ of the remaining unlabelled instances. We also investigate the effect of such hyperparameters on performance. When experimenting with tracked acquisition functions, we chose the temporal period, $\Delta t = 1$.

Selective Oracle Questioning. Recall that we delegate selective oracle questioning to the network only when $\mathcal{D}_H \geq S$. Given $\mathcal{D}_H$'s increasing trend during training (see Fig. 4.5c), we chose $S = 0.15$ to balance between the reliability of the proxy and the independence of the network from an oracle. We also explore the sensitivity of SoQal to this choice of S.

Network Architecture and Perturbation Details

When conducting the MCP and BALC experiments, we perturbed each of the time-series frames with additive Gaussian noise, $\epsilon \sim \mathcal{N}(0, \sigma)$ where we chose σ based on the specific dataset to avoid introducing too much noise. The details of these perturbations can be found in Table B.2. We applied all perturbations to the input data *before* normalization.

Table B.1: **Network architecture used for experiments.** K, C_{in}, and C_{out} represent the kernel size, number of input channels, and number of output channels, respectively. A stride of 3 was used for Conv1D operators.

Layer Number	Layer Components	Kernel Dimension
1	Conv 1D BatchNorm ReLU MaxPool(2) Dropout(0.1)	$7 \times 1 \times 4$ ($K \times C_{\text{in}} \times C_{\text{out}}$)
2	Conv 1D BatchNorm ReLU MaxPool(2) Dropout(0.1)	$7 \times 4 \times 16$
3	Conv 1D BatchNorm ReLU MaxPool(2) Dropout(0.1)	$7 \times 16 \times 32$
4	Linear ReLU	320×100
5	Linear	$100 \times C$ (classes)

Table B.2: **Batchsize, learning rates, and perturbations used for training with different datasets.** The Adam optimizer was used for all experiments.

Dataset	Batchsize	Learning Rate	Perturbation
$\mathcal{D}_1$	256	10^{-4}	$\epsilon \sim \mathcal{N}(0, 100)$
$\mathcal{D}_2$	256	10^{-4}	$\epsilon \sim \mathcal{N}(0, 100)$
$\mathcal{D}_3$	256	10^{-4}	$\epsilon \sim \mathcal{N}(0, 100)$
$\mathcal{D}_4$	16	10^{-4}	$\epsilon \sim \mathcal{N}(0, 100)$

Baseline Implementations

In this section, we outline our implementation of the baseline methods used in the selective oracle questioning experiments.

Entropy response. This approach is anchored around the idea that network outputs that exhibit high entropy (i.e., close to a uniform distribution) are likely to correspond to instances that the network is uncertain of. Consequently, we exploited this idea to determine whether a label is requested from an oracle or if a pseudo-label should be generated instead. More specifically, we introduced a threshold, $\mathrm{S}_{Entropy} = w \times \mathrm{S}_{Max}$, which is a fraction of the maximum entropy possible for a

particular classification problem. As mentioned, $S_{Max} = \log C$, where C is the number of classes. We chose $w = 0.9$ to balance between oracle dependence and pseudo-label accuracy. This value was kept fixed during training. In our implementation, we take the mean of the network outputs as a result of the perturbations, calculate its entropy, and determine whether it exceeds the aforementioned threshold. If it does, then the uncertainty is deemed high and a label is requested from an oracle.

Epsilon greedy. This approach is inspired by the reinforcement learning literature and is used to decay the dependence of network on the oracle. More specifically, we define $\epsilon = e^{\frac{-\text{epoch}}{k \times \tau}}$ where epoch represents the training epoch number and τ is the epoch interval at which acquisitions are performed. ϵ decays from $1 \rightarrow 0$ as training progresses. We chose $k = \tau = 5$ in order to balance between oracle dependence and pseudo-label accuracy. To determine whether a label is requested from an oracle, we generate a random number, $R \sim \mathcal{U}(0, 1)$, from a uniform distribution and check whether it is below ϵ. If this is satisfied, then an oracle is requested for a label, and a pseudo-label is generated otherwise. As designed, this approach starts off with 100% dependence on an oracle and decays towards minimal dependence as training progresses.

B.1.3 *Additional Results*

Test Set Performance in the Absence of Oracle

In this section, we quantify and compare the performance of our consistency-based active learning framework to state-of-the-art AL methods on four diverse datasets, $\mathcal{D}_1$ - $\mathcal{D}_4$ and for a range of fraction values, $\beta = (0.1, 0.3, 0.5, 0.7, 0.9)$. Across Tables B.3a - B.3d, we show that our method outperforms the baseline methods in 17 out of 28 (61%) experimental categories. Moreover, in half of all experimental categories, temporal acquisition functions perform best.

Table B.3: **Evaluation of acquisition functions on the test sets of $\mathcal{D}_1$ - $\mathcal{D}_4$ as the fraction of available labelled training data is varied $\beta = (0.1, 0.3, 0.5, 0.7, 0.9)$.** Bolded elements represent the best performing method and acquisition function α for each fraction β. No AL represents training without an active learning strategy. Results shown across 5 seeds.

(a) PhysioNet 2015 PPG, $\mathcal{D}_1$

Fraction β	Method	Acquisition Metric								No AL
		Non-temporal				Temporal				
		Var Ratio	Entropy	BALD	-	Var Ratio	Entropy	BALD	-	
0.1	MCD	0.476 ± 0.022	0.475 ± 0.020	0.465 ± 0.017	-	0.468 ± 0.032	0.492 ± 0.022	0.476 ± 0.015	-	0.577 ± 0.014
	MCP	0.475 ± 0.037	0.448 ± 0.019	0.464 ± 0.023	-	0.497 ± 0.028	0.490 ± 0.032	**0.515 ± 0.025**	-	
	$BALC_{JSD}$	-	-	-	0.511± 0.031	-	-	-	0.494 ± 0.021	
	$BALC_{KLD}$	-	-	-	0.500 ± 0.023	-	-	-	0.496 ± 0.024	
0.3	MCD	0.603 ± 0.021	0.618 ± 0.026	0.606 ± 0.032	-	0.607 ± 0.012	0.614 ± 0.009	0.617± 0.035	-	0.653 ± 0.017
	MCP	0.633 ± 0.024	0.607 ± 0.015	0.598 ± 0.015	-	0.626 ± 0.032	0.627 ± 0.026	0.606 ± 0.031	-	
	$BALC_{JSD}$	-	-	-	0.594 ± 0.009	-	-	-	0.600 ± 0.016	
	$BALC_{KLD}$	-	-	-	**0.633± 0.017**	-	-	-	0.617 ± 0.019	
0.5	MCD	0.650 ± 0.011	0.650 ± 0.011	0.660 ± 0.013	-	0.654 ± 0.014	0.653 ± 0.024	0.655 ± 0.008	-	0.665 ± 0.007
	MCP	0.655 ± 0.013	0.653 ± 0.008	0.647 ± 0.019	-	0.642 ± 0.027	**0.669 ± 0.016**	0.648 ± 0.012	-	
	$BALC_{JSD}$	-	-	-	0.658 ± 0.003	-	-	-	0.652 ± 0.025	
	$BALC_{KLD}$	-	-	-	0.662± 0.015	-	-	-	0.661 ± 0.014	
0.7	MCD	0.650 ± 0.008	0.640 ± 0.008	**0.658 ± 0.010**	-	0.656 ± 0.008	0.636 ± 0.0134	0.655 ± 0.007	-	0.642 ± 0.015
	MCP	0.653 ± 0.010	0.652 ± 0.009	0.654 ± 0.008	0.646 ± 0.015	-	0.649 ± 0.006	0.651 ± 0.010	-	
	$BALC_{JSD}$	-	-	-	0.642 ± 0.011	-	-	-	0.649 ± 0.008	
	$BALC_{KLD}$	-	-	-	0.653 ± 0.010	-	-	-	0.656 ± 0.012	
0.9	MCD	**0.704 ± 0.009**	0.691 ± 0.012	0.698 ± 0.018	-	0.690 ± 0.022	0.693 ± 0.015	0.692 ± 0.020	-	0.680 ± 0.039
	MCP	0.702 ± 0.019	0.678 ± 0.020	0.700 ± 0.015	-	0.703 ± 0.016	0.680 ± 0.017	0.692 ± 0.009	-	
	$BALC_{JSD}$	-	-	-	0.690 ± 0.006	-	-	-	0.700 ± 0.028	
	$BALC_{KLD}$	-	-	-	0.699 ± 0.011	-	-	-	0.689 ± 0.013	

(b) PhysioNet 2015 ECG, $\mathcal{D}_2$

Fraction β	Method	Acquisition Metric								No AL
		Non-temporal				Temporal				
		Var Ratio	Entropy	BALD	-	Var Ratio	Entropy	BALD	-	
0.1	MCD	0.567 ± 0.029	0.591 ± 0.040	0.573 ± 0.063	-	0.547 ± 0.058	0.584 ± 0.055	0.598 ± 0.050	-	0.679 ± 0.040
	MCP	0.567 ± 0.027	0.557 ± 0.032	0.589 ± 0.045	-	0.548 ± 0.036	0.549 ± 0.046	0.554 ± 0.055	-	
	$BALC_{JSD}$	-	-	-	0.576 ± 0.050	-	-	-	0.574 ± 0.057	
	$BALC_{KLD}$	-	-	-	**0.602 ± 0.044**	-	-	-	0.575 ± 0.017	
0.3	MCD	0.675 ± 0.022	0.666 ± 0.053	0.643 ± 0.036	-	0.644 ± 0.019	**0.692 ± 0.020**	0.684 ± 0.035	-	0.605 ± 0.020
	MCP	0.678 ± 0.036	0.660 ± 0.071	0.665 ± 0.051	-	0.643 ± 0.038	0.668 ± 0.020	0.658 ± 0.026	-	
	$BALC_{JSD}$	-	-	-	0.654 ± 0.033	-	-	-	0.677 ± 0.032	
	$BALC_{KLD}$	-	-	-	0.634 ± 0.032	-	-	-	0.672 ± 0.049	
0.5	MCD	0.676 ± 0.0434	0.700 ± 0.031	0.668 ± 0.0185	-	0.709 ± 0.0407	0.694 ± 0.0431	0.669 ± 0.0238	-	0.703 ± 0.032
	MCP	0.687 ± 0.0183	0.695 ± 0.0212	0.712 ± 0.0235	-	0.700 ± 0.0135	0.709 ± 0.0261	0.680 ± 0.0247	-	
	$BALC_{JSD}$	-	-	-	0.701 ± 0.026	-	-	-	0.703 ± 0.018	
	$BALC_{KLD}$	-	-	-	0.705 ± 0.045	-	-	-	**0.726 ± 0.031**	
0.7	MCD	0.758 ± 0.016	0.765 ± 0.027	0.754 ± 0.014	-	0.753 ± 0.020	0.766 ± 0.025	0.755 ± 0.024	-	0.747 ± 0.010
	MCP	0.744 ± 0.031	0.759 ± 0.022	0.745 ± 0.027	-	0.757 ± 0.013	**0.777 ± 0.025**	0.764 ± 0.014	-	
	$BALC_{JSD}$	-	-	-	0.750 ± 0.006	-	-	-	0.746 ± 0.016	
	$BALC_{KLD}$	-	-	-	0.730 ± 0.035	-	-	-	0.761 ± 0.028	
0.9	MCD	0.742 ± 0.016	0.745 ± 0.048	0.757 ± 0.015	-	0.769 ± 0.0261	0.766 ± 0.018	0.754 ± 0.015	-	0.747 ± 0.011
	MCP	0.765 ± 0.013	0.759 ± 0.028	0.751 ± 0.013	-	0.758 ± 0.018	0.759 ± 0.021	0.743 ± 0.025	-	
	$BALC_{JSD}$	-	-	-	0.726 ± 0.008	-	-	-	**0.771 ± 0.018**	
	$BALC_{KLD}$	-	-	-	0.762 ± 0.037	-	-	-	0.749 ± 0.020	

(c) PhysioNet 2017 ECG, $\mathcal{D}_3$

Fraction β	Method	Acquisition Metric								No AL
		Non-temporal				Temporal				
		Var Ratio	Entropy	BALD	-	Var Ratio	Entropy	BALD	-	
0.1	MCD	0.628 ± 0.006	0.620 ± 0.006	0.581 ± 0.014	-	0.614 ± 0.03	0.610 ± 0.013	0.562 ± 0.019	-	0.716 ± 0.012
	MCP	0.624 ± 0.017	0.621 ± 0.018	0.623 ± 0.020	-	0.605 ± 0.027	0.613 ± 0.026	0.622 ± 0.026	-	
	$BALC_{JSD}$	-	-	-	0.613 ± 0.013	-	-	-	0.611 ± 0.015	
	$BALC_{KLD}$	-	-	-	**0.631 ± 0.010**	-	-	-	0.600 ± 0.005	
0.3	MCD	0.705 ± 0.003	0.672 ± 0.009	0.688 ± 0.011	-	0.704 ± 0.016	0.685 ± 0.010	0.684 ± 0.0081	-	0.766 ± 0.012
	MCP	0.688 ± 0.018	0.673 ± 0.007	**0.719 ± 0.016**	-	0.671 ± 0.016	0.684 ± 0.018	0.699 ± 0.023	-	
	$BALC_{JSD}$	-	-	-	0.694 ± 0.006	-	-	-	0.681 ± 0.010	
	$BALC_{KLD}$	-	-	-	0.703 ± 0.023	-	-	-	0.701 ± 0.015	
0.5	MCD	0.744 ± 0.013	0.735 ± 0.007	0.749 ± 0.012	-	**0.772 ± 0.015**	0.743 ± 0.018	0.758 ± 0.009	-	0.790 ± 0.012
	MCP	0.744 ± 0.008	0.733 ± 0.006	0.747 ± 0.004	-	0.741 ± 0.013	0.752 ± 0.019	0.732 ± 0.038	-	
	$BALC_{JSD}$	-	-	-	0.763 ± 0.022	-	-	-	0.771 ± 0.011	
	$BALC_{KLD}$	-	-	-	0.769 ± 0.006	-	-	-	0.761 ± 0.003	
0.7	MCD	0.802 ± 0.006	0.811 ± 0.007	0.809 ± 0.004	-	0.807 ± 0.010	0.807 ± 0.003	**0.815 ± 0.010**	-	0.810 ± 0.008
	MCP	0.786 ± 0.003	0.782 ± 0.011	0.784 ± 0.016	-	0.772 ± 0.014	0.765 ± 0.013	0.762 ± 0.018	-	
	$BALC_{JSD}$	-	-	-	0.803 ± 0.011	-	-	-	0.813 ± 0.010	
	$BALC_{KLD}$	-	-	-	0.809 ± 0.006	-	-	-	0.810 ± 0.005	
0.9	MCD	0.820 ± 0.006	0.824 ± 0.005	0.828 ± 0.004	-	0.821 ± 0.011	0.823 ± 0.005	0.825 ± 0.006	-	0.827 ± 0.004
	MCP	0.826 ± 0.002	0.821 ± 0.007	0.807 ± 0.011	-	0.828 ± 0.008	0.812 ± 0.009	0.808 ± 0.012	-	
	$BALC_{JSD}$	-	-	-	0.825 ± 0.003	-	-	-	0.824 ± 0.011	
	$BALC_{KLD}$	-	-	-	0.827 ± 0.005	-	-	-	**0.829 ± 0.007**	

(d) Cardiology ECG, $\mathcal{D}_4$

Fraction β	Method	Acquisition Metric								No AL
		Non-temporal				Temporal				
		Var Ratio	Entropy	BALD	-	Var Ratio	Entropy	BALD	-	
0.1	MCD	0.475 ± 0.039	**0.518 ± 0.016**	0.486 ± 0.011	-	0.485± 0.029	0.491 ± 0.022	0.484 ± 0.040	-	0.486 ± 0.023
	MCP	0.508 ± 0.031	0.492 ± 0.022	0.493 ± 0.030	-	0.500 ± 0.024	0.478 ± 0.024	0.492 ± 0.022	-	
	$BALC_{JSD}$	-	-	-	0.460 ± 0.043	-	-	-	0.487 ± 0.042	
	$BALC_{KLD}$	-	-	-	0.505 ± 0.032	-	-	-	0.511 ± 0.030	
0.3	MCD	0.487 ± 0.012	0.510 ± 0.018	0.498 ± 0.026	-	0.491 ± 0.014	0.496 ± 0.015	0.500 ± 0.025	-	0.533 ± 0.020
	MCP	0.520 ± 0.007	0.480 ± 0.019	0.494 ± 0.019	-	0.497 ± 0.007	**0.529 ± 0.035**	0.498 ± 0.021	-	
	$BALC_{JSD}$	-	-	-	0.488 ± 0.025	-	-	-	0.487 ± 0.016	
	$BALC_{KLD}$	-	-	-	0.510 ± 0.030	-	-	-	0.494 ± 0.014	
0.5	MCD	0.563 ± 0.021	**0.591 ± 0.008**	0.562 ± 0.011	-	0.557 ± 0.025	0.580 ± 0.006	0.569 ± 0.010	-	0.581 ± 0.019
	MCP	0.529 ± 0.027	0.554 ± 0.024	0.544 ± 0.015	-	0.557 ± 0.021	0.536 ± 0.013	0.526 ± 0.012	-	
	$BALC_{JSD}$	-	-	-	0.559 ± 0.001	-	-	-	0.559 ± 0.003	
	$BALC_{KLD}$	-	-	-	0.575 ± 0.028	-	-	-	0.576 ± 0.011	
0.7	MCD	0.637± 0.010	0.615 ± 0.010	0.639 ± 0.016	-	0.633 ± 0.016	0.652 ± 0.028	0.662 ± 0.014	-	0.630 ± 0.008
	MCP	0.626 ± 0.018	0.626 ± 0.013	0.623± 0.031	-	0.623 ± 0.012	0.623 ± 0.003	0.624 ± 0.010	-	
	$BALC_{JSD}$	-	-	-	0.634 ± 0.024	-	-	-	**0.648 ± 0.023**	
	$BALC_{KLD}$	-	-	-	0.625 ± 0.015	-	-	-	0.632 ± 0.028	
0.9	MCD	0.651 ± 0.008	0.666 ± 0.011	0.666 ± 0.017	-	0.670 ± 0.007	0.653 ± 0.025	**0.677 ± 0.009**	-	0.660 ± 0.013
	MCP	0.655 ± 0.027	0.673 ± 0.009	0.672 ± 0.017	-	0.663 ± 0.006	0.662± 0.005	0.670± 0.009	-	
	$BALC_{JSD}$	-	-	-	0.656 ± 0.015	-	-	-	0.656 ± 0.019	
	$BALC_{KLD}$	-	-	-	0.666 ± 0.025	-	-	-	0.663 ± 0.013	

B.2.1 *Implementation Details*

Hyperparameters

During self-supervised pre-training, we chose the temperature parameter, $\tau = 0.1$, as per Chen et al. (2020a). For BYOL, we chose the decay rate, $\tau_d = 0.90$, after experimenting with various alternatives (see Appendix B.2.2). For all experiments, we use a neural architecture composed of three 1D convolutional layers followed by two fully connected layers (shown next).

Network Architecture

In this section, we outline the architecture of the neural network used for all experiments. For pre-training, the final layer (Layer 5) was removed and representations with dimension E were learned. During training on the downstream tasks, the final layer was introduced.

Baseline Implementations

Supervised Pre-training. In this implementation, we pre-train on the specified dataset under the assumption that 100% of the data is labelled and available for training (i.e., $F = 1$). Given the presence of labels, pre-training involves solving a supervised classification task to diagnose the cardiac arrhythmia that corresponds to each ECG recording. In our context, supervised pre-training is expected to generate the best downstream generalization performance due to the availability of labels *and* the high similarity between the upstream and downstream tasks, namely cardiac arrhythmia classification.

MT-SSL. In this implementation, we introduce six different pre-text tasks that are used for pre-training a network. We follow the multi-task pre-training setup proposed by (Sarkar and Etemad, 2020) where six different classification heads are

Table B.4: **Network architecture used for all experiments.** K, C_{in}, and C_{out} represent the kernel size, number of input channels, and number of output channels, respectively. A stride of 3 was used for all convolutional layers. E represents the dimension of the final representation.

Layer Number	Layer Components	Kernel Dimension
1	Conv 1D BatchNorm ReLU MaxPool(2) Dropout(0.1)	$7 \times 1 \times 4$ ($K \times C_{\text{in}} \times C_{\text{out}}$)
2	Conv 1D BatchNorm ReLU MaxPool(2) Dropout(0.1)	$7 \times 4 \times 16$
3	Conv 1D BatchNorm ReLU MaxPool(2) Dropout(0.1)	$7 \times 16 \times 32$
4	Linear ReLU	$320 \times E$
5	Linear	$E \times C$ (classes)

Table B.5: **Batchsize and learning rates used for training with different datasets.** The Adam optimizer was used for all experiments.

Dataset	Batchsize	Learning Rate
PhysioNet 2020	256	10^{-4}
Chapman	256	10^{-4}
Cardiology	16	10^{-4}
PhysioNet 2017	256	10^{-4}

used to solve each of the six tasks. These tasks comprise binary classification where the network is asked to discriminate between ECG instances and their perturbed counterpart. Such perturbations take on the form of 1) Gaussian noise addition, 2) scaling, 3) negation, 4) temporal inversion, 5) permutation, and 6) time-warping. For the Chapman dataset, we only pre-train using scaling, negation, and temporal inversion since additional tasks prevented the network from converging. On the PhysioNet2020 dataset, however, we pre-train using all of the aforementioned tasks.

BYOL. In this implementation, an instance is perturbed by applying two stochastic transformations. In our setup, these transformations can include any of those outlined

in Sec. 5.4.4. This process results in two views of the same instance, each of which is passed through an online network and a target network. The target network is an exponential moving average of the online network, and is thus a delayed version of the online network. This delay is dictated by the decay rate, τ_d. We chose $\tau_d = 0.9$ based on experiments in Appendix B.2.2. A key difference between the two networks is that they are *asymmetric*, with the online network consisting of an additional prediction head. The goal is for the representation from the online network to predict that from the target network. This is done by minimizing the mean squared error of the two representations. In our setup, we introduce asymmetry by repeating Layer 4 shown in Table B.4. This is similar to that performed by Grill et al. (2020).

SimCLR. In this implementation, an instance is perturbed by applying two stochastic transformations. In our setup, these transformations can include any of those outlined in Sec. 5.4.4. This process results in two views of the same instance, each of which is passed through the same network. The InfoNCE loss is used to attract representations that are similar to one another and repel those that are different. Whether representations should be attracted to one another depends on whether they belong to the same original instance.

B.2.2 *Additional Results*

In this section, we show that in 21/24 (88%) of all experiments conducted, our family of contrastive learning methods outperforms the state-of-the-art method, SimCLR. This can be seen by the bold test AUC results in Table B.6.

Effect of τ_d on BYOL Implementation

In the BYOL implementation, two networks exist; an online network and a target network. The latter is a delayed version of the former where its parameters are an exponential moving average of the parameters of the online network. This exponential moving average is a function of the hyperparameter, τ_d. In this section, we outline the

Table B.6: **Comparison of self-supervised methods when used to initialize parameters before fine-tuning on downstream datasets.** Pre-training, fine-tuning, and evaluating multi-lead datasets* using 4 leads. Mean and standard deviation are shown across 5 seeds.

(a) $F = 0.25$

Pretraining Dataset	Chapman*			PhysioNet 2020*		
Downstream Dataset	Cardiology	PhysioNet 2017	PhysioNet 2020*	Cardiology	PhysioNet 2017	Chapman*
Random Init.	0.625 ± 0.015	0.746 ± 0.006	0.764 ± 0.016	0.625 ± 0.015	0.746 ± 0.006	0.894 ± 0.002
Supervised	0.671 ± 0.009	0.786 ± 0.012	0.804 ± 0.005	0.679 ± 0.011	0.805 ± 0.005	0.942 ± 0.011
Self-supervised Pre-training						
BYOL	0.620 ± 0.013	0.726 ± 0.013	0.764 ± 0.013	0.624 ± 0.021	0.752 ± 0.011	0.904 ± 0.006
SimCLR	0.634 ± 0.014	0.738 ± 0.006	0.777 ± 0.015	0.631 ± 0.022	0.727 ± 0.014	0.903 ± 0.007
CMSC	**0.691 ± 0.015**	**0.768 ± 0.005**	**0.813 ± 0.007**	**0.671 ± 0.018**	**0.756 ± 0.009**	**0.911 ± 0.016**
CMLC	0.639 ± 0.010	0.745 ± 0.012	0.770 ± 0.006	0.641 ± 0.014	0.746 ± 0.014	0.897 ± 0.003
CMSMLC	0.671 ± 0.016	0.755 ± 0.011	0.781 ± 0.012	0.668 ± 0.011	0.751 ± 0.007	0.903 ± 0.009

(b) $F = 0.5$

Pretraining Dataset	Chapman*			PhysioNet 2020*		
Downstream Dataset	Cardiology	PhysioNet 2017	PhysioNet 2020*	Cardiology	PhysioNet 2017	Chapman*
Random Init.	0.678 ± 0.011	0.763 ± 0.005	0.803 ± 0.008	0.678 ± 0.011	0.763 ± 0.005	0.907 ± 0.006
Supervised	0.684 ± 0.015	0.799 ± 0.008	0.827 ± 0.001	0.730 ± 0.002	0.810 ± 0.009	0.954 ± 0.003
Self-supervised Pre-training						
BYOL	0.678 ± 0.021	0.748 ± 0.014	0.802 ± 0.013	0.674 ± 0.022	0.757 ± 0.01	0.916 ± 0.009
SimCLR	0.676 ± 0.011	0.772 ± 0.010	0.823 ± 0.011	0.658 ± 0.027	0.762 ± 0.009	0.923 ± 0.010
CMSC	0.695 ± 0.024	0.773 ± 0.013	**0.830 ± 0.002**	**0.714 ± 0.014**	0.760 ± 0.013	**0.932 ± 0.008**
CMLC	0.665 ± 0.016	0.767 ± 0.013	0.810 ± 0.011	0.675 ± 0.013	0.762 ± 0.007	0.910 ± 0.012
CMSMLC	**0.717 ± 0.006**	**0.774 ± 0.004**	0.814 ± 0.009	0.698 ± 0.011	**0.774 ± 0.012**	0.930 ± 0.012

(c) $F = 0.75$

Pretraining Dataset	Chapman*			PhysioNet 2020*		
Downstream Dataset	Cardiology	PhysioNet 2017	PhysioNet 2020*	Cardiology	PhysioNet 2017	Chapman*
Random Init.	0.675 ± 0.020	0.775 ± 0.005	0.831 ± 0.011	0.675 ± 0.020	0.775 ± 0.005	0.937 ± 0.008
Supervised	0.712 ± 0.017	0.799 ± 0.014	0.837 ± 0.005	0.731 ± 0.007	0.815 ± 0.007	0.958 ± 0.004
Self-supervised Pre-training						
BYOL	0.671 ± 0.022	0.754 ± 0.009	0.825 ± 0.009	0.700 ± 0.02	0.751 ± 0.033	0.930 ± 0.005
SimCLR	0.694 ± 0.019	0.776 ± 0.013	0.834 ± 0.009	0.686 ± 0.019	**0.785 ± 0.011**	0.931 ± 0.013
CMSC	0.700 ± 0.012	**0.801 ± 0.013**	**0.840 ± 0.004**	0.707 ± 0.015	0.777 ± 0.016	**0.942 ± 0.012**
CMLC	0.670 ± 0.019	0.771 ± 0.010	0.831 ± 0.004	0.682 ± 0.005	0.772 ± 0.009	0.917 ± 0.011
CMSMLC	**0.719 ± 0.011**	0.792 ± 0.014	0.837 ± 0.008	**0.711 ± 0.011**	0.777 ± 0.017	0.938 ± 0.010

(d) $F = 1$

Pretraining Dataset	Chapman*			PhysioNet 2020*		
Downstream Dataset	Cardiology	PhysioNet 2017	PhysioNet 2020*	Cardiology	PhysioNet 2017	Chapman*
Random Init.	0.702 ± 0.016	0.773 ± 0.010	0.843 ± 0.002	0.702 ± 0.016	0.773 ± 0.010	0.930 ± 0.013
Supervised	0.712 ± 0.017	0.799 ± 0.011	0.844 ± 0.003	0.732 ± 0.008	0.821 ± 0.006	0.961 ± 0.004
Self-supervised Pre-training						
BYOL	0.697 ± 0.006	0.774 ± 0.017	0.834 ± 0.011	0.709 ± 0.017	0.771 ± 0.022	0.935 ± 0.008
SimCLR	0.705 ± 0.008	**0.810 ± 0.016**	0.844 ± 0.005	0.700 ± 0.012	**0.795 ± 0.021**	0.941 ± 0.006
CMSC	0.715 ± 0.018	0.804 ± 0.018	**0.846 ± 0.002**	**0.725 ± 0.020**	0.779 ± 0.024	0.942 ± 0.009
CMLC	0.698 ± 0.007	0.781 ± 0.014	0.836 ± 0.003	0.681 ± 0.005	0.785 ± 0.011	0.933 ± 0.014
CMSMLC	**0.732 ± 0.003**	0.793 ± 0.012	0.844 ± 0.005	0.716 ± 0.010	0.778 ± 0.025	**0.945 ± 0.005**

effect of τ_d on the downstream generalization performance of networks in the linear evaluation scenario. This can be found in Table B.7. We find that the results associated with $\tau_d = 0.900$ lead to the best performance, and are thus quoted in Chapter 5.

Table B.7: **Effect of the value of τ_d during BYOL pre-training on the downstream generalization performance of a linear evaluation scenario.** Pre-training, fine-tuning, and evaluating multi-lead datasets* using 4 leads. Mean and standard deviation are shown across 5 seeds. We show that $\tau_d = 0.9$ is most suitable for our applications.

(a) $F = 0.25$

Dataset	Chapman*	PhysioNet 2020*
$\tau_d = 0.500$	0.602 ± 0.072	0.581 ± 0.010
$\tau_d = 0.900$	**0.671 ± 0.042**	**0.587 ± 0.021**
$\tau_d = 0.990$	0.597 ± 0.068	0.571 ± 0.028

(b) $F = 0.5$

Dataset	Chapman*	PhysioNet 2020*
$\tau_d = 0.500$	0.618 ± 0.087	0.590 ± 0.010
$\tau_d = 0.900$	**0.643 ± 0.043**	**0.595 ± 0.018**
$\tau_d = 0.990$	0.604 ± 0.079	0.578 ± 0.033

(c) $F = 0.75$

Dataset	Chapman*	PhysioNet 2020*
$\tau_d = 0.500$	0.635 ± 0.075	0.597 ± 0.008
$\tau_d = 0.900$	**0.666 ± 0.032**	**0.598 ± 0.022**
$\tau_d = 0.990$	0.613 ± 0.085	0.586 ± 0.026

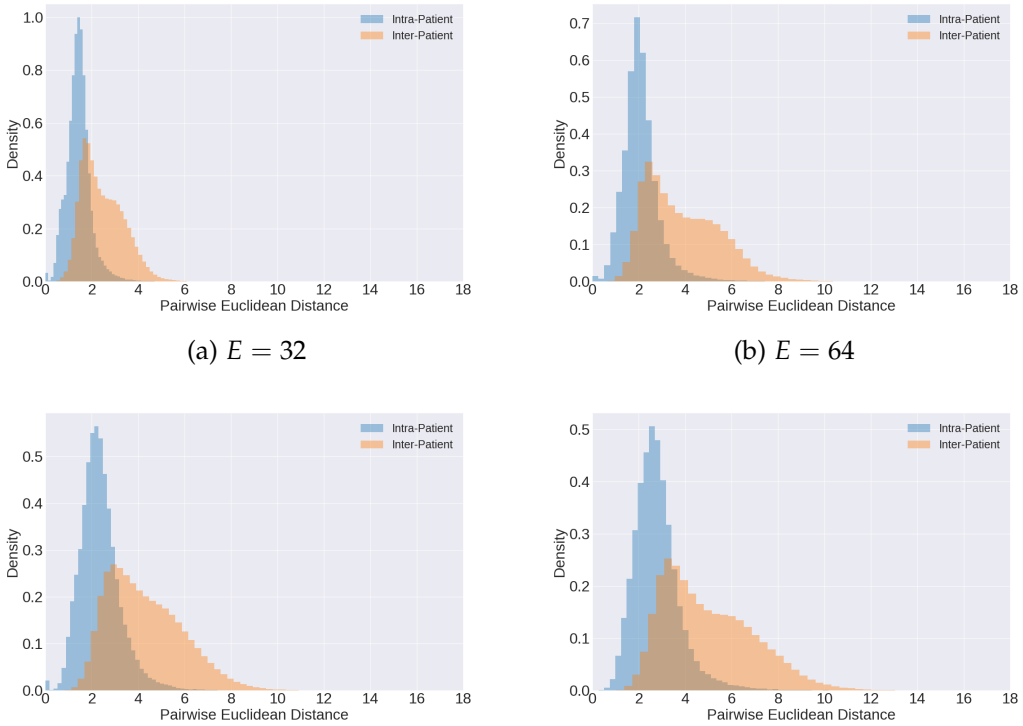
(d) $F = 1$

Dataset	Chapman*	PhysioNet 2020*
$\tau_d = 0.500$	0.637 ± 0.082	0.601 ± 0.008
$\tau_d = 0.900$	**0.653 ± 0.026**	**0.602 ± 0.015**
$\tau_d = 0.990$	0.619 ± 0.088	0.592 ± 0.026

Intra and Inter-patient Representation Distances

In Fig. B.1, we show that, when using a low embedding dimension ($E = 32$), the intra-patient distances are the lowest with a mean of around 1. As $E = 32 \rightarrow 256$, the distributions begin to shift to higher values. Such high pairwise distances imply that

maintaining similar representations at higher dimensions is more difficult. Moreover, we clearly see two distinct distributions belonging to intra-patient and inter-patient distances. This suggests that the training procedure worked as expected, leading to representations that are more similar within patients than across patients.



(a) $E = 32$ (b) $E = 64$

(c) $E = 128$ (d) $E = 256$

Figure B.1: **Distribution of pairwise Euclidean distance between representations belonging to the same patient (Intra-Patient) and those belonging to different patients (Inter-Patient).** Representations are of instances present in the validation set of PhysioNet 2020. Self-supervision was performed with CMSC on PhysioNet 2020 using 4 leads. We find that our framework leads to the learning of patient-specific representations.

C

DOING MORE WITH MORE

C.1 CLUSTERING AND RETRIEVAL OF CARDIAC SIGNALS

C.1.1 *Implementation Details*

Hyperparameters

During optimization, we chose the temperature parameter, $\tau_s = 0.1$ as in Chapter 5, and $\tau_w = 1$. We specified $\beta = 0.2$, in the regularization term, after experimenting with several values. Too small a value of β would decrease the distance between class-specific clinical prototypes. Too large a value of β would cause clinical prototypes from *different* classes to overlap with one another and thus reduce class separability. For Chapman and PTB-XL, sex $\in \{\text{M}, \text{F}\}$, age is converted to quartiles, and $|\text{class}| = 4$ and 5, respectively. Therefore, $M = |\text{class}| \times |\text{sex}| \times |\text{age}| = 32$ and 40, for the two datasets, respectively. The network, f_θ, comprises 1D convolutional operators and we chose the embedding dimension $E = 128$ and 256 for Chapman and PTB-XL, respectively.

Network Architectures

In this section, we outline the neural network architectures used for our experiments. More specifically, we use the architecture shown in Table C.1 for all experiments pertaining to the Chapman dataset. Given the size of the PTB-XL dataset, we opted for a more complex network. We modified the ResNet18 architecture whereby the number of blocks per layer was reduced from two to one, effectively reducing the number of

parameters by a factor of two. We chose this architecture after experimenting with several variants.

Table C.1: **Network architecture used for experiments conducted on the Chapman dataset.** K, C_{in}, and C_{out} represent the kernel size, number of input channels, and number of output channels, respectively. A stride of 3 was used for all convolutional layers. E represents the dimension of the final representation.

Layer Number	Layer Components	Kernel Dimension
1	Conv 1D BatchNorm ReLU MaxPool(2) Dropout(0.1)	$7 \times 1 \times 4$ ($K \times C_{\text{in}} \times C_{\text{out}}$)
2	Conv 1D BatchNorm ReLU MaxPool(2) Dropout(0.1)	$7 \times 4 \times 16$
3	Conv 1D BatchNorm ReLU MaxPool(2) Dropout(0.1)	$7 \times 16 \times 32$
4	Linear ReLU	$320 \times E$

Table C.2: **Batchsize and learning rates used for training with different datasets.** The Adam optimizer was used for all experiments.

Dataset	Batchsize	Learning Rate
Chapman	256	10^{-4}
PTB-XL	128	10^{-5}

Baseline Implementations

DeepCluster. In the implementation by Caron et al. (2018), a forward pass of each instance in the training set is performed. This generates a set of representation which are then clustered, in an unsupervised manner, using k-means. This involves a decision regarding the value of K, i.e., the number of clusters. In our supervised setting, we have this information available and therefore set the value of K to be equal to the number of distinct cardiac arrhythmia classes. Once the clustering is complete, each instance is assigned a pseudo-label according to the cluster to which it belongs.

Such pseudo-labels are used as the ground-truth for supervised training during the next epoch. We repeat this process after each epoch for a total of 30 epochs after realizing that the validation loss plateaus at that point.

IIC. In this implementation, the network is tasked with maximizing the mutual information between the representation of an instance and that of its perturbed counterpart. Such perturbations must be class-preserving and, in computer vision, consist of random crops, rotations, and modifications to the brightness of the images. In our setup involving time-series data, we perturb instances by using additive Gaussian noise in order to avoid erroneously flipping the class of a particular instance. In addition to the aforementioned, we implement the auxiliary over-clustering method suggested by the authors. This approach allows one to model additional 'distractor' classes that may be present in the dataset, and was shown by Ji et al. (2019) to improve generalization performance. In our setup, we set the number of total clusters to the number of attribute combinations, M.

SeLA. In this implementation, each instance is *assigned* a posterior probability distribution. For all instances, this results in an assigned matrix of posterior probability distributions. Each instance's label is obtained by identifying the index associated with the largest posterior probability distribution. Deriving the aforementioned matrix is the crux of SeLA. It does by solving the Sinkhorn-Knopp algorithm under the assumption that the dataset can be evenly split into K clusters. Our setup does not deviate from the original implementation found in Asano et al. (2020).

DTCluster. In this implementation, the distance between each representation and each cluster prototype is calculated to generate a probability distribution over classes, p. The distribution, p, is encouraged to be similar to a target distribution, z, by minimizing the KL divergence of these two distributions. In the original unsupervised implementation, the target distribution is a sharper version of the empirical distribution Han et al. (2019). In our supervised implementation, we initialize the prototypes similarly to our approach and modify the target distribution

to incorporate labels. As with our soft-assignment, we aim for a target distribution that reflects discrepancies, d, between the representation attributes, A_i, and the prototype attributes, A_j. Mathematically, our target distribution, z, is as follows:

$$z_j = \frac{e^{\omega_{ij}}}{\sum_l^{|L|} e^{\omega_{il}}} \tag{C.1}$$

$$\omega_{ij} = \begin{cases} \frac{e^{d(A_i, A_j)}}{\sum_l^{|L|} e^{d(A_i, A_l)}} & \text{if } \alpha_1^i = \alpha_1^j \\ 0 & \text{otherwise} \end{cases} \tag{C.2}$$

$$d(A_i, A_j) = \frac{1}{\tau_\omega} \cdot [\delta(\alpha_c^i = \alpha_c^j) + \delta(\alpha_s^i = \alpha_s^j) + \delta(\alpha_a^i = \alpha_a^j)] \tag{C.3}$$

K-means EP. In this implementation by Gee *et al.* Gee et al. (2019), each instance is first passed through the encoder network to generate a representation. This representation serves multiple functions: a) it is passed through the decoder network to reconstruct the input, and b) passed through a prototype network that works as follows. The Euclidean distance between the representation and M randomly-initialized embeddings (prototypes) is calculated to generate a single M-dimensional representation. This newly-generated representation is then passed through a linear classification head to predict the cardiac arrhythmia class associated with the original instance. In our setup, we set the number of prototypes to coincide with the number of clinical prototypes that we use. For clustering, we apply the k-means algorithm to the representations learned via this framework.

Deep Temporal Clustering Representation. In this implementation by Ma et al. (2019), the network consists of three main components: 1) an encoder, 2) a decoder, and 3) a classifier head. A synthetic version of each instance is first generated by permuting a certain fraction, α, of the time-points in the original instance. The original instance and its synthetic counterpart are then passed through the encoder to obtain a pair of representations (a real and synthetic one). The classifier is tasked with

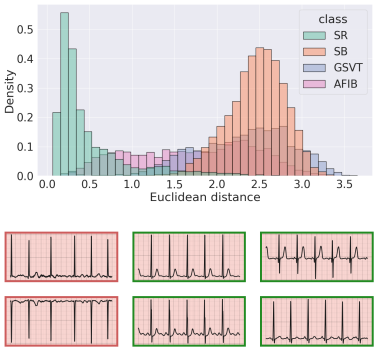
identifying whether such representations are real or fake (binary classification akin to discriminator in generative adversarial networks). Moreover, the decoder reconstructs the original instance by operating on the real representation. Lastly, the k-means loss is approximated based on the Gram matrix of the mini-batch of real representations. We follow the original implementation, and choose $\alpha = 0.2$, and $\lambda = 10^{-3}$ as the coefficient of the k-means loss in the objective function.

C.1.2 *Additional Results*

Deploying Clinical Prototypes in the Retrieval Setting

Chapman. In Chapter 7, we qualitatively evaluated the retrieval performance of a DTC-derived prototype and a clinical prototype on the Chapman dataset. In this section, we continue this evaluation however for a different query; a TP CROCS query, which reflects the average of representations associated with a set of patient attributes. In Fig. C.1 (top row), we present the distributions of the Euclidean distance between the query and the representations in the validation set of Chapman. In Fig. C.1 (bottom row), we illustrate the six cardiac signals that are closest to the query. We find that the query is closer to representations of the class attribute than to those of a different class attribute. This is evident by the long tail of distance values exhibited between representation with SR and the query $\{\text{SR}, \text{male}, \text{under 49}\}$.

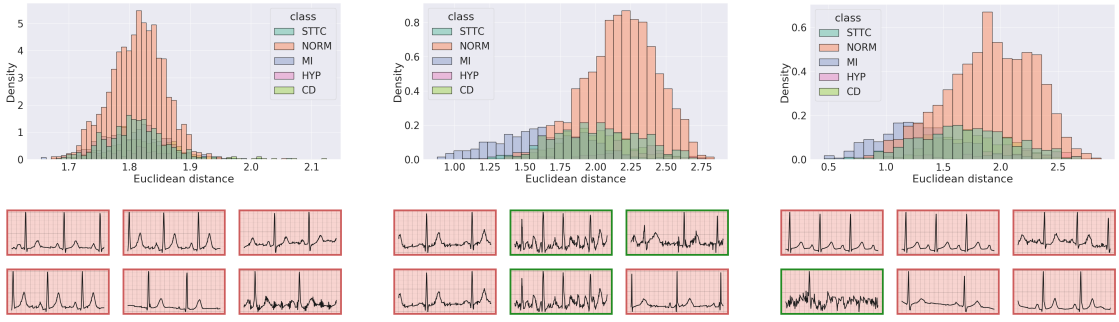
PTB-XL. In this section, we continue our qualitative evaluation of the retrieval performance of various methods. In Fig. C.2 (top row), we present the distributions of the Euclidean distance between the query (DTC-derived prototype or clinical prototype) and the representations in the validation set of PTB-XL. In Fig. C.2 (bottom row), we illustrate the six cardiac signals that are closest to the respective prototypes. As with the results in the main manuscript, we find that the clinical prototype is closer to representations of the same class attribute than to those with a different class attribute. This is evident by the long tail of distance values exhibited between representation with MI and the clinical prototype $\{\text{MI}, \text{female}, \text{over 95}\}$. This behaviour, which is



(a) TP CROCS query {SR, male, under 49}

Figure C.1: **Qualitative retrieval performance of a TP CROCS query.** (**top row**) Euclidean distance from a query to representations in the validation set of Chapman. (**bottom row**) Six closest cardiac signals to each query. The query is associated with a set of patient attributes {disease, sex, age}. Retrieved cardiac signals with green borders indicate those whose class attribute matches that of the query. We see that the mean representation query is closer to representations of the same class (SR) than to those of a different class, and thus retrieves relevant cardiac signals.

non-existent for the DTC-derived prototype, can explain the relatively improved retrieval performance of clinical prototypes. This is further supported by the retrieved cardiac signals (Fig. C.2 bottom row) where the DTC-derived prototype and the clinical prototype retrieve relevant instances 0% and 50% of the time, respectively.



(a) DTC query {MI, female, over 95}

(b) CP CROCS query {MI, female, over 95}

(c) TP CROCS query {MI, female, over 95}

Figure C.2: **Qualitative retrieval performance of two distinct queries.** (**top row**) Euclidean distance from a query (a) DTC (b) CP CROCS or (c) TP CROCS query, to representations in the validation set of PTB-XL. (**bottom row**) Six closest cardiac signals to each query. Each query is associated with a set of patient attributes {disease, sex, age}. Retrieved cardiac signals with green borders indicate those whose class attribute matches that of the query. We show that the clinical prototype is closer to representations of the same class (MI) and thus retrieve relevant cardiac signals.

Investigating Marginal Impact of Design Choices

In this section, we quantify the marginal impact of the design choices of our CROCS framework on the retrieval performance. In Table C.3, we present the precision of retrieved cardiac signals when evaluated based on both partial and exact matches of attributes also represented by the query. Each query is a clinical prototype that is learned in a variant of the CROCS framework. We find that clinical prototypes learned via our full framework (+ Regularization) add value relative to those learned under the Hard assignment framework. For example, at $K = 1$, and when # attribute matches $\geq$ 2, the approaches Hard, Uniform, Modulated, and + Regularization achieve a precision of 27.5, 51.5, 58.5, and 63.0, respectively.

Table C.3: **Marginal impact of design choices of CROCS on the precision of K retrieved representations, v, in the validation set of PTB-XL, that are closest to the query.** Results are shown for partial and exact matches of the attributes (# attribute matches) represented by the query and retrieved cardiac signals, and are averaged across five random seeds. Brackets indicate standard deviation and bold reflects the top-performing method.

# attribute matches	Query	PTB-XL $K = 1$	PTB-XL 5	PTB-XL 10
≥ 1	$\mathcal{L}_{NCE-hard}$	70.0 (3.9)	100.0 (0.0)	100.0 (0.0)
	$\mathcal{L}_{NCE-soft}$			
	$\tau_\omega = \infty$	91.5 (2.0)	100.0 (0.0)	100.0 (0.0)
	$\tau_\omega \neq \infty$	88.0 (6.0)	100.0 (0.0)	100.0 (0.0)
	$+\mathcal{L}_{reg}$	**92.5** (0.0)	100.0 (0.0)	100.0 (0.0)
≥ 2	$\mathcal{L}_{NCE-hard}$	27.5 (3.9)	67.5 (0.0)	93.0 (4.0)
	$\mathcal{L}_{NCE-soft}$			
	$\tau_\omega = \infty$	51.5 (2.0)	94.5 (1.0)	100.0 (0.0)
	$\tau_\omega \neq \infty$	58.5 (2.0)	100.0 (0.0)	100.0 (0.0)
	$+\mathcal{L}_{reg}$	**63.0** (1.9)	96.5 (2.0)	99.5 (1.0)
$= 3$	$\mathcal{L}_{NCE-hard}$	7.0 (1.0)	16.0 (3.0)	26.5 (4.6)
	$\mathcal{L}_{NCE-soft}$			
	$\tau_\omega = \infty$	9.5 (1.0)	30.5 (4.0)	38.5 (3.0)
	$\tau_\omega \neq \infty$	12.5 (0.0)	36.5 (2.0)	43.5 (1.2)
	$+\mathcal{L}_{reg}$	12.5 (0.0)	33.5 (3.0)	43.0 (4.0)

C.2.1 *Implementation Details*

In this section, we outline the hyperparameters and architecture of the neural network used for all experiments. We chose this architecture due to its simplicity and its simultaneous ability to learn on the datasets provided. We also outline the batchsize and the learning rate used for training on the various datasets.

Hyperparameters

Depending on the continual learning scenario, we chose $\tau = 20$ or 40, as we found that to achieve strong performance on the respective validation sets. We chose $\tau_{MC} = 40 + n$ and the sample epochs $\tau_S = 41 + n$ where $n \in \mathbb{N}^+$ in order to sample data from the buffer at every epoch following the first task. The values must satisfy $\tau_S \geq \tau_{MC} > \tau$. For computational reasons, we chose the storage fraction $b = 0.1$ of the size of the training dataset and the acquisition fraction $a = 0.25$ of the number of samples per task in the buffer. To calculate the acquisition function, we chose the number of Monte Carlo samples, $T = 20$. We chose the regularization coefficient, $\lambda = 10$.

Network Architecture

Table C.4: **Network architecture used for all experiments.** K, C_{in}, and C_{out} represent the kernel size, number of input channels, and number of output channels, respectively. A stride of 3 was used for all convolutional layers.

Layer Number	Layer Components	Kernel Dimension
1	Conv 1D BatchNorm ReLU MaxPool(2) Dropout(0.1)	$7 \times 1 \times 4$ ($K \times C_{\text{in}} \times C_{\text{out}}$)
2	Conv 1D BatchNorm ReLU MaxPool(2) Dropout(0.1)	$7 \times 4 \times 16$
3	Conv 1D BatchNorm ReLU MaxPool(2) Dropout(0.1)	$7 \times 16 \times 32$
4	Linear ReLU	320×100
5	Linear	$100 \times C$ (classes)

Table C.5: **Batchsize and learning rates used for training with different datasets.** The Adam optimizer was used for all experiments.

Dataset	Batchsize	Learning Rate
$\mathcal{D}_1$	16	10^{-4}
$\mathcal{D}_2$	256	10^{-4}
$\mathcal{D}_3$	256	10^{-4}

Baseline Replay-based Methods

In this section, we outline how we have adapted two replay-based methods for application in our continual learning scenarios. This adaptation was necessary since both of these methods were designed for a more extreme online setting whereby instances are only seen once and never again.

GEM. At the end training on each task, we randomly select a subset of the instances and store them in the buffer. Since our data are shuffled, this is equivalent to the strategy proposed by the original authors which involves storing the most recent instances. On subsequent tasks, and at each iteration, we check the gradient dot product requirement outlined by the original authors, and if that is violated, we solve a quadratic programming problem. This implementation is exactly the same as that found in the original MIR manuscript.

MIR. At the end of training on each, we randomly select a subset of the instances and store them in the buffer. This is more suitable for our scenarios relative to reservoir sampling, which was implemented by the original authors. As our approach is task-aware, our buffer is task-aware, unlike the original MIR implementation. Such information would only strengthen the performance of our adapted version. On subsequent tasks, and at each iteration, we randomly select a subset of the instances from the buffer, and score them according to the metric proposed by the original authors. We sort these instances in descending order of the scores and acquire the top scoring instances from each task. We ensure that the total number of instances replayed is equal to the number of instances in the mini-batch of the current task.

Evaluation Metrics

Metrics used to evaluate continual learning methodologies are of utmost importance as they provide us with a glimpse of the strengths and weaknesses of various learning strategies. In this section, we outline the traditional metrics used within continual learning. We argue that such metrics are limited in that they only capture the global behaviour of strategies. Consequently, we propose two more fine-grained metrics in Chapter 8.

Average AUC. Let R_j^i represent the performance of the network in terms of AUC on task j after having been trained on task i.

$$\text{Average AUC} = \frac{1}{T}\sum_{j=1}^{T}\mathrm{R}_j^T \tag{C.4}$$

Backward transfer (BWT). $\text{BWT} \in [-1, 1]$ is used to quantify the degree to which training on subsequent tasks affects the performance on previously-seen tasks. Whereas positive BWT is indicative of constructive interference, negative BWT is indicative of destructive interference.

$$\text{BWT} = \frac{1}{T-1}\sum_{j=1}^{T-1}\mathrm{R}_j^T - \mathrm{R}_j^j \tag{C.5}$$

C.2.2 *Additional Results*

Effect of Storage Function Form

In Chapter 8, we proposed the following storage function as a way to guide the storage of instances into the replay buffer. This function is dependent upon the task-instance parameters, β, as they have been tracked throughout the training process.

$$s_{ik} = \int_0^{\tau}\beta_{ik}(t)dt \approx \sum_{t=0}^{\tau}\left(\frac{\beta_{ik}(t+\Delta t)+\beta_{ik}(t)}{2}\right)\Delta t \tag{C.6}$$

In this section, we aim to quantify the effect of the form of the storage function on the generalization performance of our network. More specifically, we explore a different form of the storage function where the task-instance parameters, β, are squared before the trapezoidal rule is applied. Mathematically, the storage function now takes on the following form.

$$s_{ik} = \int_0^{\tau} \beta_{ik}^2(t)dt \approx \sum_{t=0}^{\tau} \left(\frac{\beta_{ik}^2(t+\Delta t) + \beta_{ik}^2(t)}{2} \right) \Delta t \tag{C.7}$$

To compare these two storage functions, we conduct experiments in the Class-IL scenario as we vary the storage fraction, b, and acquisition fraction, a. In Fig. C.3, we present the validation AUC of these two experiments conducted across five seeds. We find that our proposed storage function, and the one used throughout Chapter 8 (Eq. C.6), is more advantageous than that shown in Eq. C.7. This is evident by the consistently higher Average AUC scores. A possible explanation for this observation is the following. Recall that task-instance parameters, β, are a rough proxy for the difficulty of an instance to be classified. Therefore, when ranking such instances based on Eq. C.6 for storage purposes, we are effectively storing the relatively 'easiest-to-classify' instances into the buffer. We hypothesize that upon squaring the task-instance parameters, as is done in Eq. C.7, this interpretation no longer holds. As a result, the discrepancy between instances diminishes and thus hinders the storage process.

Effect of Task Order

The order in which tasks are presented to a continual learner can significantly impact the degree of destructive interference. To quantify this phenomenon and evaluate the robustness of CLOPS to task order, we repeat the Class-IL experiment conducted in Chapter 8 after having randomly shuffled the tasks.

Class IL. Task order does have an effect on destructive interference. By comparing the evaluation metric values shown in Table C.6 and Table 8.3 of Sec. 8.5.1, we show

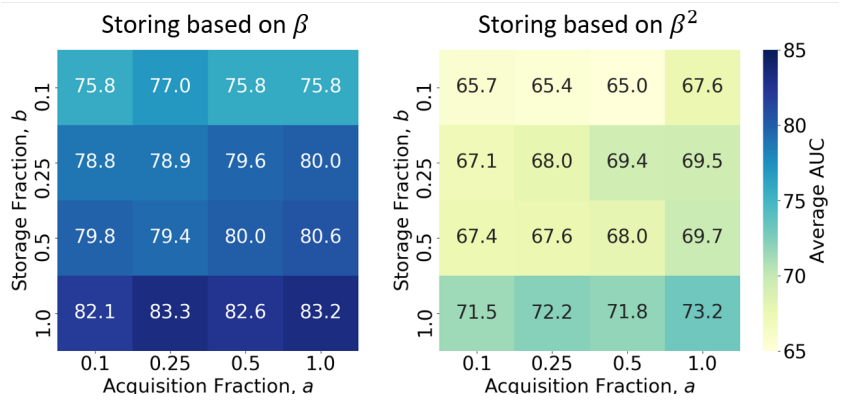
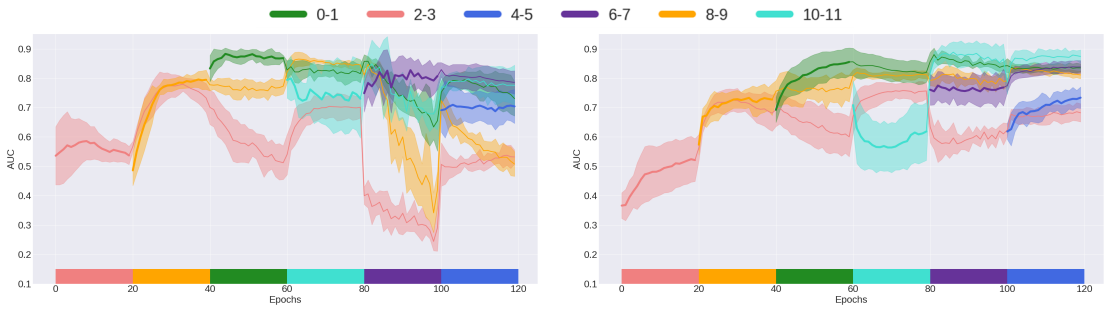


Figure C.3: **Effect of form of storage function on the generalization performance of the network.** Mean validation AUC when buffer storage is based on **(left)** Eq. C.6 and **(right)** Eq. C.7. We show that our proposed storage function, and the one used throughout Chapter 8 (Eq. C.6), is more advantageous than that shown in Eq. C.7.

that poorer generalization performance is achieved when learning sequentially on tasks that have been randomly shuffled. Given that one has limited control over the order in which data is streamed, continual learning approaches must be robust to task order. Indeed, we claim that CLOPS is robust to such changes. This can be seen by its achievement of an AUC $= 0.796$ regardless of task order (in Tables 8.3 and C.6).



(a) Fine-tuning (b) CLOPS

Figure C.4: **Mean validation AUC of a) fine-tuning strategy and b) CLOPS ($b = 0.25$ and $a = 0.50$) in the Class-IL scenario with 6 tasks after being *randomly re-ordered*.** Each task belongs to a mutually-exclusive pair of classes from $\mathcal{D}_1$. Coloured blocks indicate tasks on which the learner is currently being trained. The shaded area represents one standard deviation from the mean across 5 seeds.

Distribution of Task-Instance Parameters. We also illustrate the distribution of the tracked task-instance parameters in Fig. C.5. These distributions differ in terms of overlap compared to those in Fig. 8.7 of Sec. 8.5.1. They both, however, follow a Gaussian distribution and roughly maintain their relative locations to one another. For

Table C.6: **Performance of CL strategies in the Class-IL scenario after tasks have been randomly re-ordered.** Storage and acquisition fractions are $b = 0.25$ and $a = 0.50$, respectively. Results are shown across five seeds. Values enclosed in parentheses are negative.

Method	Average AUC	BWT	BWT_t	BWT_λ
Fine-tuning	0.672 ± 0.022	(0.081) ± 0.049	0.014 ± 0.034	(0.055) ± 0.025
CLOPS	**0.796 ± 0.006**	**0.110 ± 0.021**	**0.096 ± 0.025**	**0.095 ± 0.036**

instance, task $[6, 7]$ and $[10, 11]$ consistently generate the lowest and highest s values, respectively, regardless of task order. Such a finding illustrates that task-instance parameters are agnostic to task-order, which is a desirable trait when attempting to quantify task difficulty.

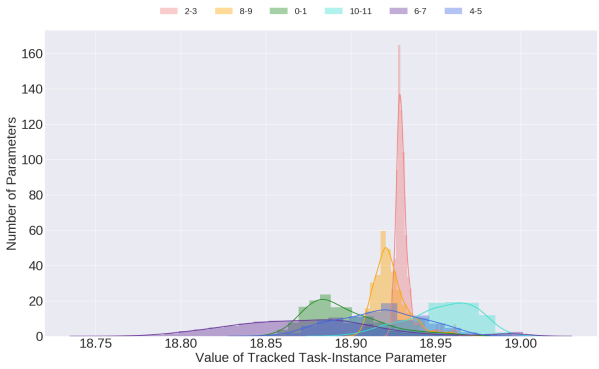


Figure C.5: **Distribution of the tracked task-instance parameter values, s, corresponding to our proposed strategy in the Class-IL scenario with 6 tasks.** Storage and acquisition fractions are $b = 0.25$ and $a = 0.50$, respectively. Each colour corresponds to a different task. Results are shown for one seed. We leverage these distributions to quantify task-similarity.

Effect of Storage and Acquisition Fraction on Performance

Replay-based continual learning strategies can be computationally expensive and resource-hungry due to the presence of a buffer and the need to acquire instances from it. To map out the performance of CLOPS under various resource constraints, we set out to investigate the effect of changes in the storage and acquisition fractions on the various evaluation metrics.

To simultaneously validate our decision of storing the top b instances from each task into the buffer, we conduct the aforementioned experiments in two scenarios: 1) The first scenario involves sorting the instances according to their s value (Eq. 8.4) and storing the top b fraction of instances into the buffer **(Storing top b fraction)**. 2) The second scenario involves storing the bottom b fraction of instances **(Storing bottom b fraction)**. By conducting this specific experiment, we look to determine the relative benefit of replaying either relatively easy or difficult instances.

Average AUC. Larger storage and acquisition fractions should further alleviate destructive interference and improve generalization performance. The intuition is that large fractions will expose the learner to a more representative distribution of data from previous tasks. We quantify this graded response in Fig. C.6 where the AUC $= 0.758 \rightarrow 0.832$ as $b, a = 0.1 \rightarrow 1$. We also claim that a performance bottleneck lies at the storage phase. This can be seen by the larger improvement in performance as a result of an increased storage fraction compared to that observed for a similar increase in acquisition fraction. Despite this, a strategy with fractions as low as $b = 0.25$ and $a = 0.1$ is sufficient to outperform the fine-tuning strategy.

In addition to exploring the graded effect of fraction values, we wanted to explore the effect of storing the bottom, most difficult, b fraction of instances in a buffer. The intuition is that if a learner can perform well on these difficult replayed instances, then strong performance should be expected on the remaining relatively easier instances. We show in Fig. C.6 (right) that although the performance seems to be on par with that in Fig. C.6 (left) for $b = (0.5, 1)$, the most notable differences arise at fractions

$b < 0.5, a < 0.5$ (red box). We believe this is due to the extreme 'hard-to-classify' nature of instances with low s values. These findings justify our storing of the top b fraction of instances.

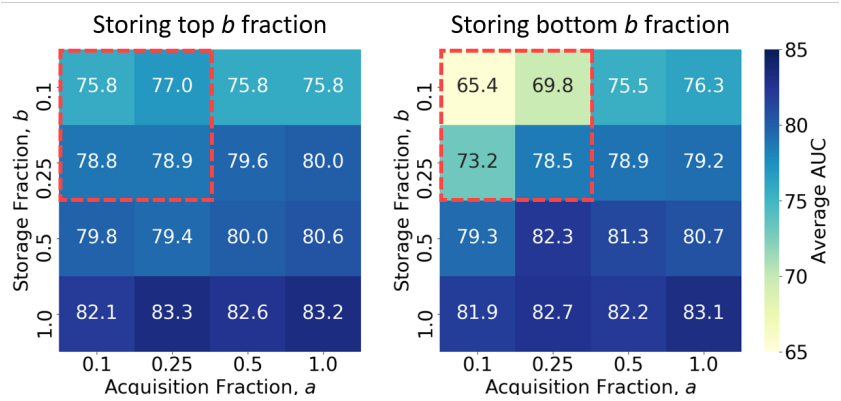
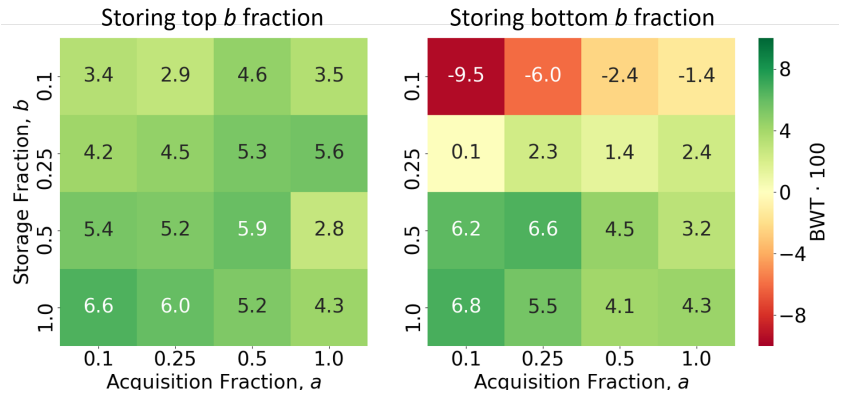


Figure C.6: **Mean validation AUC of CLOPS in the Class-IL scenario while either storing b instances with the (left) highest s value or (right) lowest s value.** Please refer to Eq. 8.4 for the definition of s. Results are shown as a function of storage fractions, b, and acquisition fractions, a and are an average across five seeds. Darker coloured cells indicate higher generalization performance.

Backward weight transfer. In this section, we illustrate the results of the two scenarios 1) Storing top b fraction and 2) Storing bottom b fraction on backward weight transfer. In the top left heatmap, we show that as the storage fraction $b = 0.1 \rightarrow 1$ at an acquisition fraction, $a = 0.1$, is associated with improved constructive interference (BWT $= 0.034 \rightarrow 0.066$). Although this is also true for the scenario in the second column, we show that performance is worse in this case. Such a finding corroborates our claim in Chapter 8 that storing the top instances, which correspond to the 'easiest-to-classify' instances, is more beneficial in the context of CL. This provides further evidence that supports our use of the buffer-storage strategy represented by the first column for all experiments conducted in Chapter 8

Effect of Task-Instance Parameters and Acquisition Function

In this section, we illustrate the effect of two ablation studies on backward weight transfer. **Random Storage** (RS) is a training procedure that stores instances randomly into a buffer yet acquires them using an acquisition function. **Random Acquisition**



(a) BWT

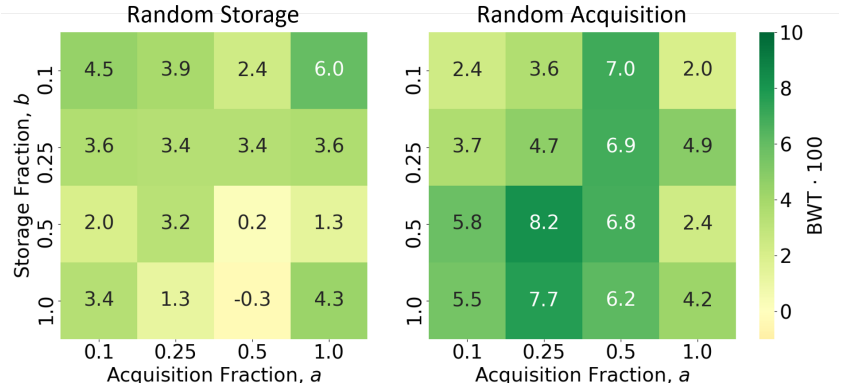
Figure C.7: **Backward weight transfer on the validation set in the Class-IL scenario while using CLOPS and either storing (left column) the top b fraction of instances or (right column) bottom b fraction of instances.** Results are shown as a function of storage fractions, b, and acquisition fractions, a and are an average across five seeds.

(RA), on the other hand, employs task-instance parameters for buffer-storage yet acquires instances randomly from the buffer.

When comparing the BWT_λ heatmaps, we show that the random storage scenario leads to greater destructive interference compared to random acquisition. This can be seen at low storage and acquisition fractions ($b < 0.5$ and $a < 0.5$). Since RA and RS can be thought of as 'smart' storage and 'smart' acquisition, respectively, the superiority of the former implies that a storage strategy is more important than an acquisition strategy in this context and when evaluated from the perspective of backward weight transfer. With most recent CL replay strategies solely focusing on acquisition, our finding suggests that further research into storage functions could be of value.

Effect of Weighting Replayed Instances

In Chapter 8, we stated that replayed instances are *not* weighted with task-instance parameters. Our hypothesis was that this would negatively interfere with the learning process on subsequent tasks. To quantify the effect of such a weighting, we perform several experiments in which replayed instances are weighted according to their corresponding task-instance parameters. Notably, we freeze these parameters and no



(a) BWT

Figure C.8: **Backward weight transfer on the validation set in the Class-IL scenario while using a (left column) Random Storage or (right column) Random Acquisition.** Results are shown as a function of storage fractions, b, and acquisition fractions, a and are an average across five seeds.

longer update them on subsequent tasks. In other words, their previous dual role as a weighting and buffer-storage mechanism now collapses to the former only. In Table C.7, we illustrate the performance of CLOPS with and without the weighting of replayed instances using the task-instance parameters.

We find that weighting *replayed* instances with frozen task-instance parameters is a detriment to backward weight transfer. For example, in the Class-IL scenario, BWT $= 0.053$ and 0.042 for CLOPS without and with the weighting coefficient, respectively. This can be seen across the continual learning scenarios. We hypothesize that this behaviour is due to the 'down-weighting' of replayed instances. More specifically, since task-instance parameters, $\beta < 1$, networks end up learning less from replayed instances.

Qualitative Evaluation of Task-Instance Parameters

In this section, we aim to qualitatively evaluate the interpretation of task-instance parameters as proxies for task-difficulty. To do so for the various CL scenarios, we plot two ECG tracings that correspond to the highest and lowest s values (see Eq. 8.4). As instances with low s values should be more difficult to classify, these ECG tracings might be expected to exhibit abnormalities that make it difficult for a cardiologist to

Table C.7: **Performance of CL strategies in the three continual learning scenarios with and without weighted replayed instances.** Storage and acquisition fractions are $b = 0.25$ and $a = 0.50$, respectively. Mean and standard deviation are shown across five seeds.

Method	Average AUC	BWT	BWT_t	BWT_λ
Class-IL				
CLOPS	0.796 ± 0.013	**0.053 ± 0.023**	**0.018 ± 0.010**	**0.008 ± 0.016**
CLOPS weighted	**0.800 ± 0.006**	0.042 ± 0.020	0.005 ± 0.014	(0.014) ± 0.016
Time-IL				
CLOPS	**0.834 ± 0.014**	**(0.018) ± 0.004**	**(0.007) ± 0.003**	0.007 ± 0.003
CLOPS weighted	0.818 ± 0.012	(0.024) ± 0.012	(0.011) ± 0.006	0.007 ± 0.002
Domain-IL				
CLOPS	0.731 ± 0.001	**(0.011) ± 0.002**	**(0.020) ± 0.004**	**(0.019) ± 0.009**
CLOPS weighted	**0.741 ± 0.007**	(0.016) ± 0.007	(0.024) ± 0.001	(0.024) ± 0.003

correctly diagnose. Conversely, ECG tracings with high s values should be easy to classify from an expert's perspective. We illustrate these tracings, which are coloured based on the task to which they belong, for the Time-IL and Domain-IL scenarios.

Time-IL. In Fig. C.9, we see that the network had a difficult time classifying the ECG tracing that corresponds to the lowest s value with a ground truth label of GSVT (Supra-ventricular Tachycardia). On the other end of the spectrum, the network was able to comfortably classify the ECG tracing that corresponds to the highest s value with a ground truth label of SB (Sudden Bradycardia).

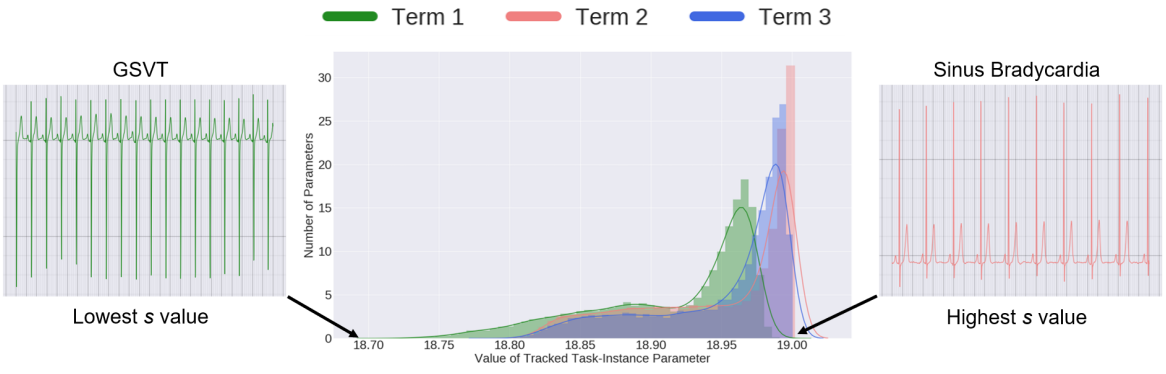


Figure C.9: **Distribution of the tracked task-instance parameter values, s, corresponding to our proposed strategy in the Time-IL scenario.** Each colour corresponds to a different task. Results are shown for one seed. The ECG tracings correspond to the lowest and highest s values.

Domain-IL. In Fig. C.10, we see that the network had a difficult time classifying the ECG tracing that corresponds to the lowest s value with a ground truth label of AF (Atrial Fibrillation). This could be due to the amount of noise present in the tracing. On the other end of the spectrum, the network was able to comfortably classify the ECG tracing that corresponds to the highest s value with a ground truth label of AF also.

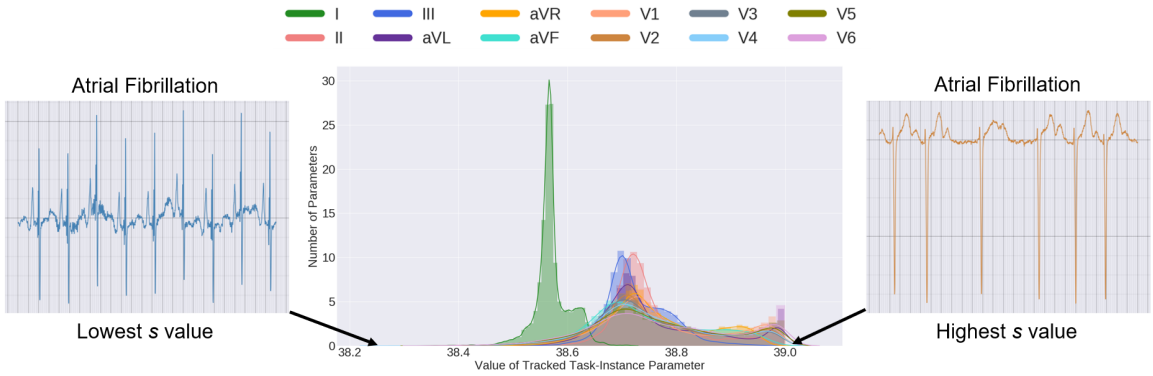


Figure C.10: **Distribution of the tracked task-instance parameter values corresponding to our proposed strategy in the Domain-IL scenario.** Each colour corresponds to a different task. Results are shown for one seed. The ECG tracings correspond to the lowest and highest s values. We find that the tracing associated with the lowest s value contains some noise, which could have confused the network during training.

C.3.1 *Translation Details*

In this section, we outline the steps taken to translate the ECG reports originally found in the PTB-XL dataset. We remind readers that although these ECG reports are a mixture of English and German, they are predominantly in the latter. As a result, we treat German as the source language from which we translate the reports to other languages. More specifically, we follow these steps. 1) We leverage the Google Translate API to first detect the source language of each ECG report. Although the majority of the reports are in German, some are in English, and this language detection step ensures that the ultimate translation is of a higher quality. 2) We continue to leverage the Google Translate API to translate ECG reports from the identified source language to the target language of interest. 3) Due to imperfections in the Google Translate API, certain ECG reports may not be translated in full or translated at all. To minimize the incidence of such cases, we repeat Step 2 several times and stop once we reach the following criterion: we deploy the language detection module of the Google Translate API on the translated reports to confirm that over 90% of them are indeed in the translated language. Although this implies that the final translated reports may have some noise, we found that this did not prevent our algorithm from learning appropriately.

C.3.2 *Implementation Details*

Hyperparameters

We conduct our experiments using PyTorch (Paszke et al., 2019) and the Adam optimizer. We pre-train the encoder and decoder with a patience value of 10 and 25 epochs, respectively, on the validation loss. When transferring the parameters to the task of cardiac signal captioning, we use those associated with the lowest validation

loss. When fine-tuning, we checkpoint the parameters associated with the highest validation BLEU score.

Network Architectures

In this section, we outline the neural network architectures used for our encoder and decoder. More specifically, we use the architecture shown in Table C.8 for the encoder and that shown in Table C.9 for the decoder.

Table C.8: **Encoder architecture used for experiments conducted on the PTB-XL dataset.** K, C_{in}, and C_{out} represent the kernel size, number of input channels, and number of output channels, respectively. A stride of 3 was used for all convolutional layers. M represents the dimension of the final representation. We only use layer 5 when performing supervised pre-training. When captioning, layer 4 outputs L temporal features.

Layer Number	Layer Components	Kernel Dimension
1	Conv 1D BatchNorm ReLU MaxPool(2) Dropout(0.1)	$7 \times 12 \times 32$ ($K \times C_{\text{in}} \times C_{\text{out}}$)
2	Conv 1D BatchNorm ReLU MaxPool(2) Dropout(0.1)	$7 \times 32 \times 64$
3	Conv 1D BatchNorm ReLU MaxPool(2) Dropout(0.1)	$7 \times 64 \times 128$
4	Linear ReLU	$128 \times M$
5	Linear	$M \times$ C (classes)

Table C.9: **Decoder architecture used for experiments conducted on the PTB-XL dataset.** $E = 300$ represents the dimension of the representations both from the encoder and of the decoder tokens. $H = 4$ represents the number of heads used in the self and cross-attention modules. C_{lang} represents the number of tokens in a specific language.

Layer Number	Layer Components	Kernel Dimension
1	Transformer Decoder Layer	E, H
2	Transformer Decoder Layer	E, H
3	Transformer Decoder Layer	E, H
4	Transformer Decoder Layer	E, H
5	Linear	E x C_{lang}

Table C.10: **Batchsize and learning rates used for training.** The Adam optimizer was used for all experiments.

Stage	Batchsize	Learning Rate
Encoder		
Supervised Pre-training	128	10^{-5}
Decoder		
MLM Pre-training	128	10^{-3}
ELECTRA Pre-training	128	10^{-3}
RTLP Pre-training	128	10^{-3}
MARGE Pre-training	64	10^{-4}
Combined		
Fine-tuning	128	10^{-3}

Encoder Pre-training

In this section, we outline the task used to pre-train the encoder of the captioning system in a supervised manner. Specifically, we learn an encoder, $f_\theta : \boldsymbol{x} \in \mathbb{R}^{P \times D} \rightarrow \boldsymbol{y} \in \mathbb{R}^C$ parameterized by $\boldsymbol{\theta}$, that maps $P = 12$ D-dimensional ECG signals, $\boldsymbol{x}$, (where P represents the number of leads) to a C-dimensional output representing the probability assigned to each of the cardiac arrhythmia classes. When leveraging the PTB-XL dataset, $C = 5$. For a mini-batch of size, B, and where c_i represents the ground-truth class for a particular instance, $\boldsymbol{x_i}$, we learn this behaviour by optimizing the following categorical cross-entropy loss.

$$\mathcal{L}_{\mathcal{CE}} = -\frac{1}{B}\sum_{i=1}^{B} \log p_\theta(y_i = c_i) \tag{C.8}$$

We checkpoint, and eventually exploit, the parameters, θ, that coincide with the lowest loss observed on the validation set. This ensures that we use parameters that do not exhibit overfitting.

Baseline Implementations

Masked Language Modelling. Masked language modelling (MLM) can be thought of as analogous to a denoising autoencoder. Inputs are perturbed and the network is tasked with generating the original, unperturbed version of the input. In the context of natural language processing, a fraction $F = 0.15$ of the tokens in a sentence are chosen to be masked. Of these chosen tokens, 80% are replaced with the token [MASK], 10% are replaced with a random token from the vocabulary, and the final 10% are not replaced at all. The motivation behind this task lies in the ability of the network to leverage the context of masked tokens to correctly predict them. This, in turn, allows for the learning of rich representations. In our context, and to allow for a fair comparison to the multilingual pre-training methods, we follow the original implementation introduced by Devlin et al. (2018) for each of the language mini-

batches. More specifically, at each iteration, we load L mini-batches corresponding to L languages and perform MLM on each of these batches.

ELECTRA. ELECTRA, as opposed to MLM introduced above, is a discriminative language representation learning method. ELECTRA builds upon the implementation of MLM in the following ways. First, instead of masking tokens and tasking the network with generating the original token, ELECTRA performs a binary classification of whether a token was replaced or not. The motivation for doing so lies in the alleged unnecessary complexity associated with generative language representation learning methods. Moreover, instead of replacing tokens with the [MASK] token, ELECTRA proposes to do so by exploiting the predicted outputs of an MLM. This increases the likelihood that replaced tokens are in-distribution. As a result, ELECTRA simultaneously trains an MLM network and a binary classifier. In our context, we follow the original implementation introduced by Clark et al. (2020) for each of the L mini-batches.

MARGE. MARGE is a generative multilingual language representation learning method that exploits source documents in various languages to generate text from a similar yet distinct target document. For example, M source documents with $M * S$ tokens are encoded and leveraged by a decoder to generate the T tokens in the target document. In doing so, the network is able to capture relationships between languages and thus learn representations useful for downstream multilingual tasks. In the original implementation (Lewis et al., 2020), similar documents need to be retrieved from a database. In our context, however, our ECG reports are available in L different languages and thus the target document is formed by a report in one language and the source documents are formed by reports in the remaining $L-1$ languages. Since our ECG reports were translated from a single original language, we used reports in this language as target documents. For PTB-XL, this amounts to using German.

BIBLIOGRAPHY

Ablett, J. (1967). Analysis and main experience in 82 patients treated in leeds tetanus unit. In *Symposium on tetanus in Great Britain, 1967*, pages 1–10. Natinal Lending Library.

Adam, G., Rampášek, L., Safikhani, Z., Smirnov, P., Haibe-Kains, B., and Goldenberg, A. (2020). Machine learning approaches to drug response prediction: challenges and recent progress. *NPJ Precision Oncology*, 4(1):1–10.

Adebayo, J., Gilmer, J., Muelly, M., Goodfellow, I., Hardt, M., and Kim, B. (2018). Sanity checks for saliency maps. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 9525–9536, Red Hook, NY, USA. Curran Associates Inc.

Agency, F. R. (2019). Data quality and artificial intelligence - mitigating bias and error to protect fundamental rights.

Akobeng, A. K. (2005). Understanding randomised controlled trials. *Archives of Disease in Childhood*, 90(8):840–844.

Aljundi, R., Belilovsky, E., Tuytelaars, T., Charlin, L., Caccia, M., Lin, M., and Page-Caccia, L. (2019a). Online continual learning with maximal interfered retrieval. In *Advances in Neural Information Processing Systems*, pages 11849–11860.

Aljundi, R., Lin, M., Goujaud, B., and Bengio, Y. (2019b). Gradient based sample selection for online continual learning. In *Advances in Neural Information Processing Systems*, pages 11816–11825.

Antoniou, A., Storkey, A., and Edwards, H. (2017). Data Augmentation Generative Adversarial Networks. *arXiv e-prints*, page arXiv:1711.04340.

Arjovsky, M., Bottou, L., Gulrajani, I., and Lopez-Paz, D. (2019). Invariant risk minimization. *arXiv preprint arXiv:1907.02893*.

Arjovsky, M., Chintala, S., and Bottou, L. (2017). Wasserstein generative adversarial networks. In *International Conference on Machine Learning*, pages 214–223.

Artetxe, M., Labaka, G., and Agirre, E. (2020). Translation artifacts in cross-lingual transfer learning. *arXiv preprint arXiv:2004.04721*.

Asano, Y., Rupprecht, C., and Vedaldi, A. (2020). Self-labelling via simultaneous clustering and representation learning. In *International Conference on Learning Representations*.

Attia, Z. I., Friedman, P. A., Noseworthy, P. A., Lopez-Jimenez, F., Ladewig, D. J., Satam, G., Pellikka, P. A., Munger, T. M., Asirvatham, S. J., Scott, C. G., et al. (2019a). Age and sex estimation using artificial intelligence from standard 12-lead ecgs. *Circulation: Arrhythmia and Electrophysiology*, 12(9):e007284.

Attia, Z. I., Kapa, S., Lopez-Jimenez, F., McKie, P. M., Ladewig, D. J., Satam, G., Pellikka, P. A., Enriquez-Sarano, M., Noseworthy, P. A., Munger, T. M., et al. (2019b). Screening for cardiac contractile dysfunction using an artificial intelligence–enabled electrocardiogram. *Nature Medicine*, 25(1):70–74.

Aznan, N. K. N., Atapour-Abarghouei, A., Bonner, S., Connolly, J., Moubayed, N. A., and Breckon, T. (2019). Simulating brain signals: Creating synthetic EEG data via neural-based generative models for improved ssvep classification. *arXiv preprint arXiv:1901.07429*.

Bachem, O., Lucic, M., and Krause, A. (2018). Scalable k -means clustering via lightweight coresets. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '18, page 1119–1127, New York, NY, USA. Association for Computing Machinery.

Bachman, P., Hjelm, R. D., and Buchwalter, W. (2019). Learning representations by maximizing mutual information across views. In *Advances in Neural Information Processing Systems*, pages 15509–15519.

Banerjee, S. and Lavie, A. (2005). Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

Bansal, G., Nushi, B., Kamar, E., Horvitz, E., and Weld, D. S. (2021). Is the most accurate ai the best teammate? optimizing ai for teamwork. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 11405–11414.

Bate, A., Lindquist, M., Edwards, I. R., Olsson, S., Orre, R., Lansner, A., and De Freitas, R. M. (1998). A bayesian neural network method for adverse drug reaction signal generation. *European journal of clinical pharmacology*, 54(4):315–321.

Begoli, E., Bhattacharya, T., and Kusnezov, D. (2019). The need for uncertainty quantification in machine-assisted medical decision making. *Nature Machine Intelligence*, 1(1):20–23.

Bengio, Y., Louradour, J., Collobert, R., and Weston, J. (2009). Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 41–48.

Biswal, S., Xiao, C., Glass, L. M., Milkovits, E., and Sun, J. (2020). Doctor2vec: Dynamic doctor representation learning for clinical trial recruitment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 557–564.

Borji, A. (2019). Pros and cons of gan evaluation measures. *Computer Vision and Image Understanding*, 179:41–65.

Brailer, D. J., Kroch, E., and Pauly, M. V. (1997). The impact of computer-assisted test interpretation on physician decision making: the case of electrocardiograms. *Medical Decision Making*, 17(1):80–86.

Brisimi, T. S., Chen, R., Mela, T., Olshevsky, A., Paschalidis, I. C., and Shi, W. (2018). Federated learning of predictive models from federated electronic health records. *International journal of medical informatics*, 112:59–67.

Brophy, E., Wang, Z., and Ward, T. E. (2019). Quick and easy time series generation with established image-based gans. *arXiv preprint arXiv:1902.05624*.

Byrd, J. and Lipton, Z. (2019). What is the effect of importance weighting in deep learning? In *International Conference on Machine Learning*, pages 872–881. PMLR.

Caron, M., Bojanowski, P., Joulin, A., and Douze, M. (2018). Deep clustering for unsupervised learning of visual features. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 132–149.

Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P., and Joulin, A. (2020). Unsupervised learning of visual features by contrasting cluster assignments. *arXiv preprint arXiv:2006.09882*.

Cartwright, N. (2007). Are RCTs the gold standard? *BioSocieties*, 2(1):11–20.

Caruana, R. A. (1993). Multitask connectionist learning. In *In Proceedings of the 1993 Connectionist Models Summer School*. Citeseer.

Castro, D. C., Walker, I., and Glocker, B. (2020). Causality matters in medical imaging. *Nature Communications*, 11(1):1–10.

Chakrabarti, S. and Stuart, A. (2005). Understanding cardiac arrhythmias. *Archives of Disease in Childhood*, 90(10):1086.

Chamberlin, S. R., Bedrick, S. D., Cohen, A. M., Wang, Y., Wen, A., Liu, S., Liu, H., and Hersh, W. (2019). Evaluation of patient-level retrieval from electronic health record data for a cohort discovery task. *medRxiv*, page 19005280.

Charlton, P. H., Birrenkott, D. A., Bonnici, T., Pimentel, M. A., Johnson, A. E., Alastruey, J., Tarassenko, L., Watkinson, P. J., Beale, R., and Clifton, D. A. (2017). Breathing rate estimation from the electrocardiogram and photoplethysmogram: A review. *IEEE Reviews in Biomedical Engineering*, 11:2–20.

Chaudhry, A., Ranzato, M., Rohrbach, M., and Elhoseiny, M. (2018). Efficient lifelong learning with a-gem. *arXiv preprint arXiv:1812.00420*.

Chen, R. J., Chen, T. Y., Lipkova, J., Wang, J. J., Williamson, D. F., Lu, M. Y., Sahai, S., and Mahmood, F. (2021). Algorithm fairness in ai for medicine and healthcare. *arXiv preprint arXiv:2110.00603*.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. (2020a). A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*.

Chen, T., Kornblith, S., Swersky, K., Norouzi, M., and Hinton, G. E. (2020b). Big self-supervised models are strong semi-supervised learners. *Advances in Neural Information Processing Systems*, 33:22243–22255.

Chen, X., Fan, H., Girshick, R., and He, K. (2020c). Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*.

Chen, Y., Wang, Y., Kirschen, D., and Zhang, B. (2018). Model-free renewable scenario generation using generative adversarial networks. *IEEE Transactions on Power Systems*, 33(3):3265–3275.

Cheng, J. Y., Goh, H., Dogrusoz, K., Tuzel, O., and Azemi, E. (2020). Subject-aware contrastive learning for biosignals. *arXiv preprint arXiv:2007.04871*.

Chittajallu, D. R., Dong, B., Tunison, P., Collins, R., Wells, K., Fleshman, J., Sankaranarayanan, G., Schwaitzberg, S., Cavuoto, L., and Enquobahrie, A. (2019). Xai-cbir: Explainable ai system for content based retrieval of video frames from minimally invasive surgery videos. In *2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019)*, pages 66–69. IEEE.

Chow, C. (1970). On optimum recognition error and reject tradeoff. *IEEE Transactions on Information Theory*, 16(1):41–46.

Clark, K., Luong, M.-T., Le, Q. V., and Manning, C. D. (2020). Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.

Clifford, G. D., Liu, C., Moody, B., Li-wei, H. L., Silva, I., Li, Q., Johnson, A., and Mark, R. G. (2017). Af classification from a short single lead ECG recording: the physionet/computing in cardiology challenge 2017. In *2017 Computing in Cardiology*, pages 1–4.

Clifford, G. D., Silva, I., Moody, B., Li, Q., Kella, D., Shahin, A., Kooistra, T., Perry, D., and Mark, R. G. (2015). The physionet/computing in cardiology challenge 2015: reducing false arrhythmia alarms in the icu. In *2015 Computing in Cardiology Conference*, pages 273–276.

Cohen, I. G. and Mello, M. M. (2019). Big data, big tech, and protecting patient privacy. *JAMA*, 322(12):1141–1142.

Cohn, D., Atlas, L., and Ladner, R. (1994). Improving generalization with active learning. *Machine Learning*, 15(2):201–221.

Conneau, A., Baevski, A., Collobert, R., Mohamed, A., and Auli, M. (2020). Unsupervised cross-lingual representation learning for speech recognition. *arXiv preprint arXiv:2006.13979*.

Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., and Stoyanov, V. (2019). Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

Conneau, A. and Lample, G. (2019). Cross-lingual language model pretraining. In *Advances in Neural Information Processing Systems*, pages 7059–7069.

Conneau, A., Lample, G., Rinott, R., Williams, A., Bowman, S. R., Schwenk, H., and Stoyanov, V. (2018). Xnli: Evaluating cross-lingual sentence representations. *arXiv preprint arXiv:1809.05053*.

Connell, A., Montgomery, H., Martin, P., Nightingale, C., Sadeghi-Alavijeh, O., King, D., Karthikesalingam, A., Hughes, C., Back, T., Ayoub, K., et al. (2019). Evaluation of a digitally-enabled care pathway for acute kidney injury management in hospital emergency admissions. *NPJ Digital Medicine*, 2(1):1–9.

Cortes, C., DeSalvo, G., and Mohri, M. (2016). Learning with rejection. In *International Conference on Algorithmic Learning Theory*, pages 67–82. Springer.

Darabi, S., Kachuee, M., Fazeli, S., and Sarrafzadeh, M. (2020). Taper: Time-aware patient ehr representation. *IEEE Journal of Biomedical and Health Informatics*.

D'Avolio, L. W., Nguyen, T. M., Farwell, W. R., Chen, Y., Fitzmeyer, F., Harris, O. M., and Fiore, L. D. (2010). Evaluation of a generalizable approach to clinical information retrieval using the automated retrieval console (arc). *Journal of the American Medical Informatics Association*, 17(4):375–382.

Dekel, O., Gentile, C., and Sridharan, K. (2012). Selective sampling and active learning from single and multiple teachers. *Journal of Machine Learning Research*, 13(Sep):2655–2697.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

DeVries, T. and Taylor, G. W. (2017). Dataset Augmentation in Feature Space. *arXiv e-prints*, page arXiv:1702.05538.

Dong, H.-W., Hsiao, W.-Y., Yang, L.-C., and Yang, Y.-H. (2018). Musegan: Multi-track sequential generative adversarial networks for symbolic music generation and accompaniment.

Einthoven, W. (1903). The string galvanometer and the human electrocardiogram. In *KNAW proceedings*, volume 6, pages 107–115.

El-Yaniv, R. and Wiener, Y. (2010). On the foundations of noise-free selective classification. *Journal of Machine Learning Research*, 11(May):1605–1641.

Elgendi, M. (2012). On the analysis of fingertip photoplethysmogram signals. *Current Cardiology Reviews*, 8(1):14–25.

Esteban, C., Hyland, S. L., and Rätsch, G. (2017). Real-valued (Medical) Time Series Generation with Recurrent Conditional GANs. *arXiv e-prints*, page arXiv:1706.02633.

Fawaz, H. I., Forestier, G., Weber, J., Idoumghar, L., and Muller, P.-A. (2018). Data augmentation using synthetic data for time series classification with deep residual networks. *arXiv preprint arXiv:1808.02455*.

Finlayson, S. G., Subbaswamy, A., Singh, K., Bowers, J., Kupke, A., Zittrain, J., Kohane, I. S., and Saria, S. (2020). The clinician and dataset shift in artificial intelligence. *The New England Journal of Medicine*, pages 283–286.

Gal, Y. and Ghahramani, Z. (2016). Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *international Conference on Machine Learning*, pages 1050–1059.

Gal, Y., Islam, R., and Ghahramani, Z. (2017). Deep Bayesian active learning with image data. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1183–1192. JMLR. org.

Gao, M., Zhang, Z., Yu, G., Arik, S. O., Davis, L. S., and Pfister, T. (2019). Consistency-based semi-supervised active learning: towards minimizing labeling cost. *arXiv preprint arXiv:1910.07153*.

Gee, A. H., Garcia-Olano, D., Ghosh, J., and Paydarfar, D. (2019). Explaining deep classification of time-series data with learned prototypes. *arXiv preprint arXiv:1904.08935*.

Geifman, Y. and El-Yaniv, R. (2017). Selective classification for deep neural networks. In *Advances in Neural Information Processing Systems*, pages 4878–4887.

Geifman, Y. and El-Yaniv, R. (2019). Selectivenet: A deep neural network with an integrated reject option. *arXiv preprint arXiv:1901.09192*.

Geirhos, R., Jacobsen, J.-H., Michaelis, C., Zemel, R., Brendel, W., Bethge, M., and Wichmann, F. A. (2020). Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673.

Gerke, S., Yeung, S., and Cohen, I. G. (2020). Ethical and legal aspects of ambient intelligence in hospitals. *Jama*, 323(7):601–602.

Ghosh, A., Kulharia, V., Namboodiri, V. P., Torr, P. H., and Dokania, P. K. (2018). Multi-agent diverse generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8513–8521.

Gidaris, S. and Komodakis, N. (2018). Dynamic few-shot visual learning without forgetting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4367–4375.

Glymour, C., Zhang, K., and Spirtes, P. (2019). Review of causal discovery methods based on graphical models. *Frontiers in Genetics*, 10:524.

Gong, W., Tschiatschek, S., Turner, R., Nowozin, S., and Hernández-Lobato, J. M. (2019). Icebreaker: element-wise active information acquisition with bayesian deep latent gaussian model. *arXiv preprint arXiv:1908.04537*.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative Adversarial Networks. *arXiv e-prints*, page arXiv:1406.2661.

Goodwin, T. R. and Harabagiu, S. M. (2016). Multi-modal patient cohort identification from eeg report and signal data. In *AMIA Annual Symposium Proceedings*, volume 2016, page 1794. American Medical Informatics Association.

Goodwin, T. R. and Harabagiu, S. M. (2018). Learning relevance models for patient cohort retrieval. *JAMIA open*, 1(2):265–275.

Grabocka, J., Schilling, N., Wistuba, M., and Schmidt-Thieme, L. (2014). Learning time-series shapelets. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 392–401.

Gregor Hartmann, K., Tibor Schirrmeister, R., and Ball, T. (2018). EEG-GAN: Generative adversarial networks for electroencephalograhic (EEG) brain signals. *arXiv e-prints*, page arXiv:1806.01875.

Gretton, A., Borgwardt, K. M., Rasch, M. J., Schölkopf, B., and Smola, A. (2012). A kernel two-sample test. *Journal of Machine Learning Research*, 13(Mar):723–773.

Grill, J.-B., Strub, F., Altché, F., Tallec, C., Richemond, P. H., Buchatskaya, E., Doersch, C., Pires, B. A., Guo, Z. D., Azar, M. G., et al. (2020). Bootstrap your own latent: A new approach to self-supervised learning. *arXiv preprint arXiv:2006.07733*.

Guennec, A. L., Malinowski, S., and Tavenard, R. (2016). Data augmentation for time series classification using convolutional neural networks. *Iarchives-ouvertes*.

Gupta, A., Johnson, J., Fei-Fei, L., Savarese, S., and Alahi, A. (2018). Social gan: Socially acceptable trajectories with generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2255–2264.

Gurulingappa, H., Toldo, L., Schepers, C., Bauer, A., and Megaro, G. (2016). Semi-supervised information retrieval system for clinical decision support. In *TREC*.

Gurumurthy, S., Kiran Sarvadevabhatla, R., and Venkatesh Babu, R. (2017). Deligan: Generative adversarial networks for diverse and limited data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 166–174.

Ha, D., Dai, A., and Le, Q. V. (2016). Hypernetworks. *arXiv preprint arXiv:1609.09106*.

Hamburg, M. A. and Collins, F. S. (2010). The path to personalized medicine. *New England Journal of Medicine*, 363(4):301–304.

Han, K., Vedaldi, A., and Zisserman, A. (2019). Learning to discover novel visual categories via deep transfer clustering. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 8401–8409.

Han, X., Hu, Y., Foschini, L., Chinitz, L., Jankelson, L., and Ranganath, R. (2020). Deep learning models for electrocardiograms are susceptible to adversarial attack. *Nature Medicine*, pages 1–4.

Hannun, A. Y., Rajpurkar, P., Haghpanahi, M., Tison, G. H., Bourn, C., Turakhia, M. P., and Ng, A. Y. (2019). Cardiologist-level arrhythmia detection and classification in ambulatory electrocardiograms using a deep neural network. *Nature Medicine*, 25(1):65.

Hasan, S. A., Ling, Y., Liu, J., Sreenivasan, R., Anand, S., Arora, T. R., Datla, V., Lee, K., Qadir, A., Swisher, C., et al. (2018). Attention-based medical caption generation with image modality classification and clinical concept mapping. In *International Conference of the Cross-Language Evaluation Forum for European Languages*, pages 224–230. Springer.

He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. (2019). Momentum contrast for unsupervised visual representation learning. *arXiv preprint arXiv:1911.05722*.

Hersh, W. R. and Greenes, R. A. (1990). Information retrieval in medicine: state of the art. *MD Computing: Computers in Medical Practice*, 7(5):302–311.

Hibbard, J. H., Peters, E., Slovic, P., Finucane, M. L., and Tusler, M. (2001). Making health care quality reports easier to use. *The Joint Commission journal on quality improvement*, 27(11):591–604.

Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S., and Lerchner, A. (2016). Beta-vae: Learning basic visual concepts with a constrained variational framework.

Hinton, G. (2021). How to represent part-whole hierarchies in a neural network. *arXiv preprint arXiv:2102.12627*.

Hoang, M. T. V., Nguyen, T. A., Tran, T. T., Vu, T. T. H., Le, N. T. N., Nguyen, T. H. N., Le, T. H. N., Nguyen, T. T. H., Nguyen, T. H., Le, N. T. N., Truong, H. K., Du, T. Q., Ha, M. T., Ho, L. V., Do, C. V., Nguyen, T. N., Nguyen, T. M. T., Sabanathan, S., Phan, T. Q., Van, V. C. N., Thwaites, G. E., Wills, B., Thwaites, C. L., Le, V. T., and van Doorn, H. R. (2019). Clinical and aetiological study of hand, foot and mouth disease in southern vietnam, 2013–2015: Inpatients and outpatients. *International Journal of Infectious Diseases*, 80:1 – 9.

Houlsby, N., Huszár, F., Ghahramani, Z., and Lengyel, M. (2011). Bayesian active learning for classification and preference learning. *arXiv preprint arXiv:1112.5745*.

Huang, H., Liang, Y., Duan, N., Gong, M., Shou, L., Jiang, D., and Zhou, M. (2019a). Unicoder: A universal language encoder by pre-training with multiple cross-lingual tasks. *arXiv preprint arXiv:1909.00964*.

Huang, K., Altosaar, J., and Ranganath, R. (2019b). Clinicalbert: Modeling clinical notes and predicting hospital readmission. *arXiv preprint arXiv:1904.05342*.

Huang, L., Shea, A. L., Qian, H., Masurkar, A., Deng, H., and Liu, D. (2019c). Patient clustering improves efficiency of federated machine learning to predict mortality and hospital stay time using distributed electronic medical records. *Journal of Biomedical Informatics*, 99:103291.

Hyvarinen, A. and Morioka, H. (2016). Unsupervised feature extraction by time-contrastive learning and nonlinear ica. In *Advances in Neural Information Processing Systems*, pages 3765–3773.

Isele, D. and Cosgun, A. (2018). Selective experience replay for lifelong learning. In *Thirty-second AAAI Conference on Artificial Intelligence*.

Jabbour, S., Fouhey, D., Kazerooni, E., Sjoding, M. W., and Wiens, J. (2020). Deep learning applied to chest x-rays: Exploiting and preventing shortcuts. In *Machine Learning for Healthcare Conference*, pages 750–782. PMLR.

Ji, X., Henriques, J. F., and Vedaldi, A. (2019). Invariant information clustering for unsupervised image classification and segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9865–9874.

Jin, Q., Dhingra, B., Cohen, W. W., and Lu, X. (2019). Probing biomedical embeddings from language models. *arXiv preprint arXiv:1904.02181*.

Keselman, A. and Smith, C. A. (2012). A classification of errors in lay comprehension of medical documents. *Journal of Biomedical Informatics*, 45(6):1151–1163.

Khanh, T. H., Sabanathan, S., Thanh, T. T., Thoa, l. e. P. K., Thuong, T. C., Hang, V. t., Farrar, J., Hien, T. T., Chau, N. v., and van Doorn, H. R. (2012). Enterovirus 71-associated hand, foot, and mouth disease, Southern Vietnam, 2011. *Emerging Infect. Dis.*, 18(12):2002–2005.

Kisilev, P., Sason, E., Barkan, E., and Hashoul, S. (2016). Medical image captioning: Learning to describe medical image findings using multi-task-loss cnn. *Deep Learning for Precision Medicine, Riva del Garda, Italy*.

Kiyasseh, D., Tadesse, G. A., Thwaites, L., Zhu, T., Clifton, D., et al. (2020a). Plethaugment: GAN-based PPG augmentation for medical diagnosis in low-resource settings. *IEEE Journal of Biomedical and Health Informatics*, 24(11):3226–3235.

Kiyasseh, D., Zhu, T., and Clifton, D. (2021). Let your heart speak in its mother tongue: Multilingual captioning of cardiac signals.

Kiyasseh, D., Zhu, T., and Clifton, D. A. (2020b). CLOCS: Contrastive learning of cardiac signals. *arXiv preprint arXiv:2005.13249*.

Kiyasseh, D., Zhu, T., and Clifton, D. A. (2020c). CLOPS: Continual learning of physiological signals.

Kiyasseh, D., Zhu, T., and Clifton, D. A. (2020d). DROPS: Deep retrieval of physiological signals via attribute-specific clinical prototypes.

Kiyasseh, D., Zhu, T., and Clifton, D. A. (2020e). PCPs: Patient cardiac prototypes.

Kiyasseh, D., Zhu, T., and Clifton, D. A. (2020f). SoCal: Selective oracle questioning for consistency-based active learning of cardiac signals.

Koh, P. W., Sagawa, S., Xie, S. M., Zhang, M., Balsubramani, A., Hu, W., Yasunaga, M., Phillips, R. L., Gao, I., Lee, T., et al. (2021). Wilds: A benchmark of in-the-wild distribution shifts. In *International Conference on Machine Learning*, pages 5637–5664. PMLR.

Konyushkova, K., Sznitman, R., and Fua, P. (2017). Learning active learning from data. In *Advances in Neural Information Processing Systems*, pages 4225–4235.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2017). Imagenet classification with deep convolutional neural networks. *Commun. ACM*, 60(6):84–90.

Krueger, D., Caballero, E., Jacobsen, J.-H., Zhang, A., Binas, J., Zhang, D., Priol, R. L., and Courville, A. (2020). Out-of-distribution generalization via risk extrapolation (rex). *arXiv preprint arXiv:2003.00688*.

Kuhn, P., Lang, C., and Wiesbauer, F. (2014). ECG mastery blue belt workbook.

Landi, I., Glicksberg, B. S., Lee, H.-C., Cherng, S., Landi, G., Danieletto, M., Dudley, J. T., Furlanello, C., and Miotto, R. (2020). Deep representation learning of electronic health records to unlock patient stratification at scale. *arXiv preprint arXiv:2003.06516*.

Lee, J., Yoon, W., Kim, S., Kim, D., Kim, S., So, C. H., and Kang, J. (2020). Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

Lenga, M., Schulz, H., and Saalbach, A. (2020). Continual learning for domain adaptation in chest x-ray classification. *arXiv preprint arXiv:2001.05922*.

Lewis, M., Ghazvininejad, M., Ghosh, G., Aghajanyan, A., Wang, S., and Zettlemoyer, L. (2020). Pre-training via paraphrasing. *arXiv preprint arXiv:2006.15020*.

Li, J. and Jurafsky, D. (2016). Mutual information and diverse decoding improve neural machine translation. *arXiv preprint arXiv:1601.00372*.

Li, J., Zhou, P., Xiong, C., Socher, R., and Hoi, S. C. (2020a). Prototypical contrastive learning of unsupervised representations. *arXiv preprint arXiv:2005.04966*.

Li, T., Sahu, A. K., Talwalkar, A., and Smith, V. (2020b). Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60.

Li, Y., Nair, P., Lu, X. H., Wen, Z., Wang, Y., Dehaghi, A. A. K., Miao, Y., Liu, W., Ordog, T., Biernacka, J. M., et al. (2020c). Inferring multimodal latent topics from electronic health records. *Nature Communications*, 11(1):1–17.

Li, Y., Rao, S., Hassaine, A., Ramakrishnan, R., Zhu, Y., Canoy, D., Salimi-Khorshidi, G., Lukasiewicz, T., and Rahimi, K. (2020d). Deep bayesian gaussian processes for uncertainty estimation in electronic health records. *arXiv preprint arXiv:2003.10170*.

Liang, Y., Chen, Z., Liu, G., and Elgendi, M. (2018a). A new, short-recorded photoplethysmogram dataset for blood pressure monitoring in china. *Scientific data*, 5:180020.

Liang, Y., Liu, G., Chen, Z., and Elgendi, M. (2018b). PPG-BP Database.

Lin, C.-Y. (2004). Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Lippi, G., Sanchis-Gomar, F., and Cervellin, G. (2020). Global epidemiology of atrial fibrillation: An increasing epidemic and public health challenge. *International Journal of Stroke*, page 1747493019897870.

Liu, G., Hsu, T.-M. H., McDermott, M., Boag, W., Weng, W.-H., Szolovits, P., and Ghassemi, M. (2019a). Clinically accurate chest x-ray report generation. *arXiv preprint arXiv:1904.02633*.

Liu, Y., Gu, J., Goyal, N., Li, X., Edunov, S., Ghazvininejad, M., Lewis, M., and Zettlemoyer, L. (2020). Multilingual denoising pre-training for neural machine translation. *arXiv preprint arXiv:2001.08210*.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019b). Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Liu, Z., Wang, Z., Liang, P. P., Salakhutdinov, R. R., Morency, L.-P., and Ueda, M. (2019c). Deep gamblers: Learning to abstain with portfolio theory. In *Advances in Neural Information Processing Systems*, pages 10622–10632.

Lopez-Paz, D. and Ranzato, M. (2017). Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems*, pages 6467–6476.

Lu, J., Batra, D., Parikh, D., and Lee, S. (2019). Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Advances in Neural Information Processing Systems*, pages 13–23.

Lucic, M., Bachem, O., and Krause, A. (2016). Strong coresets for hard and soft bregman clustering with applications to exponential family mixtures. volume 51 of *Proceedings of Machine Learning Research*, pages 1–9, Cadiz, Spain. PMLR.

Lyu, X., Hueser, M., Hyland, S. L., Zerveas, G., and Rätsch, G. (2018). Improving clinical predictions through unsupervised time series representation learning. *arXiv preprint arXiv:1812.00490*.

Ma, Q., Zheng, J., Li, S., and Cottrell, G. W. (2019). Learning representations for time series clustering. *Advances in Neural Information Processing Systems*, 32:3781–3791.

Maaten, L. v. d. and Hinton, G. (2008). Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605.

Madiraju, N. S., Sadat, S. M., Fisher, D., and Karimabadi, H. (2018). Deep temporal clustering: Fully unsupervised learning of time-domain features. *arXiv preprint arXiv:1802.01059*.

Mair, S. and Brefeld, U. (2019). Coresets for archetypal analysis. In *Advances in Neural Information Processing Systems*, pages 7247–7255.

Mariani, G., Scheidegger, F., Istrate, R., Bekas, C., and Malossi, C. (2018). Bagan: Data augmentation with balancing gan. *arXiv preprint arXiv:1803.09655*.

Mauch, M. and Ewert, S. (2013). The audio degradation toolbox and its application to robustness evaluation.

McCallumzy, A. K. and Nigamy, K. (1998). Employing em and pool-based active learning for text classification. In *Proc. International Conference on Machine Learning*, pages 359–367.

McFee, B., Humphrey, E. J., and Bello, J. P. (2015). A software framework for musical data augmentation.

McInnes, L., Healy, J., and Melville, J. (2018). Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*.

Mehrabi, N., Morstatter, F., Saxena, N., Lerman, K., and Galstyan, A. (2021). A survey on bias and fairness in machine learning. *ACM Computing Surveys (CSUR)*, 54(6):1–35.

Mikk, K. A., Sleeper, H. A., and Topol, E. J. (2017). The pathway to patient data ownership and better health. *Jama*, 318(15):1433–1434.

Miotto, R., Li, L., Kidd, B. A., and Dudley, J. T. (2016). Deep patient: an unsupervised representation to predict the future of patients from the electronic health records. *Scientific Reports*, 6(1):1–10.

Mirza, M. and Osindero, S. (2014). Conditional Generative Adversarial Nets. *arXiv e-prints*, page arXiv:1411.1784.

Mørup, M. and Hansen, L. K. (2012). Archetypal analysis for machine learning and data mining. *Neurocomputing*, 80:54–63.

Murthy, V. H., Krumholz, H. M., and Gross, C. P. (2004). Participation in cancer clinical trials: race-, sex-, and age-based disparities. *JAMA*, 291(22):2720–2726.

Nassif, R., Vlaski, S., Richard, C., Chen, J., and Sayed, A. H. (2020). Multitask learning over graphs: An approach for distributed, streaming machine learning. *IEEE Signal Processing Magazine*, 37(3):14–25.

Ng, A. (2021). Ai doesn't have to be too complicated or expensive for your business. *Harvard Business Review*.

Obermeyer, Z., Powers, B., Vogeli, C., and Mullainathan, S. (2019). Dissecting racial bias in an algorithm used to manage the health of populations. *Science*, 366(6464):447–453.

Oh, J., Wang, J., and Wiens, J. (2018). Learning to exploit invariances in clinical time-series data using sequence transformer networks. *arXiv preprint arXiv:1808.06725*.

Oktay, O., Nanavati, J., Schwaighofer, A., Carter, D., Bristow, M., Tanno, R., Jena, R., Barnett, G., Noble, D., Rimmer, Y., et al. (2020). Evaluation of deep learning to augment image-guided radiotherapy for head and neck and prostate cancers. *JAMA Network Open*, 3(11):e2027426–e2027426.

Oord, A. v. d., Li, Y., and Vinyals, O. (2018). Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.

Ovadia, Y., Fertig, E., Ren, J., Nado, Z., Sculley, D., Nowozin, S., Dillon, J. V., Lakshminarayanan, B., and Snoek, J. (2019). Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. *arXiv preprint arXiv:1906.02530*.

Pai, S. and Bader, G. D. (2018). Patient similarity networks for precision medicine. *Journal of Molecular Biology*, 430(18):2924–2938.

Pai, S., Hui, S., Isserlin, R., Shah, M. A., Kaka, H., and Bader, G. D. (2019). netdx: Interpretable patient classification using integrated patient similarity networks. *Molecular Systems Biology*, 15(3).

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Parisi, G. I., Kemker, R., Part, J. L., Kanan, C., and Wermter, S. (2019). Continual lifelong learning with neural networks: A review. *Neural Networks*.

Park, D. S., Chan, W., Zhang, Y., Chiu, C.-C., Zoph, B., Cubuk, E. D., and Le, Q. V. (2019). Specaugment: A simple data augmentation method for automatic speech recognition. *arXiv preprint arXiv:1904.08779*.

Pascual, D., Aminifar, A., Atienza, D., Ryvlin, P., and Wattenhofer, R. (2019). Synthetic epileptic brain activities using generative adversarial networks. *arXiv preprint arXiv:1907.10518*.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. (2019). Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pages 8024–8035.

Pearl, J. (1998). Graphical models for probabilistic and causal reasoning. *Quantified representation of uncertainty and imprecision*, pages 367–389.

Pearl, J. (2009). Causal inference in statistics: An overview. *Statistics surveys*, 3:96–146.

Perez Alday, E. A., Gu, A., Shah, A., Liu, C., Sharma, A., Seyedi, S., Bahrami Rad, A., Reyna, M., and Clifford, G. (2020). Classification of 12-lead ECGs: the PhysioNet - computing in cardiology challenge 2020 (version 1.0.1). *PhysioNet*.

Pourmand, A., Tanski, M., Davis, S., Shokoohi, H., Lucas, R., and Zaver, F. (2015). Educational technology improves ecg interpretation of acute myocardial infarction among medical students and emergency medicine residents. *Western Journal of Emergency Medicine*, 16(1):133.

Pratap, V., Sriram, A., Tomasello, P., Hannun, A., Liptchinsky, V., Synnaeve, G., and Collobert, R. (2020). Massively multilingual asr: 50 languages, 1 model, 1 billion parameters. *arXiv preprint arXiv:2007.03001*.

Price, W. N. and Cohen, I. G. (2019). Privacy in the age of medical big data. *Nature Medicine*, 25(1):37–43.

Prosperi, M., Guo, Y., Sperrin, M., Koopman, J. S., Min, J. S., He, X., Rich, S., Wang, M., Buchan, I. E., and Bian, J. (2020). Causal inference and counterfactual prediction in machine learning for actionable healthcare. *Nature Machine Intelligence*, 2(7):369–375.

Qiao, S., Liu, C., Shen, W., and Yuille, A. L. (2018). Few-shot image recognition by predicting parameters from activations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7229–7238.

Quionero-Candela, J., Sugiyama, M., Schwaighofer, A., and Lawrence, N. D. (2009). *Dataset shift in machine learning*. The MIT Press.

Raghu, M., Zhang, C., Kleinberg, J., and Bengio, S. (2019). Transfusion: Understanding transfer learning for medical imaging. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Rahman, F., Kwan, G. F., and Benjamin, E. J. (2014). Global epidemiology of atrial fibrillation. *Nature Reviews Cardiology*, 11(11):639.

Rebuffi, S.-A., Kolesnikov, A., Sperl, G., and Lampert, C. H. (2017). icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2001–2010.

Ren, M., Zeng, W., Yang, B., and Urtasun, R. (2018). Learning to reweight examples for robust deep learning. In *International Conference on Machine Learning*, pages 4334–4343. PMLR.

Rhine, A. M. (2017). *Information Retrieval for Clinical Decision Support*. PhD thesis.

Richley, D. and Walters, H. (2020). Clinical guidelines by consensus recommendations for ECG reporting standards and guidance.

Rolnick, D., Ahuja, A., Schwarz, J., Lillicrap, T., and Wayne, G. (2019). Experience replay for continual learning. In *Advances in Neural Information Processing Systems*, pages 348–358.

Runge, J., Bathiany, S., Bollt, E., Camps-Valls, G., Coumou, D., Deyle, E., Glymour, C., Kretschmer, M., Mahecha, M. D., Muñoz-Marí, J., et al. (2019). Inferring causation from time series in earth system sciences. *Nature communications*, 10(1):1–13.

Sackett, D. L., Rosenberg, W. M., Gray, J. M., Haynes, R. B., and Richardson, W. S. (1996). Evidence based medicine: what it is and what it isn't.

Sagawa, S., Koh, P. W., Hashimoto, T. B., and Liang, P. (2019). Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *arXiv preprint arXiv:1911.08731*.

Salehinejad, H., Colak, E., Dowdell, T., Barfett, J., and Valaee, S. (2018). Synthesizing chest x-ray pathology for training deep convolutional neural networks. *IEEE Transactions on Medical Imaging*, 38(5):1197–1206.

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., and Chen, X. (2016). Improved Techniques for Training GANs. *arXiv e-prints*, page arXiv:1606.03498.

Saporta, A., Gui, X., Agrawal, A., Pareek, A., Truong, S. Q., Nguyen, C. D., Ngo, V.-D., Seekins, J., Blankenberg, F. G., Ng, A., et al. (2021). Deep learning saliency maps do not accurately highlight diagnostically relevant regions for medical image interpretation. *medRxiv*.

Saritha, R. R., Paul, V., and Kumar, P. G. (2019). Content based image retrieval using deep learning process. *Cluster Computing*, 22(2):4187–4200.

Sarkar, P. and Etemad, A. (2020). Self-supervised ecg representation learning for emotion recognition. *arXiv preprint arXiv:2002.03898*.

Saxena, S., Tuzel, O., and DeCoste, D. (2019). Data parameters: A new family of parameters for learning a differentiable curriculum. In *Advances in Neural Information Processing Systems*, pages 11093–11103.

Schulam, P. and Saria, S. (2017). Reliable decision support using counterfactual models. *Advances in Neural Information Processing Systems*, 30:1697–1708.

Sermanet, P., Lynch, C., Hsu, J., and Levine, S. (2017). Time-contrastive networks: Self-supervised learning from multi-view observation. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 486–487. IEEE.

Settles, B. (2009). Active learning literature survey. Technical report, University of Wisconsin-Madison, Department of Computer Sciences.

Severo, D., Amaro, F., Hruschka Jr, E. R., and Costa, A. S. d. M. (2019). Ward2icu: A vital signs dataset of inpatients from the general ward. *arXiv preprint arXiv:1910.00752*.

Shanafelt, T. D., Schein, E., Minor, L. B., Trockel, M., Schein, P., and Kirch, D. (2019). Healing the professional culture of medicine. In *Mayo Clinic Proceedings*, volume 94, pages 1556–1566. Elsevier.

Sharafoddini, A., Dubin, J. A., and Lee, J. (2017). Patient similarity in prediction models based on health data: a scoping review. *JMIR Medical Informatics*, 5(1):e7.

Shen, Z., Liu, Z., Liu, Z., Savvides, M., and Darrell, T. (2020). Rethinking image mixture for unsupervised visual representation learning. *arXiv preprint arXiv:2003.05438*.

Shivade, C., Raghavan, P., Fosler-Lussier, E., Embi, P. J., Elhadad, N., Johnson, S. B., and Lai, A. M. (2014). A review of approaches to identifying patient phenotype cohorts using electronic health records. *Journal of the American Medical Informatics Association*, 21(2):221–230.

Silver, D. L. and Mercer, R. E. (1996). The parallel transfer of task knowledge using dynamic learning rates based on a measure of relatedness. In *Learning to learn*, pages 213–233. Springer.

Sinha, S., Ebrahimi, S., and Darrell, T. (2019). Variational adversarial active learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5972–5981.

Sjoding, M. W., Dickson, R. P., Iwashyna, T. J., Gay, S. E., and Valley, T. S. (2020). Racial bias in pulse oximetry measurement. *New England Journal of Medicine*, 383(25):2477–2478.

Smailagic, A., Costa, P., Gaudio, A., Khandelwal, K., Mirshekari, M., Fagert, J., Walawalkar, D., Xu, S., Galdran, A., Zhang, P., et al. (2019). O-medal: Online active deep learning for medical image analysis. *arXiv preprint arXiv:1908.10508*.

Smailagic, A., Costa, P., Noh, H. Y., Walawalkar, D., Khandelwal, K., Galdran, A., Mirshekari, M., Fagert, J., Xu, S., Zhang, P., et al. (2018). Medal: Accurate and robust deep active learning for medical image analysis. In *IEEE International Conference on Machine Learning and Applications*, pages 481–488.

Smeaton, A. F. (1999). Using nlp or nlp resources for information retrieval tasks. In *Natural language information retrieval*, pages 99–111. Springer.

Snell, J., Swersky, K., and Zemel, R. (2017). Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, pages 4077–4087.

Spiegelhalter, D. (2020). Should we trust algorithms? *Harvard Data Science Review*, 2(1). https://hdsr.mitpress.mit.edu/pub/56lnenzj.

Strodthoff, N., Wagner, P., Schaeffter, T., and Samek, W. (2020). Deep learning for ECG analysis: Benchmarks and insights from PTB-XL. *arXiv preprint arXiv:2004.13701*.

Subbaswamy, A. and Saria, S. (2020). From development to deployment: dataset shift, causality, and shift-stable models in health ai. *Biostatistics*, 21(2):345–352.

Subbaswamy, A., Schulam, P., and Saria, S. (2019). Preventing failures due to dataset shift: Learning predictive models that transport. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 3118–3127. PMLR.

Sun, C., Myers, A., Vondrick, C., Murphy, K., and Schmid, C. (2019). Videobert: A joint model for video and language representation learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7464–7473.

Sung, F., Yang, Y., Zhang, L., Xiang, T., Torr, P. H., and Hospedales, T. M. (2018). Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1199–1208.

Suo, Q., Ma, F., Yuan, Y., Huai, M., Zhong, W., Zhang, A., and Gao, J. (2017). Personalized disease prediction using a cnn-based similarity learning method. In *2017 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 811–816. IEEE.

Thanh-Tung, H., Tran, T., and Venkatesh, S. (2019). Improving generalization and stability of generative adversarial networks. *arXiv preprint arXiv:1902.03984*.

Thickstun, J., Harchaoui, Z., Foster, D., and Kakade, S. M. (2017). Invariances and data augmentation for supervised music transcription. *arXiv e-prints*, page arXiv:1711.04845.

Thompson, N. C., Greenewald, K., Lee, K., and Manso, G. F. (2020). The computational limits of deep learning. *arXiv preprint arXiv:2007.05558*.

Thrun, S. (1998). Lifelong learning algorithms. In *Learning to learn*, pages 181–209. Springer.

Thrun, S. and O'Sullivan, J. (1996). Discovering structure in multiple learning tasks: The tc algorithm. In *ICML*, volume 96, pages 489–497.

Tian, Y., Krishnan, D., and Isola, P. (2019). Contrastive multiview coding. *arXiv preprint arXiv:1906.05849*.

Tran, T., Pham, T., Carneiro, G., Palmer, L., and Reid, I. (2017). A bayesian data augmentation approach for learning deep models. In *Advances in Neural Information Processing Systems*, pages 2797–2806.

Um, T. T., Pfister, F. M. J., Pichler, D., Endo, S., Lang, M., Hirche, S., Fietzek, U., and Kulić, D. (2017). Data augmentation of wearable sensor data for parkinson's disease monitoring using convolutional neural networks. In *Proceedings of the 19th ACM International Conference on Multimodal Interaction*, ICMI '17, pages 216–220, New York, NY, USA. ACM.

Urner, R., David, S. B., and Shamir, O. (2012). Learning from weak teachers. In *Artificial Intelligence and Statistics*, pages 1252–1260.

Van Looveren, A. and Klaise, J. (2019). Interpretable counterfactual explanations guided by prototypes. *arXiv preprint arXiv:1907.02584*.

Verma, V., Lamb, A., Kannala, J., Bengio, Y., and Lopez-Paz, D. (2019). Interpolation consistency training for semi-supervised learning. *arXiv preprint arXiv:1903.03825*.

Vilalta, R. and Drissi, Y. (2002). A perspective view and survey of meta-learning. *Artificial intelligence review*, 18(2):77–95.

Wagner, P., Strodthoff, N., Bousseljot, R.-D., Samek, W., and Schaeffter, T. (2020). PTB-XL, a large publicly available electrocardiography dataset.

Wallace, B. C., Kuiper, J., Sharma, A., Zhu, M., and Marshall, I. J. (2016). Extracting pico sentences from clinical trial reports using supervised distant supervision. *The Journal of Machine Learning Research*, 17(1):4572–4596.

Wang, F., Zhong, S.-h., Peng, J., Jiang, J., and Liu, Y. (2018a). Data augmentation for eeg-based emotion recognition with deep convolutional neural network. In Schoeffmann, K., Chalidabhongse, T. H., Ngo, C. W., Aramvith, S., O'Connor, N. E., Ho, Y.-S., Gabbouj, M., and Elgammal, A., editors, *MultiMedia Modeling*, pages 82–93, Cham. Springer International Publishing.

Wang, G., Zhang, C., Liu, Y., Yang, H., Fu, D., Wang, H., and Zhang, P. (2019a). A global and updatable ecg beat classification system based on recurrent neural networks and active learning. *Information Sciences*, 501:523–542.

Wang, H., Zhang, Q., and Yuan, J. (2017). Semantically enhanced medical information retrieval system: a tensor factorization based approach. *IEEE Access*, 5:7584–7593.

Wang, X., Peng, Y., Lu, L., Lu, Z., and Summers, R. M. (2018b). Tienet: Text-image embedding network for common thorax disease classification and reporting in chest x-rays. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9049–9058.

Wang, Z., Qu, Y., Tao, J., and Song, Y. (2019b). Image-mediated data augmentation for low-resource human activity recognition. In *Proceedings of the 3rd International Conference on Compute and Data Analysis*, ICCDA 2019, pages 49–54, New York, NY, USA. ACM.

Watkins, C. J. C. H. (1989). Learning from delayed rewards.

Wiener, Y. and El-Yaniv, R. (2011). Agnostic selective classification. In *Advances in Neural Information Processing Systems*, pages 1665–1673.

Willemink, M. J., Koszek, W. A., Hardell, C., Wu, J., Fleischmann, D., Harvey, H., Folio, L. R., Summers, R. M., Rubin, D. L., and Lungren, M. P. (2020). Preparing medical imaging data for machine learning. *Radiology*, 295(1):4–15.

Willems, J. L., Abreu-Lima, C., Arnaud, P., van Bemmel, J. H., Brohet, C., Degani, R., Denis, B., Gehring, J., Graham, I., van Herpen, G., et al. (1991). The diagnostic performance of computer programs for the interpretation of electrocardiograms. *New England Journal of Medicine*, 325(25):1767–1773.

Wu, Z., Xiong, Y., Yu, S. X., and Lin, D. (2018). Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3733–3742.

Xie, Q., Dai, Z., Hovy, E., Luong, M.-T., and Le, Q. V. (2019). Unsupervised data augmentation for consistency training.

Yan, S., Chaudhuri, K., and Javidi, T. (2016). Active learning from imperfect labelers. In *Advances in Neural Information Processing Systems*, pages 2128–2136.

Yang, D., Hong, S., Jang, Y., Zhao, T., and Lee, H. (2019). Diversity-Sensitive Conditional Generative Adversarial Networks. *arXiv e-prints*, page arXiv:1901.09024.

Yi, X., Walia, E., and Babyn, P. (2019). Generative adversarial network in medical imaging: A review. *Medical image analysis*, page 101552.

Yoon, W., Lee, J., Kim, D., Jeong, M., and Kang, J. (2019). Pre-trained language model for biomedical question answering. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 727–740. Springer.

You, Q., Jin, H., Wang, Z., Fang, C., and Luo, J. (2016). Image captioning with semantic attention. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4651–4659.

Zeng, X., Wen, L., Liu, B., and Qi, X. (2020). Deep learning for ultrasound image caption generation based on object detection. *Neurocomputing*, 392:132–141.

Zhang, B., Williams, P., Titov, I., and Sennrich, R. (2020a). Improving massively multilingual neural machine translation and zero-shot translation. *arXiv preprint arXiv:2004.11867*.

Zhang, C. and Chaudhuri, K. (2015). Active learning from weak and strong labelers. In *Advances in Neural Information Processing Systems*, pages 703–711.

Zhang, M.-L., Li, Y.-K., Yang, H., and Liu, X.-Y. (2020b). Towards class-imbalance aware multi-label learning. *IEEE Transactions on Cybernetics*.

Zhang, Q. and Liu, Y. (2018). Improving brain computer interface performance by data augmentation with conditional deep convolutional generative adversarial networks. *arXiv preprint arXiv:1806.07108*.

Zhang, Q., Wu, J., Zhang, P., Long, G., and Zhang, C. (2018). Salient subsequence learning for time series clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(9):2193–2207.

Zhang, Y., Jiang, H., Miura, Y., Manning, C. D., and Langlotz, C. P. (2020c). Contrastive learning of medical visual representations from paired images and text. *arXiv preprint arXiv:2010.00747*.

Zheng, J., Zhang, J., Danioko, S., Yao, H., Guo, H., and Rakovski, C. (2020). A 12-lead electrocardiogram database for arrhythmia research covering more than 10,000 patients. *Scientific Data*, 7(1):1–8.

Zhu, X. J. (2005). Semi-supervised learning literature survey. Technical report, University of Wisconsin-Madison, Department of Computer Sciences.

Zhu, Y., Lu, S., Zheng, L., Guo, J., Zhang, W., Wang, J., and Yu, Y. (2018). Texygen: A benchmarking platform for text generation models. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 1097–1100.

Zhu, Z., Yin, C., Qian, B., Cheng, Y., Wei, J., and Wang, F. (2016). Measuring patient similarities via a deep architecture with medical concept embedding. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 749–758. IEEE.