

Deep Generative Models for Natural Language Processing



Yishu Miao
St Hugh's College
University of Oxford

A thesis submitted for the degree of
Doctor of Philosophy

Michaelmas 2017

This thesis is dedicated to
my parents
for their persistent, unequivocal support

Acknowledgements

I am very grateful to many people at this point. My biggest gratitude certainly goes to my supervisor Prof. Phil Blunsom, for advising me through my DPhil studies. No matter how excited, agitated or helpless I was before I entered his office, there will be his sober voice waiting for me and encouraging me “sure go for it”. He usually describes his cool ideas with his unique calmness and patience, which dispels the clouds and helps me see the key points behind everything. I have learned so much from him, not only the way to carry out research work, but also the views on thinking and learning. Thus most of the ideas related to my thesis were born in Phil’s office. I sincerely appreciate his help and supervision.

Second, I’d like to thank my managers Dani Yogatama and Edward Grefenstette for hosting my internships at DeepMind in the years of 2017 and 2016, which are the great memories that I will always remember. Many thanks to Wang Ling and other fellows in the NLP group, who shared so much their experiences with me (on research, engineering and also gaming), and eventually became my good friends. Thanks Karl Hermann for giving me great advice from time to time. Specifically, I want to thank Chris Dyer for helping me so much on organising my research proposals and presentations. He is always able to pinpoint out the emphasis by a few words and share his comments at anytime. I believe everyone around Chris is ready to learn from his passionate and professional minds.

Next, I want to thank my colleagues in ML/CLG research group. I had a great office shared with Jan Buys before we moved to the new one. I really enjoyed the reading group for “deep learning beginners” held with Lei Yu and Pengyu Wang, and the nights we rushed for deadlines together. I also want to thank Prof. Nando de Freitas, who gave me valuable comments on my thesis and even helped me check every equation I listed in my reports. I always admire his patience, enthusiasm and meticulousness on research. In addition, I’d like to thank Tsung-Hsien Wen from Cambridge for the great collaboration on the dialogue models.

I also want to thank my friends in Oxford, for all of the hotpot parties, beers and DotA games. No matter what ups and downs happened in my life or work, they will always be there holding my back. I’m thankful to every one of them. Many thanks to my girlfriend for being with me through the bumpy thesis writing period, and my parents for their persistent and unequivocal support. I will always miss the “Australian snakes” lying in the basement of CS department and the days I spent with them.

Last, many thanks to Prof. Yee Whye Teh and Prof. Noah Smith for being my examiners and providing me lots of valuable comments to improve the thesis.

Abstract

Deep generative models are essential to Natural Language Processing (NLP) due to their outstanding ability to use unlabelled data, to incorporate abundant linguistic features, and to learn interpretable dependencies among data. As the structure becomes deeper and more complex, having an effective and efficient inference method becomes increasingly important. In this thesis, neural variational inference is applied to carry out inference for deep generative models. While traditional variational methods derive an analytic approximation for the intractable distributions over latent variables, here we construct an inference network conditioned on the discrete text input to provide the variational distribution. The powerful neural networks are able to approximate complicated non-linear distributions and grant the possibilities for more interesting and complicated generative models. Therefore, we develop the potential of neural variational inference and apply it to a variety of models for NLP with continuous or discrete latent variables.

This thesis is divided into three parts. Part I introduces a **generic variational inference framework** for generative and conditional models of text. For continuous or discrete latent variables, we apply a continuous reparameterisation trick or the REINFORCE algorithm to build low-variance gradient estimators. To further explore Bayesian non-parametrics in deep neural networks, we propose a family of neural networks that parameterise categorical distributions with continuous latent variables. Using the stick-breaking construction, an unbounded categorical distribution is incorporated into our deep generative models which can be optimised by stochastic gradient back-propagation with a continuous reparameterisation.

Part II explores **continuous latent variable models** for NLP. Chapter 3 discusses the Neural Variational Document Model (NVDM): an unsupervised generative model of text which aims to extract a continuous semantic latent variable for each document. In Chapter 4, the neural topic models modify the neural document models by parameterising categorical distributions with continuous latent variables, where the topics are explicitly modelled by discrete latent variables. The models are further extended to neural unbounded topic models with the help of stick-breaking construction, and a truncation-free variational inference method is proposed based on a Recurrent Stick-breaking construction (RSB). Chapter 5 describes the Neural Answer Selection Model (NASM) for learning a latent stochastic attention mechanism to model the semantics of question-answer pairs and predict their relatedness.

Part III discusses **discrete latent variable models**. Chapter 6 introduces latent sentence compression models. The Auto-encoding Sentence Compression

Model (ASC), as a discrete variational auto-encoder, generates a sentence by a sequence of discrete latent variables representing explicit words. The Forced Attention Sentence Compression Model (FSC) incorporates a combined pointer network biased towards the usage of words from source sentence, which significantly improves the performance when jointly trained with the ASC model in a semi-supervised learning fashion. Chapter 7 describes the Latent Intention Dialogue Models (LIDM) that employ a discrete latent variable to learn underlying dialogue intentions. Additionally, the latent intentions can be interpreted as actions guiding the generation of machine responses, which could be further refined autonomously by reinforcement learning.

Finally, Chapter 8 summarizes our findings and directions for future work.

Contents

1	Introduction	2
1.1	Learning from Language	2
1.2	Deep Generative Models	3
1.3	Contributions	5
1.4	Thesis Structure	7
I	Neural Variational Inference	10
2	Neural Variational Inference	11
2.1	Continuous Latent Variables	12
2.1.1	A Neural Variational Inference Framework	12
2.1.2	Location Reparameterisable Distributions	14
2.2	Discrete Latent Variables	15
2.2.1	REINFORCE Algorithm	15
2.2.2	Control Variates	16
2.2.3	Advanced Sampling Schemes	18
2.3	Parameterising Categorical Distribution	18
2.3.1	Gaussian Softmax Construction	19
2.3.2	Gaussian Stick-breaking Construction	19
2.3.3	Recurrent Stick-breaking Construction	21
2.4	Summary	22
II	Continuous Latent Variable Models	24
3	Neural Document Models	25
3.1	Introduction	25
3.2	Neural Variational Document Model (NVDM)	27

3.3	Experiments	28
3.3.1	Dataset & Setup	28
3.3.2	Evaluation on Perplexity	29
3.3.3	Interpreting Document Semantics	30
3.4	Summary	32
4	Neural Topic Models	33
4.1	Introduction	33
4.2	Neural Finite Topic Model	34
4.2.1	Parameterising the Document-Topic Distribution	35
4.2.2	Parameterising the Topic-Word Distribution	36
4.2.3	Integrating out Latent Topics	37
4.2.4	Topic Models vs. Document Models	38
4.2.5	Topic Diversity	39
4.3	Neural Unbounded Topic Model	39
4.3.1	Unbounded Topics	40
4.3.2	Truncation-free Neural Variational Inference	40
4.4	Experiments	42
4.5	Summary	47
5	Neural Answer Selection Model	49
5.1	Introduction	49
5.2	Neural Answer Selection Model (NASM)	50
5.3	Experiments	53
5.3.1	Dataset & Setup	53
5.3.2	Experimental Results	55
5.3.3	Discussion	56
5.4	Summary	57
III	Discrete Latent Variable Models	58
6	Latent Sentence Compression Models	59
6.1	Introduction	59
6.2	Related Work	61
6.3	Auto-encoding Sentence Compression Model (ASC)	62
6.3.1	Compression	63
6.3.2	Reconstruction	64

6.3.3	Inference	65
6.4	Forced Attention Sentence Compression Model (FSC)	66
6.4.1	Semi-supervised Learning	69
6.4.2	Variance Reduction	69
6.5	Experiments	70
6.5.1	Extractive Summarisation	71
6.5.2	Abstractive Summarisation	72
6.5.3	Discussion	74
6.6	Summary	75
7	Latent Intention Dialogue Models	77
7.1	Introduction	77
7.2	Related Work	78
7.3	Latent Intention Dialogue Model (LIDM)	80
7.3.1	Model Description	80
7.3.2	Inference	82
7.3.3	Semi-Supervision	84
7.3.4	Reinforcement Learning	84
7.4	Experiments	85
7.4.1	Dataset & Setup	85
7.4.2	Experimental Results	87
7.5	Summary	89
8	Conclusions and Further Work	90
A	Details of Inference	93
A.1	Neural Variational Document Model	93
A.2	Neural Answer Selection Model	94
B	Discovered Topics	96
C	Visualisation	98
D	Sentence Compression Examples	99
E	Goal Oriented Dialogue Examples	101
	Bibliography	104

List of Figures

2.1	Graphical Model of the Framework.	13
2.2	Neural Structure of GSM	19
2.3	The Stick-breaking Construction.	20
2.4	Neural Structure of GSB	20
2.5	The unrolled RNN for breaking proportions η	22
2.6	Neural Structure of RSB	22
3.1	Neural Variational Document Model.	27
3.2	Interpreting document semantics as topics.	31
4.1	Network structure of the inference model $q(\theta d)$, and of the generative model $p(d \theta)$	37
4.2	The unrolled RNN that produces the topic-word distributions β	40
4.3	Corpus level topic probability distributions of GSM.	45
4.4	Corpus level topic probability distributions of GSB.	45
4.5	Test perplexities of the neural topic models with a varying maximum number of topics on the 20NewsGroups dataset.	46
4.6	The convergence behavior of the truncation-free RSB model (RSB-TF) with different initial active topics on <i>20NewsGroups</i>	46
5.1	Neural Answer Selection Model.	50
5.2	A visualisation of attention scores on answer sentences.	54
5.3	The standard deviations of MAP scores computed by running 10 NASM models on WikiQA with different numbers of samples.	56
6.1	Auto-encoding Sentence Compression Model	62
6.2	Example of auto-encoding sentence compression.	63
6.3	Forced Attention Sentence Compression Model	67
6.4	Example of forced-attention sentence compression.	68
6.5	Perplexity on validation dataset.	74

7.1	LIDM for Goal-oriented Dialogue Modeling	80
C.1	t-SNE projection of the estimated topic proportions of each document (i.e. $q(\theta d)$) from <i>20NewsGroups</i> . The vectors are learned by the GSM model with 50 topics and each color represents one group from the 20 different groups of the dataset.	98

List of Tables

3.1	Perplexity on test dataset.	30
3.2	The topics learned by NVDM on 20News.	31
3.3	The five nearest words in the semantic space.	32
4.1	Perplexities of the topic models on the test datasets.	43
4.2	Perplexities of document models on the test datasets.	43
4.3	Topic coherence on <i>20NewsGroups</i> (higher is better).	47
5.1	Statistics of <i>QASent</i> and <i>WikiQA</i> . Judgement denotes whether correctness was determined automatically or by human annotators.	53
5.2	Results of our models (LSTM, LSTM + Att, NASM) in comparison with other state of the art models on the <i>QASent</i> and <i>WikiQA</i> dataset.	54
6.1	Extractive Summarisation Performance.	71
6.2	Abstractive Summarisation Performance.	72
6.3	Comparison on validation perplexity.	73
6.4	Comparison on test Rouge scores.	73
7.1	An example of the automatically labeled response seed set for semi-supervised learning during variational inference.	86
7.2	Corpus-based Evaluation.	87
7.3	Human evaluation. The significance test is based on a two-tailed student-t test, between NDM and LIDMs.	88
B.1	Topics learned by GSM on 20NewsGroups dataset.	96
B.2	Topics learned by GSB on 20NewsGroups dataset.	96
B.3	Topics learned by RSB on 20NewsGroups dataset.	97
D.1	Examples of the compression sentences (Part I).	99
D.2	Examples of the compression sentences (Part II).	100

E.1	A sample dialogue from the LIDM, I=100 model, one exchange per block. Each latent intention is shown by a tuple (index, probability), followed by a decoded response. The sample dialogue was produced by following the responses highlighted in bold.	102
E.2	Two sample dialogues from the LIDM+RL, I=100 model, one exchange per block. Comparing to Table E.1, the RL agent demonstrates a much greedier behavior toward task success. This can be seen in block 2 & block 4 in which the agent provides the address and phone number even before the user asks.	103

List of Related Publications

Yishu Miao, Lei Yu, Phil Blunsom. “*Neural Variational Inference for Text Processing*”. In Proceedings of the 33rd International Conference on Machine Learning (ICML), 2016.

Yishu Miao, Phil Blunsom. “*Language as a Latent Variable. Discrete Generative Models for Sentence Compression*”. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2016.

Tsung-Hsien Wen*, Yishu Miao*, Phil Blunsom, Steve J. Young. “*Latent Intention Dialogue Models*”. In Proceedings of the 34th International Conference on Machine Learning (ICML), 2017. (*equal contribution)

Yishu Miao, Edward Grefenstette, Phil Blunsom. “*Discovering Discrete Latent Topics with Neural Variational Inference*”. In Proceedings of the 34th International Conference on Machine Learning (ICML), 2017.

Chapter 1

Introduction

1.1 Learning from Language

Language is a complex system of communication. By speaking, reading, and writing language we are able to communicate our intelligence with other people. It is natural for human beings to learn and develop their language ability while using it in their daily life, however, it is very challenging for a machine to learn both the underlying complexities of a language system and the carried intelligence of a discourse at the same time. Thus a lot of recent progress in Natural Language Processing (NLP) is focused on employing large-scale of annotated corpora for solving specific problems, for example machine translation, sentiment analysis, and linguistic analysis tasks including parsing, part-of-speech tagging and named-entity-recognition. Especially with the recent advances of deep learning, neural networks are widely used as discriminative models to learn end-to-end systems specified for each different problem. Whereas, due to the limitations of the existing labelled datasets, it is very difficult to learn general knowledge and handle multiple different tasks at the same time. Hence, to cope with the bottleneck we ought to explore the vast sea of unlabelled corpora and teach the machine to learn from language by itself. Certainly, there is a long way to go before we have an artificial general intelligence agent learning from language, but to design unsupervised/semi-supervised models exploring unlabelled data to further improve natural language understanding, natural language generation and natural language grounding is a pressing necessity. The difficulty is how to introduce inductive biases and proper priors to guide the models to learn on a mass of disorganized textual corpora, and how to carry out effective and efficient inference for the generative models.

This thesis discusses deep generative models that combine neural networks and probabilistic models focused on unsupervised, semi-supervised or supervised learning

NLP tasks with small amounts of labelled data. With sufficient training data, neural networks are able to fit complex distributions, as demonstrated by a variety of well-developed discriminative models achieving state-of-the-art performance across many fields. In contrast, deep generative models are more difficult to apply successfully due to the fact that they are generally used in scenarios where labelled data is deficient or inaccessible. Meanwhile, the deep neural structure of such generative models requires more advanced inference methods to provide better gradient estimators. Therefore, we choose neural variational inference as the major approach for inference in the deep generative models proposed in this thesis.

What the chosen application scenarios have in common is that there are relatively fewer labelled resources than the tasks well-explored by neural networks such as neural machine translation. We demonstrate the effectiveness of deep generative models on learning stochastic continuous representations of documents (document/topic modelling), constructing discrete (sentence compression) or continuous (question answer selection) representations of sentences, and modelling latent intentions in human utterances (conversational dialogue). With the help of these neural models we are able to better exploit the labelled/unlabelled data and to learn knowledge from language directly.

1.2 Deep Generative Models

Probabilistic generative models underpin many successful applications within the field of Natural Language Processing (NLP). Their popularity stems from their ability to use unlabelled data effectively, to incorporate abundant linguistic features, and to learn interpretable dependencies among data. However these successes are tempered by the fact that as the structure of such generative models becomes deeper and more complex, true Bayesian inference becomes intractable due to the high dimensional integrals required. Hence, in this thesis, the neural variational inference approach is applied to carry out inference in deep generative models. Instead of seeking an analytic approximation, as in traditional variational Bayesian methods, neural variational inference directly applies neural networks to model the posterior probability. Powerful neural networks are able to not only learn good representations, but also approximate complicated non-linear distributions which affords the possibilities for more interesting and complicated generative models applied in NLP. Generally, based on the settings of the latent variables (continuous or discrete), we divide the models into two folds: continuous generative models and discrete generative models.

For continuous latent variable models, an inference network is introduced to parameterise the latent distribution. Typically, a diagonal Gaussian is adopted as the variational distribution, since the reparameterisation trick [84, 56] grants the ability to jointly update the inference networks and generative distributions by back-propagating unbiased and low-variance gradients with a single sample from the variational distribution. For unsupervised learning, the models can be easily built in the framework of variational auto-encoders, where the inference networks generate powerful stochastic vector representations used for document modelling, topic modelling and sentence modelling. For supervised learning, the inference networks are able to be conditioned on all the observations and jointly updated with the model parameters in a tight variational lower bound. Hence, with the help of the effective amortised variational inference with Gaussian reparameterisation, the neural models with continuous latent variables gain more capacity compared to the discriminative counterparts.

For discrete latent variable models, instead, REINFORCE algorithm with control variates [77, 79] is employed due to the high variance issue of sampling-based variational inference. Though the inference process becomes more complicated compared to the continuous models, the discrete latent variables have more advantages for developing the potential of deep neural networks. Generally, the introduction of discrete latent variables allows for clustering, structure prediction and incorporating an inductive bias into existing generative models. Moreover, discrete representations have better interpretability and can be easily employed to incorporate labelled data to carry out semi-supervised learning. In a different vein, the inference network can also be considered as an agent and the samples are the actions made by the agent, then the variational lower bound is the reward that provides the gradient to update the inference networks and generative distributions. Hence, the connection between neural variational inference and reinforcement learning would lead to the emergence of a series of effective and interesting generative models that can be trained with only a handful of labelled data.

In addition, there is a popular research direction in probabilistic models that has been widely explored as Bayesian non-parametrics [103, 27]. Hence, in this thesis, we also propose deep generative models with a stick-breaking construction to explore the potential of non-parametrics in neural networks. By parameterising the categorical distribution with a continuous latent variable, the generative model can be directly trained by stochastic gradient back-propagation with continuous reparameterisation in the neural variational inference. With the help of the recurrent stick-breaking

construction, a truncation-free algorithm can be applied to efficiently and effectively carry out Bayesian non-parametric inference in a neural way.

1.3 Contributions

The contributions of this thesis can be summarised as following:

1. Generic Neural Variational Inference Framework for NLP. For both continuous and discrete latent variable models for NLP, we propose a generic inference framework – neural variational inference which constructs neural networks conditioned on discrete text input to provide the variational distribution. We demonstrate the effectiveness and efficiency of the framework by applying it to unsupervised (neural variational document model), supervised (neural answer selection model) and semi-supervised (auto-encoding sentence compression model) learning models. These models are simple, expressive and can be trained efficiently with either the highly scalable stochastic gradient back-propagation algorithm or the REINFORCE algorithm with control variates. The neural variational framework can be generalised to incorporate any type of neural networks e.g. multilayer perceptrons (MLP), convolutional neural networks (CNN), and recurrent neural networks (RNN). The relevant publication of this chapter is:

Yishu Miao, Lei Yu, Phil Blunsom. “*Neural Variational Inference for Text Processing*”. In Proceedings of the 33rd International Conference on Machine Learning (ICML), 2016.

2. Discrete Representations of Language. Generally, the introduction of discrete latent variables is motivated to improve clustering, structure prediction and incorporate inductive biases into existing generative models. In deep neural networks, they can also be used for modelling discrete interpretable representations other than commonly applied continuous vector representations. Here, as language is naturally discrete, we proposed a new perspective of learning language: modelling language as a latent variable. A sequence of discrete latent variables (representing words) are employed to extract discrete representations of sentences which are language as well. With the help of discrete variational auto-encoder, we are able to learn a variable-length compact summary as the sentence compression. Integrated with a combined pointer network model, we achieved good performance on both abstractive and extractive sentence compression task. The relevant publication of this chapter is:

Yishu Miao, Phil Blunsom. “*Language as a Latent Variable. Discrete Generative Models for Sentence Compression*”. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2016.

3. Bridging Generative Models and Reinforcement Learning. Learning an end-to-end dialogue system is appealing but challenging because of insufficient training data for goal-oriented dialogues results in over-fitting and produces less effective and scalable interactions. By introducing discrete latent variables into neural dialogue systems we decompose the learning of language and the internal dialogue decision-making, where a latent distribution as the policy network models an autonomous decision-making agent for composing appropriate machine responses. In variational inference, the generative model is updated by the learning signal from the variational lower bound. While in reinforcement learning, the latent distribution as policy network is updated by the rewards from dialogue success and sentence BLEU score. Hence, the latent variable bridges the different learning paradigms and brings them together under the same framework, which allows the agent to directly learn dialogue policies through interactions other than solely depending on supervised sequence-to-sequence learning. The relevant publication of this chapter is:

Tsung-Hsien Wen*, Yishu Miao*, Phil Blunsom, Steve J. Young. “*Latent Intention Dialogue Models*”. In Proceedings of the 34th International Conference on Machine Learning (ICML), 2017. (*equal contribution)

4. Bayesian Non-parametrics in Deep Neural Networks. To extend the variational auto-encoder with Bayesian non-parametrics, we introduce a family of neural networks for parameterising categorical distributions with continuous latent variables, in which the parameterisation networks can be easily updated together with the generative model by gradient back-propagation. More importantly, with the help of a stick-breaking construction, we are free to choose the dimension of the discrete latent space which yields to unbounded topics or vector representations for learning documents or sentences. In addition, we propose a recurrent stick-breaking construction for parameterising categorical distribution which allows truncation-free variational inference so that we could dynamically choose the dimension of latent space during the inference process. The relevant publication of this chapter is:

Yishu Miao, Edward Grefenstette, Phil Blunsom. “*Discovering Discrete Latent Topics with Neural Variational Inference*”. In Proceedings of the 34th International Conference on Machine Learning (ICML), 2017.

1.4 Thesis Structure

This thesis is constructed into three different parts: Part I Neural Variational Inference, Part II Continuous Latent Variable Models and Part III Discrete Latent Variable Models. Part I explains the main inference method applied in this thesis which is neural variational inference including the continuous reparameterisation and REINFORCE algorithms for both continuous and discrete latent variables. Part II introduces the continuous latent variable models for NLP, including neural document models, neural topic models and neural answer selection models. Part III discusses the discrete latent variable models, which contains the latent sentence compression models and latent intention dialogue models respectively. Abstracts of each chapters are summarised as following:

Chapter 1 Introduction briefly summarises the intuitions about employing deep generative models for NLP, the proposed continuous/discrete latent variable models and their corresponding inference methods. The contributions and structure of the thesis are also listed in this chapter.

Part I Neural Variational Inference

Chapter 2 Neural Variational Inference introduces a generic variational inference framework for generative and conditional models of text, which is suitable for both unsupervised and supervised learning tasks. For continuous or discrete latent variables, we introduce the continuous reparameterisation trick or REINFORCE algorithm with control variates to build low-variance gradient estimators. In addition, we discuss the alternative location reparameterisable latent distributions for continuous latent variables, and advanced sampling schemes for discrete latent variables. To further explore Bayesian non-parametrics in deep neural networks, we propose a families of neural networks that parameterise categorical distributions with continuous latent variables, including the Gaussian softmax construction (GSM), Gaussian stick-breaking construction (GSB) and recurrent stick-breaking construction (RSB). Using these stick-breaking constructions, an unbounded categorical distribution is

able to be incorporated into the deep generative models, which can be easily updated by stochastic gradient back-propagation with continuous reparameterisation.

Part II Continuous Latent Variable Models

Chapter 3 Neural Document Models discusses the neural variational document model (NVDM), which is an unsupervised generative model of text aiming to extract a continuous semantic latent variable for each document. This model can be interpreted as a variational auto-encoder: an MLP encoder (inference network) compresses the bag-of-words document representation into a continuous latent distribution, and a softmax decoder (generative model) reconstructs the document by generating the words independently. A primary feature of NVDM is that each word is generated directly from a dense continuous document representation instead of the more common binary semantic vector.

Chapter 4 Neural Topic Models extend the neural document models to topic models, where the ‘topics’ are modelled explicitly by parameterised categorical distributions with continuous latent variables. As the topic-word distribution can be easily integrated out, we are able to directly apply continuous reparameterisation trick for updating the inference networks. Moreover, the models are further extended to neural unbounded topic models with the help of stick-breaking construction. In addition, we propose a truncation-free variational inference method using the recurrent stick-breaking construction (RSB), where the categorical distribution is constructed autoregressively by a recurrent neural network. Without pre-defining the number of topics, the unbounded topic models are able to dynamically create new topics during the inference process.

Chapter 5 Neural Answer Selection Models discuss supervised conditional models which imbue LSTMs with a latent stochastic attention mechanism to model the semantics of question-answer pairs and predict their relatedness. An attention model is designed to focus on the phrases of an answer that are strongly connected to the question semantics and is modelled by a latent distribution. This mechanism allows the model to deal with the ambiguity inherent in the task and learns pair-specific representations that are more effective at predicting answer matches, rather than independent embeddings of question and answer sentences. Bayesian inference provides a natural safeguard against overfitting, especially as the training sets available for this task are small. The experiments show that the LSTM with a latent

stochastic attention mechanism learns an effective attention model and outperforms both previously published results, and our proposed strong non-stochastic attention baselines.

Part III Discrete Latent Variable Models

Chapter 6 Latent Sentence Compression Models describes the applications of discrete latent variable models on sentence compression and goal-oriented dialogue systems. By introducing a sequence of discrete latent variables representing explicit words, we are able to use language to model the discrete representations of sentences, in which case the language is a latent variable that summarises meaning of the source sentences. We construct this Auto-encoding Sentence Compression Model (ASC) in the framework of variational auto-encoder, where the latent variables are discrete rather than continuous. In order to further improve the performance of sentence compression, we introduce a combined pointer network into the sequence-to-sequence learning model, where it balances the probability of choosing a word from the source sentence or generating a new word from the full vocabulary. The Forced Attention Sentence Compression Model (FSC) can be trained together with ASC model in a semi-supervised learning framework, and provide extra supervision as supplementary to the unsupervised auto-encoding sentence compression. The models are able to be applied to both extractive sentence summarisation and abstractive sentence summarisation.

Chapter 7 Latent Intention Dialogue Models introduces Latent Intention Dialogue Models (LIDM) that employ a discrete latent variable to learn underlying dialogue intentions in the framework of neural variational inference. Different from the end-to-end discriminative models that directly learns to generate natural language sentences as machine response, LIDM employs latent intentions as actions guiding the generation of machine responses in the goal-oriented dialogue scenario. To further improve the success rate of the dialogues, we incorporate semi-supervised learning and reinforcement learning to refine the modelling of the latent distributions. The experiments demonstrate the effectiveness of discrete latent variable models on learning goal-oriented dialogues, and the results outperform the published benchmarks on both corpus-based evaluation and human evaluation.

Chapter 8 Conclusions and Further Work summarises the findings of this thesis. We also discuss the works and directions to be explored in the future.

Part I

Neural Variational Inference

Chapter 2

Neural Variational Inference

Latent variable modelling is popular in many NLP problems, but it is non-trivial to carry out effective and efficient inference for models with complex and deep structure. True Bayesian inference becomes intractable due to the high dimensional integrals required. For example, Markov chain Monte Carlo (MCMC) [81, 1] and variational inference [47, 2, 7] are the standard approaches for approximating these integrals. However the computational cost of the former results in impractical training for the large and deep neural networks which are now fashionable, and the latter is conventionally confined due to the underestimation of posterior variance. The lack of effective and efficient inference methods hinders our ability to create highly expressive models of text, especially in the situation where the model is non-conjugate.

This chapter introduces a neural variational framework for generative models of text processing, inspired by the variational auto-encoder [84, 56]. The principle idea is to build an inference network, implemented by a deep neural network conditioned on text, to approximate the distributions over the latent variables. Instead of providing an analytic approximation, as in traditional variational Bayes, neural variational inference learns to model the posterior probability, thus endowing the model with strong generalisation abilities. Due to the flexibility of deep neural networks, the inference network is capable of learning complicated non-linear distributions and processing structured inputs such as word sequences. Inference networks can be designed as, but not restricted to, multilayer perceptrons (MLP), convolutional neural networks (CNN), and recurrent neural networks (RNN), approaches which are rarely used in conventional generative models.

Training an inference network to approximate the variational distribution was first proposed in the context of Helmholtz machines [42, 39, 21], but applications of these directed generative models come up against the problem of establishing low variance gradient estimators, especially when applied in deep neural networks. Recent

advances in neural variational inference mitigate this problem by reparameterising the continuous random variables [84, 56], using control variates [77] or approximating the posterior with importance sampling [13]. The instantiations of these ideas [28, 54, 4] have demonstrated strong performance on the tasks of image processing. This chapter discusses the neural variational inference methods for continuous latent variables, discrete latent variables and non-parametric extensions.

2.1 Continuous Latent Variables

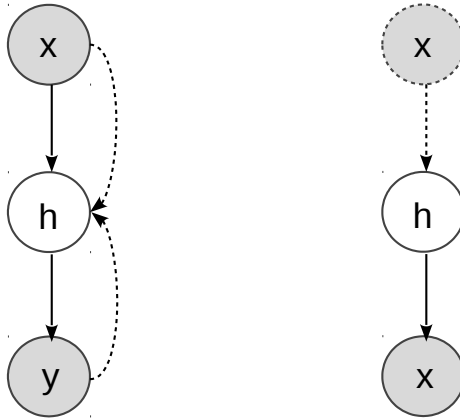
In traditional graphical models, continuous latent variables are usually applied for factor analysis, where the latent distribution describes the correlations and joint variations of the factors. In neural networks, however, we consider stochastic representations as more powerful and expressive than their deterministic counterparts, but generally do not model correlations directly via latent distributions. Hence, the diagonal Gaussian is a default selection as the latent distribution. By using the reparameterisation method [84, 56], the variational lower bound can be easily optimised by back-propagating unbiased and low variance gradients w.r.t. the continuous latent variables. The following contents focus on the inference method for generative models with continuous latent variables, while the related applications in natural language processing will be discussed in the next chapter.

2.1.1 A Neural Variational Inference Framework

Here we introduce a generic neural variational inference framework for continuous latent variable models. We define a generative model with a latent variable \mathbf{h} , which can be considered as the stochastic units in deep neural networks. We designate the observed parent and child nodes of \mathbf{h} as \mathbf{x} and \mathbf{y} respectively (e.g. Figure 2.1a). Hence, the joint distribution of the generative model is $p_\theta(\mathbf{x}, \mathbf{y}) = \sum_{\mathbf{h}} p_\theta(\mathbf{y}|\mathbf{h})p_\theta(\mathbf{h}|\mathbf{x})p(\mathbf{x})$, and the variational lower bound \mathcal{L} is derived as:

$$\begin{aligned} \mathcal{L} &= \mathbb{E}_{q(\mathbf{h})}[\log p_\theta(\mathbf{y}|\mathbf{h})p_\theta(\mathbf{h}|\mathbf{x})p(\mathbf{x}) - \log q(\mathbf{h})] \\ &\leq \log \int \frac{q(\mathbf{h})}{q(\mathbf{h})} p_\theta(\mathbf{y}|\mathbf{h})p_\theta(\mathbf{h}|\mathbf{x})p(\mathbf{x})d\mathbf{h} = \log p_\theta(\mathbf{x}, \mathbf{y}) \end{aligned} \quad (2.1)$$

where θ parameterises the generative distributions $p_\theta(\mathbf{y}|\mathbf{h})$ and $p_\theta(\mathbf{h}|\mathbf{x})$. In order to have a tight lower bound, the variational distribution $q(\mathbf{h})$ should approach the true posterior $p(\mathbf{h}|\mathbf{x}, \mathbf{y})$. Here, we employ a parameterised diagonal Gaussian $\mathcal{N}(\mathbf{h}|\boldsymbol{\mu}(\mathbf{x}, \mathbf{y}), \boldsymbol{\sigma}^2(\mathbf{x}, \mathbf{y}))$ as $q_\phi(\mathbf{h}|\mathbf{x}, \mathbf{y})$. Throughout this thesis we employ diagonal



(a) General Learning Case (b) Unsupervised Learning Case

Figure 2.1: Graphical Model of the Framework.

Gaussian distributions. As such we use $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2)$ to represent the Gaussian distributions, where $\boldsymbol{\sigma}^2$ is the diagonal of the covariance matrix. The three steps to construct the inference network are:

- (1) Construct vector representations of the observed variables: $\mathbf{u} = f_x(\mathbf{x})$, $\mathbf{v} = f_y(\mathbf{y})$.
- (2) Assemble a joint representation: $\boldsymbol{\pi} = g(\mathbf{u}, \mathbf{v})$.
- (3) Parameterise the variational distribution over the latent variable: $\boldsymbol{\mu} = l_1(\boldsymbol{\pi})$, $\log \boldsymbol{\sigma} = l_2(\boldsymbol{\pi})$.

$f_x(\cdot)$ and $f_y(\cdot)$ can be any type of deep neural network that are suitable for the observed data; $g(\cdot)$ is an MLP that concatenates the vector representations of the conditioning variables; $l(\cdot)$ is a linear transformation which outputs the parameters of the Gaussian distribution. By sampling from the variational distribution, $\mathbf{h} \sim q_\phi(\mathbf{h}|\mathbf{x}, \mathbf{y})$, we are able to use stochastic back-propagation to optimise the lower bound (Eq. 2.1).

During training, the model parameters θ together with the inference network parameters ϕ are updated by stochastic back-propagation based on the samples \mathbf{h} drawn from $q_\phi(\mathbf{h}|\mathbf{x}, \mathbf{y})$. For the gradients w.r.t. θ , we have the form:

$$\nabla_\theta \mathcal{L} \simeq \frac{1}{L} \sum_{l=1}^L \nabla_\theta \log p_\theta(\mathbf{y}|\mathbf{h}^{(l)}) p_\theta(\mathbf{h}^{(l)}|\mathbf{x}) \quad (2.2)$$

For the gradients w.r.t. ϕ we reparameterise $\mathbf{h} = \boldsymbol{\mu} + \boldsymbol{\sigma} \cdot \boldsymbol{\epsilon}$ and sample $\boldsymbol{\epsilon}^{(l)} \sim \mathcal{N}(0, \mathbf{I})$ to reduce the variance in stochastic estimation [84, 56]. The update of ϕ can be carried

out by back-propagating the gradients w.r.t. $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$:

$$s(\mathbf{h}) = \log p_\theta(\mathbf{y}|\mathbf{h})p_\theta(\mathbf{h}|\mathbf{x}) - \log q_\phi(\mathbf{h}|\mathbf{x}, \mathbf{y})$$

$$\nabla_\mu \mathcal{L} \simeq \frac{1}{L} \sum_{l=1}^L \nabla_{\mathbf{h}^{(l)}} [s(\mathbf{h}^{(l)})] \quad (2.3)$$

$$\nabla_\sigma \mathcal{L} \simeq \frac{1}{2L} \sum_{l=1}^L \boldsymbol{\epsilon}^{(l)} \nabla_{\mathbf{h}^{(l)}} [s(\mathbf{h}^{(l)})] \quad (2.4)$$

It is worth mentioning that unsupervised learning is a special case of the neural variational framework (e.g. Figure 2.1b) where \mathbf{h} has no parent node \mathbf{x} . In that case \mathbf{h} is directly drawn from the prior $p(\mathbf{h})$ instead of the conditional distribution $p_\theta(\mathbf{h}|\mathbf{x})$, and $s(\mathbf{h}) = \log p_\theta(\mathbf{y}|\mathbf{h})p_\theta(\mathbf{h}) - \log q_\phi(\mathbf{h}|\mathbf{y})$, hence it reduces to a variational auto-encoder [56].

2.1.2 Location Reparameterisable Distributions

Generally, the continuous reparameterisation trick [56, 84] is carried out with an diagonal Gaussian distribution. The advantages of the Gaussian reparameterisation can be concluded into two folds: 1) the variational lower bound is differentiable; 2) the Gaussian KL divergence can be integrated out. Thus the estimated variational bound can be carried out by using one sample, and the gradient estimator is unbiased and low-variance. However, there are also disadvantages of using Gaussian as latent distribution: 1) the unimodality of Gaussian makes the latent distribution difficult to fit in multimodal situations; 2) Gaussian is exponentially bounded so it has no heavy tails. Nevertheless, Gaussian as variational distribution modelled by inference networks is satisfiable in most cases we have explored.

In addition to the Gaussian distribution, there are some other commonly used distributions that have inverse CDF can be applied with the reparameterisation tricks. For example, the Cauchy distribution has quantile function (inverse CDF):

$$q(p; x, \gamma) = x + \gamma \cdot \tan(\pi \cdot (\xi - \frac{1}{2})) \quad (2.5)$$

where $\xi \sim \text{Uniform}(0, 1)$, x and γ are the median and scale respectively. Therefore, the Cauchy can be an alternative distribution to the Gaussian for carrying out neural variational inference of continuous latent variables. Whereas, Cauchy has a closed-form solution for entropy, but the cross-entropy term requires sampling, which might cause high variance. In addition, Beta distribution, Gaussian mixture distribution or logistic normal distribution can also be considered in some scenarios.

2.2 Discrete Latent Variables

The previous section discussed the scenario where the latent variables are continuous and the parameterised diagonal Gaussian is generally employed as the latent distribution. In fact, the aforementioned framework is also suitable for discrete latent variables, and the only modification needed is to replace the Gaussian with a multinomial distribution parameterised by the outputs of a softmax function. Compared to generative models with continuous latent variables, models with discrete ones are more suitable for building predictive systems. In addition, discrete representations are easier to be interpreted, and it is more straightforward to incorporate inductive biases to benefit the learning of deep neural networks. More importantly, discrete latent variable models, provide a principled framework for semi-supervised learning [54, 74, 115]. This is critical for NLP tasks, especially where additional supervision and external knowledge can be utilized for bootstrapping. However, variational inference with discrete latent variables is relatively difficult due to intractable marginalisation and the problem of high variance during sampling, because the Gaussian reparameterisation trick for continuous variables is not applicable for this case. Therefore, a policy gradient approach [77] can help to alleviate the high variance problem during the sampling-based neural variational inference, which is also known as REINFORCE algorithm. The recent works [71, 46] propose a categorical reparameterization approach with the help of Gumbel-Softmax trick, which allows efficient inference for discrete latent variables. Here, in this chapter, we discuss the general REINFORCE Algorithm, which is the main approach applied for the discrete latent variable models proposed in this thesis.

2.2.1 REINFORCE Algorithm

REINFORCE is a policy gradient method commonly used in reinforcement learning problems. Here, we apply it to help construct an unbiased gradient estimator for discrete latent distributions. Assume an unsupervised learning case, where \mathbf{z} is the discrete latent variable and \mathbf{x} is the observation. The variational lower bound \mathcal{L} is:

$$\begin{aligned}\mathcal{L} &= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})p(\mathbf{z}) - \log q_\phi(\mathbf{z}|\mathbf{x})] & (2.6) \\ &= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - D_{KL}[q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z})] \\ &\leq \log \sum_{\mathbf{z}} \frac{q_\phi(\mathbf{z}|\mathbf{x})}{q_\phi(\mathbf{z}|\mathbf{x})} p_\theta(\mathbf{x}, \mathbf{z}) = \log p(\mathbf{x})\end{aligned}$$

where θ parameterises the generative distributions $p_\theta(\mathbf{z}|\mathbf{x})$ and ϕ is the parameter set of the inference network. As the continuous reparameterisation trick is not applicable, we generate M samples $\mathbf{z}^{(m)} \sim q_\phi(\mathbf{z}|\mathbf{x})$ for Monte Carlo estimation, and the estimated variational lower bound is:

$$\mathcal{L} \approx \frac{1}{M} \sum_m [\log p_\theta(\mathbf{x}, \mathbf{z}^{(m)}) - \log q_\phi(\mathbf{z}^{(m)}|\mathbf{x})] \quad (2.7)$$

Therefore, for the parameters θ in the generative model, we directly update:

$$\frac{\partial L}{\partial \theta} = \frac{1}{M} \sum_m \frac{\partial \log p_\theta(\mathbf{x}, \mathbf{z}^{(m)})}{\partial \theta} \quad (2.8)$$

and for the parameters ϕ in the recognition model, we update:

$$\begin{aligned} \frac{\partial L}{\partial \phi} &= \frac{1}{M} \sum_m \frac{\partial q_\phi(\mathbf{z}^{(m)}|\mathbf{x})}{\partial \phi} [\log p_\theta(\mathbf{x}, \mathbf{z}^{(m)}) - \log q_\phi(\mathbf{z}^{(m)}|\mathbf{x})] \\ &\quad - \frac{1}{M} \sum_m \frac{\partial \log q_\phi(\mathbf{z}^{(m)}|\mathbf{x})}{\partial \phi} \cdot q_\phi(\mathbf{z}^{(m)}|\mathbf{x}) \\ &= \frac{1}{M} \sum_m q_\phi(\mathbf{z}^{(m)}|\mathbf{x}) \cdot \frac{\partial \log q_\phi(\mathbf{z}^{(m)}|\mathbf{x})}{\partial \phi} [\log p_\theta(\mathbf{x}, \mathbf{z}^{(m)}) - \log q_\phi(\mathbf{z}^{(m)}|\mathbf{x})] \\ &\quad - \frac{1}{M} \sum_m \frac{\partial q_\phi(\mathbf{z}^{(m)}|\mathbf{x})}{\partial \phi} \cdot \frac{1}{q_\phi(\mathbf{z}^{(m)}|\mathbf{x})} \cdot q_\phi(\mathbf{z}^{(m)}|\mathbf{x}) \\ &= \frac{1}{M} \sum_m \frac{\partial \log q_\phi(\mathbf{z}^{(m)}|\mathbf{x})}{\partial \phi} (\log p_\theta(\mathbf{x}, \mathbf{z}^{(m)}) - \log q_\phi(\mathbf{z}^{(m)}|\mathbf{x})) \end{aligned} \quad (2.9)$$

Here, we achieve an unbiased gradient estimator, namely the likelihood ratio estimator or REINFORCE estimator. Compared to Equation 2.8, the gradients w.r.t. ϕ contain a potentially large scale term $\log p_\theta(\mathbf{x}, \mathbf{z}^{(m)}) - \log q_\phi(\mathbf{z}^{(m)}|\mathbf{x})$ which depends on the samples of \mathbf{z} . Since $\mathbf{z}^{(m)}$ are discrete samples, the estimates can be very high due to the scaling of the gradient inside the expectation, which results in a high-variance gradient estimator and might cause slow convergence during training. Thus, in order to mitigate the high-variance issue in this case, we introduce control variates method in next section.

2.2.2 Control Variates

The main idea of a control variate is to introduce an extra term that is analytically tractable in the expectation in Equation 2.6 and independent of the latent variable.

By subtracting the control variate c , the original expectation to be optimised remains unchanged:

$$\begin{aligned}
\mathcal{L} &= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}, \mathbf{z}) - \log q_\phi(\mathbf{z}|\mathbf{x}) - c] \\
&= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}, \mathbf{z}) - \log q_\phi(\mathbf{z}|\mathbf{x})] - \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[c] \\
&= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}, \mathbf{z}) - \log q_\phi(\mathbf{z}|\mathbf{x})] - c
\end{aligned} \tag{2.10}$$

but the variance of the gradients w.r.t. ϕ can be reduced:

$$\begin{aligned}
\frac{\partial L}{\partial \phi} &= \frac{1}{M} \sum_m \frac{\partial q_\phi(\mathbf{z}^{(m)}|\mathbf{x})}{\partial \phi} [\log p_\theta(\mathbf{x}, \mathbf{z}^{(m)}) - \log q_\phi(\mathbf{z}^{(m)}|\mathbf{x}) - c] \\
&\quad - \frac{1}{M} \sum_m \frac{\partial \log q_\phi(\mathbf{z}^{(m)}|\mathbf{x})}{\partial \phi} \cdot q_\phi(\mathbf{z}^{(m)}|\mathbf{x}) \\
&= \frac{1}{M} \sum_m q_\phi(\mathbf{z}^{(m)}|\mathbf{x}) \cdot \frac{\partial \log q_\phi(\mathbf{z}^{(m)}|\mathbf{x})}{\partial \phi} [\log p_\theta(\mathbf{x}, \mathbf{z}^{(m)}) - \log q_\phi(\mathbf{z}^{(m)}|\mathbf{x}) - c] \\
&\quad - \frac{1}{M} \sum_m \frac{\partial q_\phi(\mathbf{z}^{(m)}|\mathbf{x})}{\partial \phi} \cdot \frac{1}{q_\phi(\mathbf{z}^{(m)}|\mathbf{x})} \cdot q_\phi(\mathbf{z}^{(m)}|\mathbf{x}) \\
&= \frac{1}{M} \sum_m \frac{\partial \log q_\phi(\mathbf{z}^{(m)}|\mathbf{x})}{\partial \phi} (\log p_\theta(\mathbf{x}, \mathbf{z}^{(m)}) - \log q_\phi(\mathbf{z}^{(m)}|\mathbf{x}) - c)
\end{aligned} \tag{2.11}$$

where the control variate c is directly added into the scaling term in the expectation. Compared to Equation 2.9, c is able to suppress the potentially large gradients, which yields to a lower variance estimator. Generally, we could introduce a **moving-average baseline** c_0 as the simplest control variate. Since c_0 is constant during every epoch, its contribution to the gradient estimate is always zero in expectation. And the original expectation to be optimised is unchanged, so it maintains an unbiased gradient estimator.

A more effective control variate is the **input-dependent baseline** $c(\mathbf{x})$ proposed by Mnih & Gregor [77]. Since $c(\mathbf{x})$ is only dependent on the input observation, it is still analytically tractable from the expectation in the lower bound. In practice, it can be implemented by a neural network and trained jointly with the model to minimize the expected square of the moving-average baseline c :

$$\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[(l(\mathbf{z}, \mathbf{x}) - c_0 - c(\mathbf{x}))^2] \tag{2.12}$$

where $l(\mathbf{z}, \mathbf{x}) = \log p_\theta(\mathbf{x}, \mathbf{z}) - \log q_\phi(\mathbf{z}|\mathbf{x})$ is the original learning signal, c_0 captures the moving average and $c(\mathbf{x})$ learns the systematic differences in the learning signal

for different observations. Hence, it makes the update more adapted to the input observations \mathbf{x} , and can be easily applied and customised for different discrete latent variable models implemented by neural networks.

2.2.3 Advanced Sampling Schemes

In addition to the control variate approach, there are several other methods for mitigating the high-variance issue of sampling-based neural variational inference. MuProp [31] extends the likelihood ratio estimator with control variates by first-order Taylor expansion of a mean-field network, which is an unbiased gradient estimator for stochastic networks and improves the performance of neural variational inference. Intuitively, generating more samples for the gradient estimator is able to straightforwardly reduce the variance, but due to the computational expense we ought to make full use of the limited number of samples. Importance weighted autoencoders [15] provide a tighter log-likelihood lower bound for the continuous variational autoencoder derived from multi-sample importance weighting, and VIMCO (Variational Inference for Monte Carlo Objectives) [78] further extends the importance sampling approach to discrete latent variables and introduces an unbiased gradient estimator designed for importance-sampled objectives.

2.3 Parameterising Categorical Distribution

Generally, the learning of continuous and discrete latent variables are separated since variational inference applies different gradient estimators (e.g. Gaussian reparameterisation and REINFORCE algorithm). From the perspective of neural variational inference, continuous latent variables are commonly preferred over discrete latent variables since the Gaussian reparameterisation suffers less from the high-variance problem than REINFORCE algorithm, though there exist a few techniques, e.g. control variate for mitigating the issue. However, we are able to parameterise the categorical distributions by continuous latent variables. Thus in some cases where the computational expense allows integrating out discrete latent variables, we are able to directly apply the Gaussian reparameterisation for carrying out the inference.

In this section, we are going to introduce several approaches for parameterising categorical distributions, including Gaussian Softmax Construction (GSM), Gaussian Stick-breaking Construction (GSB) and Recurrent Stick-breaking Construction (RSB). It is worth mentioning that, by using the stick-breaking construction, we are able to introduce Bayesian non-parametrics for deep neural networks.

2.3.1 Gaussian Softmax Construction

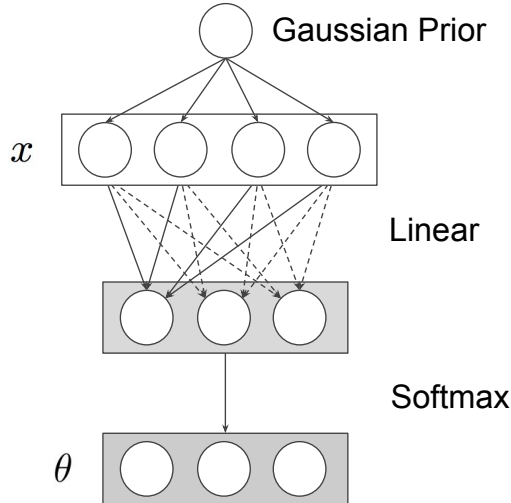


Figure 2.2: Neural Structure of GSM

In deep learning, an energy-based function is generally used to construct probability distributions [65]. Here we pass a Gaussian random vector through a softmax function to parameterise the categorical distribution, and $\theta \sim G_{\text{GSM}}(\mu_0, \sigma_0^2)$ is defined as:

$$x \sim \mathcal{N}(\mu_0, \sigma_0^2)$$

$$\theta = \text{softmax}(W_1^T x)$$

where W_1 is a linear transformation, and we leave out the bias terms for brevity. μ_0 and σ_0^2 are hyper-parameters which we set for a zero mean and unit variance Gaussian. Then, we easily construct a categorical distribution parameterised by a Gaussian, and the neural structure is shown in Figure 2.2. Here, the Gaussian Softmax construction can be considered as parameterised logistic normal distribution, which is similar to the prior distribution applied in correlated topic models [10].

2.3.2 Gaussian Stick-breaking Construction

In Bayesian non-parametrics, the stick-breaking process [90] is used as a constructive definition of the Dirichlet process, where sequentially drawn Beta random variables define breaks from a unit stick. In our case, following [51], we transform the modelling of multinomial probability parameters into the modelling of the logits of binomial probability parameters using Gaussian latent variables. More specifically, conditioned

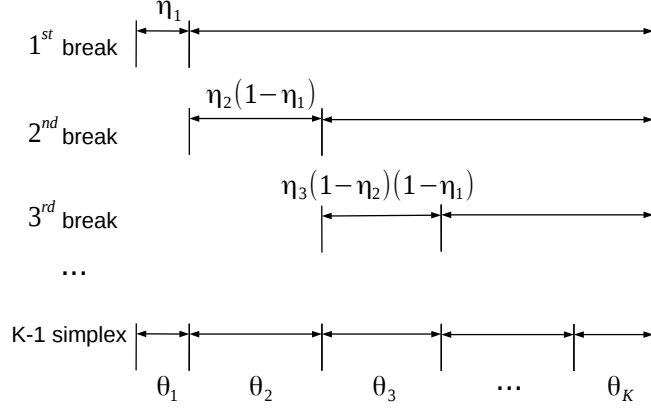


Figure 2.3: The Stick-breaking Construction.

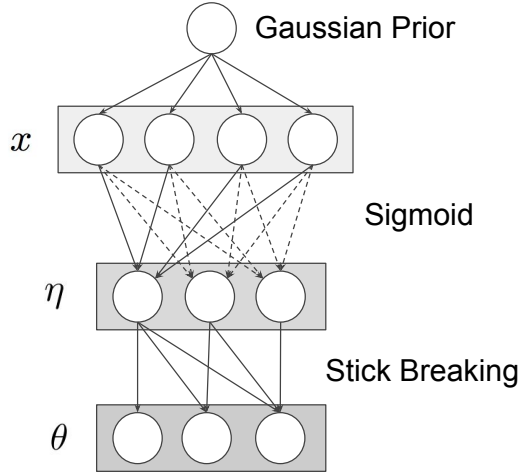


Figure 2.4: Neural Structure of GSB

on a Gaussian sample $x \in \mathbb{R}^H$, the breaking proportions $\eta \in \mathbb{R}^{K-1}$ are generated by applying the sigmoid function $\eta = \text{sigmoid}(W_2^T x)$ where $W \in \mathbb{R}^{H \times K-1}$. Starting with the first piece of the stick, the probability of the first category is modelled as a break of proportion η_1 , while the length of the remainder of the stick is left for the next break. Thus each dimension can be deterministically computed by $\theta_k = \eta_k \prod_{i=1}^{k-1} (1 - \eta_i)$ until $k=K-1$, and the remaining length is taken as the probability of the K th category $\theta_K = \prod_{i=1}^{K-1} (1 - \eta_i)$.

For instance assume $K = 3$, θ is generated by 2 breaks where $\theta_1 = \eta_1$, $\theta_2 = \eta_2(1 - \eta_1)$ and the remaining stick $\theta_3 = (1 - \eta_2)(1 - \eta_1)$. If the model proceeds to break the stick for $K = 4$, the remaining stick θ_3 is broken into (θ'_3, θ'_4) , where $\theta'_3 = \eta_3 \cdot \theta_3$, $\theta'_4 = (1 - \eta_3) \cdot \theta_3$ and $\theta_3 = \theta'_3 + \theta'_4$. Hence, for different values of K , it always satisfies $\sum_{k=1}^K \theta_k = 1$. The stick-breaking construction $f_{\text{SB}}(\eta)$ is illustrated in

Figure 2.3 and the distribution $\theta \sim G_{\text{GSB}}(\mu_0, \sigma_0^2)$ is defined as:

$$\begin{aligned} x &\sim \mathcal{N}(\mu_0, \sigma_0^2) \\ \eta &= \text{sigmoid}(W_2^T x) \\ \theta &= f_{\text{SB}}(\eta) \end{aligned}$$

Compared to the stick-breaking of Dirichlet process which has exchangeability (viz. the probability of generating a sequence of draws $\theta_1, \theta_2, \dots, \theta_k$ equals the probability of drawing them in any alternative order) and is invariant to size-biased permutations (viz. the fragments $\eta_1, \eta_2, \dots, \eta_k$ for constructing θ are independent samples from an identical distribution e.g. a Beta distribution), Gaussian stick-breaking construction maintains exchangeability but is not invariant to size-biased permutations due to that the parameters W_2 of G_{GSB} are attached to a fixed order of θ . Hence, the fragments $\eta_1, \eta_2, \dots, \eta_k$ are not independent and identically distributed. Despite of this, the Gaussian stick-breaking construction provides a more amenable form for neural variational inference. More interestingly, this stick-breaking construction introduces a non-parametric aspect to deep generative models with Gaussian as latent distribution, and the neural structure is shown in Figure 2.4.

2.3.3 Recurrent Stick-breaking Construction

Recurrent Neural Networks (RNN) are commonly used for modelling sequences of inputs in deep learning. Here we consider the stick-breaking construction as a sequential draw from an RNN, thus capturing an unbounded number of breaks with a finite number of parameters. Conditioned on a Gaussian latent variable x , the recurrent neural network $f_{\text{SB}}(x)$ produces a sequence of binomial logits which are used to break the stick sequentially. The $f_{\text{RNN}}(x)$ is decomposed as:

$$\begin{aligned} h_k &= \text{RNN}_{\text{SB}}(h_{k-1}) \\ \eta_k &= \text{sigmoid}(h_{k-1}^T x) \end{aligned}$$

where h_k is the output of the k th state, which we feed into the next state of the RNN_{SB} as an input. Figure 2.5 shows the recurrent neural network structure. Now $\theta \sim G_{\text{RSB}}(\mu_0, \sigma_0^2)$ is defined as:

$$\begin{aligned} x &\sim \mathcal{N}(\mu_0, \sigma_0^2) \\ \eta &= f_{\text{RNN}}(x) \\ \theta &= f_{\text{SB}}(\eta) \end{aligned}$$

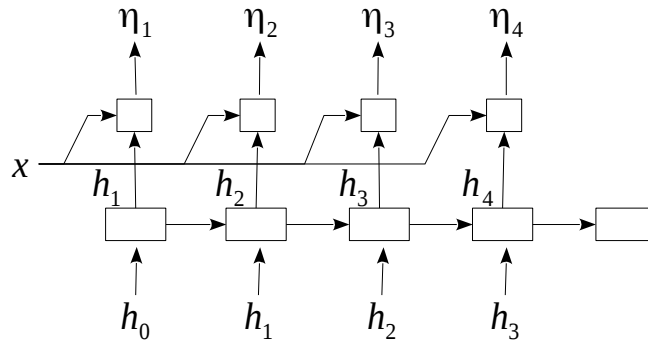


Figure 2.5: The unrolled RNN for breaking proportions η .

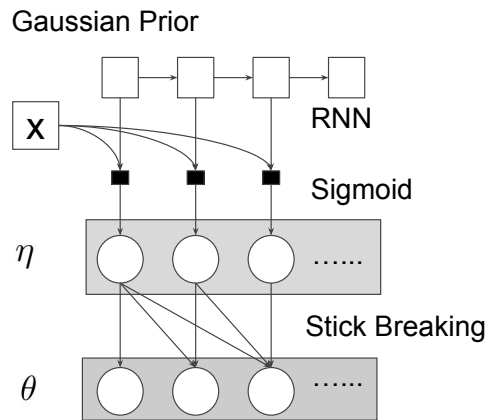


Figure 2.6: Neural Structure of RSB

where $f_{\text{SB}}(\eta)$ is equivalent to the stick-breaking function used in the Gaussian stick-breaking construction. Here, the RNN is able to dynamically produce new logits to break the stick *ad infinitum*. The expressive power of the RNN to model sequences of unbounded length is still bounded by the parametric model’s capacity, but for topic modelling it is adequate to model the countably infinite topics amongst the documents in a truncation-free fashion. The neural structure is shown in Figure 2.6.

2.4 Summary

This chapter introduces a generic neural variational inference framework for deep generative models. For the models with continuous latent variables, Gaussian reparameterisation trick can be easily applied to produce low-variance gradients to jointly update the inference network and the generative model. While for the models with discrete latent variables, REINFORCE algorithm with control variates is employed to construct the gradient estimator. Though there are several advanced sampling

schemes to further mitigate the high variance issue, it is still a major problem in the neural variational inference for discrete variables. Therefore, continuous latent variable models are generally more welcome than discrete ones from the perspective of optimisation. However, despite of the high variance issue, discrete representation is a natural fit for predictive systems with categorical properties, such as structure prediction, planning and clustering. Thus we are interested in exploring the potential of discrete latent variables and develop more sophisticated deep generative models for NLP.

Alternatively, in some cases where the marginalisation of discrete latent variable is tractable, we could use neural networks based on Gaussian draws to parameterise the discrete latent distribution. Then the model is able to not only make use of the discrete latent distribution, but also enjoy the efficiency and low-variance estimators brought by Gaussian reparameterisation trick. Moreover, with the help of the stick-breaking constructions, an unbounded categorical distribution is able to be incorporated into the deep generative models, which brings Bayesian non-parametrics and can easily updated by stochastic gradient back-propagation.

More details about the related deep generative models (continuous or discrete) for NLP will be introduced in the following chapters of this thesis.

Part II

Continuous Latent Variable Models

Chapter 3

Neural Document Models

This chapter introduces the first application of this thesis, neural variational document model (NVDM), which aims at learning continuous stochastic representations of documents in unsupervised fashion. The model is an instantiation of variational auto-encoders based on bag-of-words documents, which is simple, expressive and easy to train. In the experiments section, we also show that the neural variational document model is able to discover latent semantics of documents that can be interpreted as topics.

3.1 Introduction

Document modelling is a classical NLP task, which is useful for document classification, document clustering, information retrieval and knowledge discovery. Here, we are interested in unsupervised document modelling, which requires no document labels or categorical information. Generally, unsupervised document modelling is carried out by language models or bag-of-words models. For the approach of language modelling, a document is generated by autoregressively predicting the words as a long sequence. While for bag-of-words models, a document is modelled as a collection of words, where the syntactic information and grammars are not considered in this case. Normally, documents are preprocessed by filtering “stopwords” (i.e. the most common words in a language that have little contribution to document modelling are removed). In this chapter, the documents models discussed and compared are all bag-of-words document models.

A simple and effective way to model documents is using term frequency-inverse document frequency (tf-idf) score, which reflects how important a word is to a document in a collection or corpus. However, tf-idf models have no interpretation on topics and they are not scalable due to the sparse tf-idf matrix. In contrast, topic

models discovers the abstract topics that occur in a collection of documents so that all the documents can be modelled as a mixture of topics. Starting with latent semantic analysis (LSA [60]), models for uncovering the underlying semantic structure of a document collection have been widely applied in data mining, text processing and information retrieval. Probabilistic topic models (e.g. PLSA [45], LDA [11] and HDPs [102]) provide a robust, scalable, and theoretically sound foundation for document modelling by introducing latent variables for each token to topic assignment. LDA, as the representative of probabilistic topic models, has been widely applied for dimensionality reduction, text-mining and semantic interpretation. Beyond LDA, significant extensions have sought to capture topic correlations [10], topic evolution [9] and discover an unbounded number of topics [102]. Topic models have also been extended to capture extra context information such as time [111], authorship [85], and class labels [73]. Such extensions often require carefully tailored graphical models, and associated inference algorithms, to capture the desired context.

With the recent advances of deep neural networks, neural document models become more and more popular due to highly scalable vector representations and easy-to-implement network structures. The simplest bag-of-words document model implemented by neural networks is the bag-of-embeddings, which sums the word embeddings with semantic information rather than sparse one-hot word representations. However, the off-the-shelf word representations limits the scalability of the model. More recent neural document models are the Replicated Softmax [40]: an undirected topic model implemented by restricted Boltzmann machines; and the Sigmoid Belief Document Model [77] that applies sigmoid belief networks for modelling the semantics of documents. DocNADE [61] extends the Replicated Softmax model and estimates the probability of observing a new word in a given document given the previously observed words. DARN [29] employs a similar assumption but applies neural variational inference to infer the latent semantic information. Though these neural generative models do not aim at modelling topic distributions directly, compared to conventional probabilistic topic models [45, 11], they have achieved decent document perplexities. While the Paragraph Vector model [63] implements the idea in an alternative deterministic way, which directly concatenates the document/paragraph vector with the word vector to model whole documents like a language model.

This chapter introduces our neural variational document model (NVDM). Different from traditional probabilistic topic models, NVDM does not model topics explicitly, but it applies an easy-to-implement neural structure to discover continuous latent semantics of documents as a generative model, where the stochastic vectors (latent

semantics) can be interpreted as topics. Compared to the neural models mentioned above using binary vectors to model semantic, the continuous stochastic representations of documents are more expressive and easier to train.

3.2 Neural Variational Document Model (NVDM)

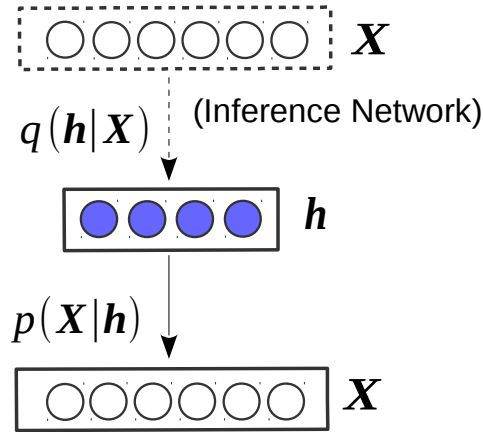


Figure 3.1: Neural Variational Document Model.

The Neural Variational Document Model (Figure 3.1) is an unsupervised generative model of text which aims to extract a continuous semantic latent variable for each document. This model can be interpreted as a variational auto-encoder: an MLP encoder (inference network) compresses the bag-of-words document representation into a continuous latent distribution, and a softmax decoder (generative model) reconstructs the document by generating the words independently. A primary feature of NVDM is that each word is generated directly from a dense continuous document representation instead of the more common binary semantic vector.

Assume a continuous hidden variable $\mathbf{h} \in \mathbb{R}^K$ that generates all the words in a document independently is introduced to represent its semantic content. Let $\mathbf{X} \in \mathbb{R}^{|V|}$ be the bag-of-words representation of a document and $\mathbf{x}_i \in \mathbb{R}^{|V|}$ be the one-hot representation of the word at position i . We construct the inference network $q(\mathbf{h}|\mathbf{X})$ to model the variational distribution that maps \mathbf{X} to \mathbf{h} , and the reconstruction model $p(\mathbf{X}|\mathbf{h}) = \prod_{i=1}^N p(\mathbf{x}_i|\mathbf{h})$ to generate $\{\mathbf{x}_i\}$ from \mathbf{h} . To maximise the log-likelihood $\log \sum_{\mathbf{h}} p(\mathbf{X}|\mathbf{h})p(\mathbf{h})$ of documents, we derive the lower bound:

$$\mathcal{L} = \mathbb{E}_{q_{\phi}(\mathbf{h}|\mathbf{X})} \left[\sum_{i=1}^N \log p_{\theta}(\mathbf{x}_i|\mathbf{h}) \right] - D_{\text{KL}}[q_{\phi}(\mathbf{h}|\mathbf{X})||p(\mathbf{h})] \quad (3.1)$$

where N is the number of words in the document and $p(\mathbf{h})$ is a Gaussian prior for \mathbf{h} . Here, we consider N is observed for all the documents. The conditional probability over words $p_\theta(\mathbf{x}_i|\mathbf{h})$ (decoder) is modelled by multinomial logistic regression and shared across documents:

$$p_\theta(\mathbf{x}_i|\mathbf{h}) = \frac{\exp\{-E(\mathbf{x}_i; \mathbf{h}, \theta)\}}{\sum_{j=1}^{|\mathcal{V}|} \exp\{-E(\mathbf{x}_j; \mathbf{h}, \theta)\}} \quad (3.2)$$

$$E(\mathbf{x}_i; \mathbf{h}, \theta) = -\mathbf{h}^T \mathbf{R} \mathbf{x}_i - \mathbf{b}_{x_i} \quad (3.3)$$

where $\mathbf{R} \in \mathbb{R}^{K \times |\mathcal{V}|}$ learns the semantic word embeddings and \mathbf{b}_{x_i} represents the bias term.

As there is no supervision information for the latent semantics, \mathbf{h} , the posterior approximation $q_\phi(\mathbf{h}|\mathbf{X})$ is only conditioned on the current document \mathbf{X} . The inference network $q_\phi(\mathbf{h}|\mathbf{X}) = \mathcal{N}(\mathbf{h}|\boldsymbol{\mu}(\mathbf{X}), \boldsymbol{\sigma}^2(\mathbf{X}))$ is modelled as:

$$\boldsymbol{\pi} = g(f_X^{\text{MLP}}(\mathbf{X})) \quad (3.4)$$

$$\boldsymbol{\mu} = l_1(\boldsymbol{\pi}), \log \boldsymbol{\sigma} = l_2(\boldsymbol{\pi}) \quad (3.5)$$

For each document \mathbf{X} , the neural network generates its own parameters $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ that parameterise the latent distribution over document semantics \mathbf{h} . Based on the samples $\mathbf{h} \sim q_\phi(\mathbf{h}|\mathbf{X})$, the lower bound (Eq. 3.1) can be optimised by back-propagating the stochastic gradients w.r.t. θ and ϕ .

Since $p(\mathbf{h})$ is a standard Gaussian prior, the Gaussian KL-Divergence between the prior and variational distribution $D_{\text{KL}}[q_\phi(\mathbf{h}|\mathbf{X})||p(\mathbf{h})]$ can be computed analytically to further lower the variance of the gradients:

$$D_{\text{KL}} = -\frac{1}{2}(K - \|\boldsymbol{\mu}\|^2 - \|\boldsymbol{\sigma}\|^2 + \log |\text{diag}(\boldsymbol{\sigma}^2)|) \quad (3.6)$$

Moreover, it also acts as a regulariser for updating the parameters of the inference network $q_\phi(\mathbf{h}|\mathbf{X})$.

3.3 Experiments

3.3.1 Dataset & Setup

We experiment with NVDM on two standard news corpora: the *20NewsGroups*¹ and the Reuters *RCV1-v2*². The former is a collection of newsgroup documents, consisting

¹<http://qwone.com/~jason/20Newsgroups>

²<http://trec.nist.gov/data/reuters/reuters.html>

of 11,314 training and 7,531 test articles. The latter is a large collection from Reuters newswire stories with 794,414 training and 10,000 test cases. The vocabulary size of these two datasets are set as 2,000 and 10,000.

To make a direct comparison with the prior work we follow the same preprocessing procedure and setup as [40], [61], [97], and [77]. We train NVDM models with 50 and 200 dimensional document representations respectively. For the inference network, we use an MLP (Eq. 3.4) with 2 layers and 500 dimension rectifier linear units, which converts document representations into embeddings. During training we carry out stochastic estimation by taking one sample for estimating the stochastic gradients, while in prediction we use 20 samples for predicting document perplexity (see the related discussion for Figure 5.3). The model is trained by Adam [53] and tuned by hold-out validation perplexity.

We alternately optimise the generative model and the inference network by fixing the parameters of one while updating the parameters of the other. This is an important optimisation trick to achieving a good performance of a bag-of-words variational auto-encoder. When the generative model is fixed, it focuses on learning the inference network (encoder) for modelling stochastic document representations and fitting the prior distribution. While when the inference network is fixed, it aims at learning the generative model (decoder) to optimise the reconstruction error. In this way, the model is encouraged to make use of the continuous latent variable for generating the observations, which helps address the optimization challenge of VAEs. The sequence VAE [14] applies KL cost annealing and word dropout for dealing with the similar issue. However, sequence generation generally relies on powerful autoregressive decoders such as LSTMs, which are even more difficult to force the generative model to use the latent variable compared to the simple softmax decoder applied in our case.

3.3.2 Evaluation on Perplexity

Table 3.1 presents the test document perplexity. The first column lists the models, and the second column shows the dimension of latent variables used in the experiments. The final two columns present the perplexity achieved by each topic model on the *20NewsGroups* and *RCV1-v2* datasets. In document modelling, perplexity is computed by $\exp(-\frac{1}{D} \sum_n \frac{1}{N_d} \log p(\mathbf{X}_d))$, where D is the number of documents, N_d represents the length of the d th document and $\log p(\mathbf{X}) = \log \int p(\mathbf{X}|\mathbf{h})p(\mathbf{h})d\mathbf{h}$ is the log probability of the words in the document. Since $\log p(\mathbf{X})$ is intractable in the NVDM, we use the estimated variational lower bound (which is an upper bound on perplexity) to compute the perplexity following [77].

Model	Dim	20News	RCV1
LDA	50	1091	1437
LDA	200	1058	1142
RSM	50	953	988
docNADE	50	896	742
SBN	50	909	784
fDARN	50	917	724
fDARN	200	—	598
NVDM	50	836	563
NVDM	200	852	550

Table 3.1: Perplexity on test dataset.

For the benchmark models in Table 3.1, LDA [11] is a traditional topic model that models documents by mixtures of topics, RSM [40] is an undirected topic model implemented by restricted Boltzmann machines, and docNADE [61] is a neural topic model based on autoregressive assumption. The models based on Sigmoid Belief Networks (SBN) and Deep AutoRegressive Neural Network (DARN) structures are implemented by Mnih and Gregor [77], which employs an MLP to build a Monte Carlo control variate estimator for stochastic estimation.

While all the baseline models listed in Table 3.1 apply discrete latent variables, here NVDM employs a continuous stochastic document representation. The experimental results indicate that NVDM achieves the best performance on both datasets. For the experiments on *RCV1-v2* dataset, the NVDM with latent variable of 50 dimension performs even better than the fDARN with 200 dimension. It demonstrates that our document model with continuous latent variables has higher expressiveness and better generalisation ability.

3.3.3 Interpreting Document Semantics

In addition to the perplexities, we also qualitatively evaluate the semantic information learned by NVDM on the *20NewsGroups* dataset. Although NVDM directly generates words conditioned on the document semantics \mathbf{h} and topics are not explicitly modelled, we use the positive connections between topics and words for interpreting the discovered topics, which are the parameters \mathbf{R} . For example in Figure 3.2, we assume each dimension of \mathbf{h} represents a topic of document collections. The words related to a specific topic can be achieved by ranking the words with the weights connecting all the words in the vocabulary \mathbf{X} and a specific dimension of \mathbf{h} . Then, we are able to interpret the document semantics as topics through the top- n words.

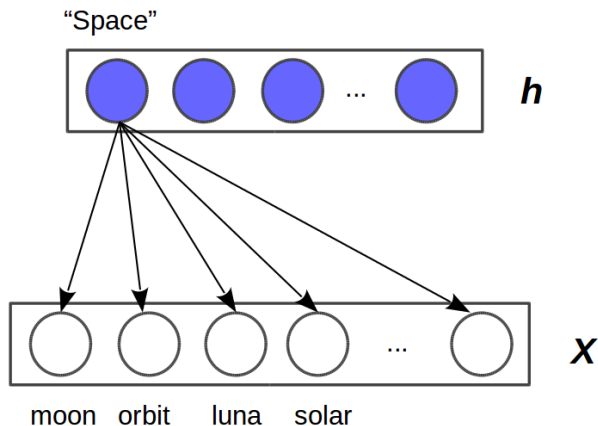


Figure 3.2: Interpreting document semantics as topics.

<i>Space</i>	<i>Religion</i>	<i>Encryption</i>	<i>Sport</i>	<i>Policy</i>
orbit	muslims	rsa	goals	bush
lunar	worship	cryptography	pts	resources
solar	belief	crypto	teams	charles
shuttle	genocide	keys	league	austin
moon	jews	pgp	team	bill
launch	islam	license	players	resolution
fuel	christianity	secure	nhl	mr
nasa	atheists	key	stats	misc
satellite	muslim	escrow	min	piece
japanese	religious	trust	buf	marc

Table 3.2: The topics learned by NVDM on 20News.

This is a drawback of NVDM when trying to interpret the neural document model as a topic model, however in next section, we will address this issue by neural topic models, where documents are modelled by a mixture of topics and each topic is modelled by multinomial distribution over words.

Table 3.2 presents 5 randomly selected topics with 10 words that have the strongest positive connection with the topic. Based on the words in each column, we can deduce their corresponding topics as: *Space*, *Religion*, *Encryption*, *Sport* and *Policy*. Although the model does not impose independent interpretability on the latent representation dimensions, we still see that the NVDM learns locally interpretable structure. Besides, Table 3.3 compares the 5 nearest words selected according to the semantic word embeddings \mathbf{R} in Equation 3.3 learned from NVDM and docNADE.

Word	weapons	medical	companies	define	israel	book
NVDM	guns	medicine	expensive	defined	israeli	books
	weapon	health	industry	definition	arab	reference
	gun	treatment	company	printf	arabs	guide
	militia	disease	market	int	lebanon	writing
	armed	patients	buy	sufficient	lebanese	pages
NADE	weapon	treatment	demand	defined	israeli	reading
	shooting	medecine	commercial	definition	israelis	read
	firearms	patients	agency	refer	arab	books
	assault	process	company	make	palestinian	relevent
	armed	studies	credit	examples	arabs	collection

Table 3.3: The five nearest words in the semantic space.

3.4 Summary

This chapter describes the proposed neural variational document model, which is a simple, expressive and easy-to-train instantiation of variational auto-encoders for NLP. Though NVDM only applies bag-of-words information to model documents, both the quantitative and qualitative evaluation demonstrate the effectiveness and efficiency of this deep generative model. Certainly, there is a drawback of interpreting the latent semantics of documents as topics, which are not modelled explicitly as distributions over words in NVDM. Thus, in the next chapter we are going to introduce neural topic models where this issue will be addressed.

Chapter 4

Neural Topic Models

This chapter discusses the deep generative models for topic modelling. Different from the NVDM introduced in the previous chapter, the neural topic models employ multinomial to model topic-word distributions explicitly, which is more similar to the generative story of traditional probabilistic topic models. In this context, we parameterise the categorical distributions by the Gaussian softmax construction (GSM), Gaussian stick-breaking construction (GSB) and Recurrent stick-breaking construction (RSB) that are introduced in Chapter 2. More interestingly, Bayesian non-parametrics can be incorporated into the neural topic model with the help of the stick-breaking constructions, and the inference can be easily carried out in the framework of neural variational inference. The following contents will be divided into neural finite topic models and neural unbounded topic models where the modelled topics are bounded and unbounded respectively.

4.1 Introduction

For the traditional Dirichlet-Multinomial topic model, efficient inference is available by exploiting conjugacy with either Monte Carlo or variational techniques [47, 2, 7]). However, as topic models have grown more expressive, in order to capture topic dependencies or exploit conditional information, inference methods have become increasingly complex. This is especially apparent for non-conjugate models [17, 10, 107]. As mentioned in Chapter 2, deep neural networks are excellent function approximators and have shown great potential for learning complicated non-linear distributions for unsupervised models. The inference network, as a black-box variational distribution parameterised by a neural network, approximates the posterior of a generative model. This allows both the generative model and the variational network to be jointly trained with backpropagation. Hence, in the framework of neural variational

inference, non-conjugacy is no longer a bottleneck for the probabilistic models, and we are allowed to have more possibilities on designing complicated and interesting deep generative models.

Here, we propose and evaluate a range of topic models parameterised with neural networks and trained with variational inference. We employ three different neural structures for constructing topic distributions introduced in Chapter 2: Gaussian softmax construction (GSM), Gaussian stick-breaking distribution construction (GSB), and recurrent stick-breaking construction (RSB), all of which are conditioned on a draw from a multivariate Gaussian distribution. The Gaussian softmax topic model constructs a finite topic distribution with a softmax function applied to the projection of the Gaussian random vector. The Gaussian stick-breaking model also constructs a discrete distribution from the Gaussian draw, but this time employing a stick-breaking construction to provide a bias towards sparse topic distributions. And the recurrent stick-breaking construction employs a recurrent neural network, again conditioned on the Gaussian draw, to progressively break the stick, yielding a neural analog of a Dirichlet Process topic model [102].

Our neural topic models combine the merits of both neural networks and traditional probabilistic topic models. They can be trained efficiently by backpropagation, scaled to large data sets, and easily conditioned on any available contextual information. Further, as probabilistic graphical models, they are interpretable and explicitly representing the dependencies amongst the random variables. Previous neural document models, such as the neural variational document model (NVDM) introduced in the previous chapter, belief networks document model [77], neural auto-regressive document model [61] and replicated softmax [40], have not explicitly modelled latent topics. Through evaluations on a range of data sets we compare our models with previously proposed neural document models and traditional probabilistic topic models, demonstrating their robustness and effectiveness.

4.2 Neural Finite Topic Model

This section discusses the neural finite topic model. Following the generative story of topic model, we first describe the parameterisation of the document-topic distribution. Then, the neural networks for parameterising the topic-word distribution will be introduced. Finally, we derive the variational lower bound that integrates out latent topics. In addition, the connections and differences between neural topic models and neural document models will be discussed.

4.2.1 Parameterising the Document-Topic Distribution

In probabilistic topic models, such as LDA [11], we use the latent variables θ_d and z_n for the topic proportion of document d , and the topic assignment for the observed word w_n , respectively. In order to facilitate efficient inference, the Dirichlet distribution (or Dirichlet process [102]) is employed as the prior to generate the parameters of the multinomial distribution θ_d for each document. The use of a conjugate prior allows the tractable computation of the posterior distribution over the latent variables' values. While alternatives have been explored, such as log-normal topic distributions [9, 10], extra approximation (e.g. the Laplace approximation [107]) is required for closed form derivations. The generative process of LDA is:

$$\begin{aligned}\theta_d &\sim \text{Dir}(\alpha_0), & \text{for } d \in D \\ z_n &\sim \text{Multi}(\theta_d), & \text{for } n \in [1, N_d] \\ w_n &\sim \text{Multi}(\beta_{z_n}), & \text{for } n \in [1, N_d]\end{aligned}$$

where β_{z_n} represents the topic distribution over words given topic assignment z_n and N_d is the number of tokens in document d . β_{z_n} can be drawn from another Dirichlet distribution, but here we consider it a model parameter. α_0 is the hyper-parameter of the Dirichlet prior and N_d is the total number of words in document d . The marginal likelihood for a document in collection D is:

$$p(d|\alpha_0, \beta) = \int_{\theta} p(\theta|\alpha_0) \prod_n \sum_{z_n} p(w_n|\beta_{z_n}) p(z_n|\theta) d\theta. \quad (4.1)$$

If we employ mean-field variational inference, the updates for the variational parameters $q(\theta)$ and $q(z_n)$ can be directly derived in closed form.

In contrast, our proposed models introduce a neural network to parameterise the multinomial topic distribution. The generative process is:

$$\begin{aligned}\theta_d &\sim G(\mu_0, \sigma_0^2), & \text{for } d \in D \\ z_n &\sim \text{Multi}(\theta_d), & \text{for } n \in [1, N_d] \\ w_n &\sim \text{Multi}(\beta_{z_n}), & \text{for } n \in [1, N_d]\end{aligned}$$

where $G(\mu_0, \sigma_0^2)$ is composed of a neural network $\theta = g(x)$ conditioned on a diagonal Gaussian $x \sim N(\mu_0, \sigma_0^2)$. The marginal likelihood is:

$$p(d|\mu_0, \sigma_0, \beta) = \int_{\theta} p(\theta|\mu_0, \sigma_0^2) \prod_n \sum_{z_n} p(w_n|\beta_{z_n}) p(z_n|\theta) d\theta. \quad (4.2)$$

Compared to Equation (4.1), here we parameterise the latent variable θ by a neural network conditioned on a draw from a Gaussian distribution. To carry out neural variational inference [75], we construct an inference network $q(\theta|\mu(d),\sigma(d))$ to approximate the posterior $p(\theta|d)$, where $\mu(d)$ and $\sigma(d)$ are functions of d that are implemented by multilayer perceptrons (MLP). By using a Gaussian prior distribution, we are able to employ the re-parameterisation trick [56] to build an unbiased and low-variance gradient estimator for the variational distribution. Without conjugacy, the updates of the parameters can still be derived directly and easily from the variational lower bound. We defer discussion of the inference process until the next section. Here we introduce several alternative neural networks for $g(x)$ which transform a Gaussian sample x into the topic proportions θ .

4.2.2 Parameterising the Topic-Word Distribution

Assume we have finite number of topics K , the topic distribution over words given a topic assignment z_n is $p(w_n|\beta, z_n) = \text{Multi}(\beta_{z_n})$, and β_{z_n} is the z_n topic-word distribution. Here we introduce topic vectors $t \in \mathbb{R}^{K \times H}$, word vectors $v \in \mathbb{R}^{V \times H}$ and generate the topic distributions over words by:

$$\beta_k = \text{softmax}(v \cdot t_k^T). \quad (4.3)$$

Therefore, $\beta \in \mathbb{R}^{K \times V}$ is a collection of simplexes achieved by computing the semantic similarity between topics and words.

A related work Gaussian-LDA [20] also attempts to explicitly model the representations of topics. It applies multivariate Gaussian distributions as topics to generate words, and each topics are drawn from conjugate priors, where a Gaussian centered at zero for the mean and an inverse Wishart distribution for the covariance. Compared to the topic-word distribution (Equation 4.3) that employs inner product distance, the probability of generating a word from a topic in Gaussian-LDA is determined by the Euclidean distance between the embeddings. The merit is that both topics and words in Gaussian-LDA share the same semantic space and the correlations of words are jointly modelled. While the drawback lies in the fact that the inference is difficult due to the high sampling complexity, even if the word embeddings are pretrained. On the contrary, in our models, the vector representations of both words and topics can be directly and easily updated by gradient back-propagation.

Following the notation introduced in Chapter 2, the prior distribution is defined as $G(\theta|\mu_0, \sigma_0^2)$ in which $x \sim \mathcal{N}(x|\mu_0, \sigma_0^2)$ and the projection network generates $\theta = g(x)$ for each document. Here, $g(x)$ can be the Gaussian Softmax $g_{\text{GSM}}(x)$, Gaussian

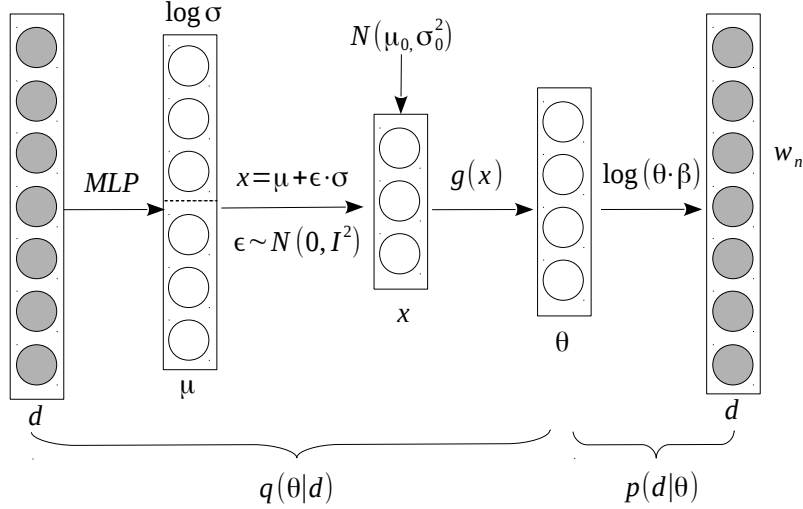


Figure 4.1: Network structure of the inference model $q(\theta | d)$, and of the generative model $p(d | \theta)$.

Stick Breaking $g_{\text{GSB}}(x)$, or Recurrent Stick Breaking $g_{\text{RSB}}(x)$ constructions with fixed length RNN_{SB} .

4.2.3 Integrating out Latent Topics

We derive a variational lower bound for the document log-likelihood according to Equation (4.2):

$$\mathcal{L}_d = \mathbb{E}_{q(\theta|d)} \left[\sum_{n=1}^N \log \sum_{z_n} [p(w_n | \beta_{z_n}) p(z_n | \theta)] \right] - D_{\text{KL}} [q(\theta|d) || p(\theta | \mu_0, \sigma_0^2)] \quad (4.4)$$

where $q(\theta|d)$ is the variational distribution approximating the true posterior $p(\theta|d)$. Following the framework of neural variational inference [75, 56, 84], we introduce an inference network conditioned on the observed document d to generate the variational parameters $\mu(d)$ and $\sigma(d)$ so that we can estimate the lower bound by sampling θ from $q(\theta|d) = G(\theta | \mu(d), \sigma^2(d))$. Here, we apply the reparameterisation trick $\hat{\theta} = \mu(d) + \hat{\epsilon} \cdot \sigma(d)$ and sample $\hat{\epsilon} \in \mathcal{N}(0, I)$.

Since the generative distribution $p(\theta | \mu_0, \sigma_0^2) = p(g(x) | \mu_0, \sigma_0^2) = p(x | \mu_0, \sigma_0^2)$ and the variational distribution $q(\theta|d) = q(g(x)|d) = q(x | \mu(d), \sigma^2(d))$, the KL term in Equation (4.4) can be easily integrated as a Gaussian KL-divergence. Note that, the parameterisation network $g(x)$ and its parameters are shared across all the documents. In addition, given a sampled $\hat{\theta}$, the latent variable z_n can be integrated out as:

$$\begin{aligned} \log p(w_n | \beta, \hat{\theta}) &= \log \sum_{z_n} [p(w_n | \beta_{z_n}) p(z_n | \hat{\theta})] \\ &= \log(\hat{\theta} \cdot \beta) \end{aligned} \quad (4.5)$$

Thus there is no need to introduce another variational approximation for the topic assignment z . The variational lower bound is therefore:

$$\mathcal{L}_d \approx \hat{\mathcal{L}}_d = \sum_{n=1}^N \left[\log p(w_n | \beta, \hat{\theta}) \right] - D_{KL} [q(x|d) || p(x)]$$

We can directly derive the gradients of the generative parameters Θ , including t , v and $g(x)$. While for the variational parameters Φ , including $\mu(d)$ and $\sigma(d)$, we use the gradient estimators:

$$\begin{aligned} \nabla_{\mu(d)} \hat{\mathcal{L}}_d &\approx \nabla_{\hat{\theta}} \hat{\mathcal{L}}_d, \\ \nabla_{\sigma(d)} \hat{\mathcal{L}}_d &\approx \hat{\epsilon} \cdot \nabla_{\hat{\theta}} \hat{\mathcal{L}}_d. \end{aligned}$$

Θ and Φ are jointly updated by stochastic gradient back-propagation. The structure of this variational auto-encoder is illustrated in Figure 4.1.

4.2.4 Topic Models vs. Document Models

In most topic models, documents are modelled by a mixture of topics, and each word is associated with a single topic latent variable, e.g. LDA and GSM. However, the neural variational document model (NVDM) introduced in Chapter 3 is implemented as a VAE [56] without modelling topics explicitly. The major difference is that NVDM employs a softmax decoder (Equation 4.6) to generate all of the words of a document conditioned on the document representation $\hat{\theta}$:

$$\log p(w_n | \beta, \hat{\theta}) = \log \text{softmax}(\hat{\theta} \cdot \beta). \quad (4.6)$$

where both $\hat{\theta}$ and β are unnormalised. Hence, it breaks the topic model assumption that each document consists of a mixture of topics. Although the latent variables can still be interpreted as topics, these topics are not modelled explicitly since there is no actual topic distribution over words. [96] interprets the above decoder as a weighted product of experts topic model, here however, we refer to such models that do not directly assign topics to words as document models instead of topic models. We can also convert our neural topic models to neural document models by replacing the mixture decoder (Equation 4.5) with the softmax decoder (Equation 4.6). For example in the GSM construction, if we remove the softmax function over θ , and directly apply Equation 4.6 to generate the words, it reduces to a variant of the NVDM model (GSM applies topic and word vectors to compute β , while NVDM directly models a projection β from the latent variables to words). The experimental evaluation in next section will have more detailed discussion comparing these models.

4.2.5 Topic Diversity

An issue that exists in both probabilistic and neural topic models is redundant topics. In neural models, however, we are able to straightforwardly regularise the distance between each of the topic vectors in order to diversify the topics. Following [120], we apply such topic diversity regularisation during the inference process. We compute the angles between each two topics $a(t_i, t_j) = \arccos(\frac{|t_i \cdot t_j|}{\|t_i\| \cdot \|t_j\|})$. Then, the mean angle of all pairs of K topics is $\zeta = \frac{1}{K^2} \sum_i \sum_j a(t_i, t_j)$, and the variance is $\nu = \frac{1}{K^2} \sum_i \sum_j (a(t_i, t_j) - \zeta)^2$. We add the following topic diversity regularisation to the variational objective:

$$\mathcal{J} = \mathcal{L} + \lambda(\zeta - \nu),$$

where λ is a hyper-parameter for the regularisation that is set as 0.1 in the experiments. During training, the mean angle is encouraged to be larger while the variance is suppressed to be smaller so that all of the topics will be pushed away from each other in the topic semantic space. Though in practice diversity regularisation does not provide a significant improvement to perplexity ($2 \sim 5$ in most cases), it helps reduce topic redundancy and can be easily applied on topic vectors instead of the simplex over the full vocabulary.

4.3 Neural Unbounded Topic Model

In this section, we further extend the neural topic model into neural unbounded topic model, where Bayesian non-parametrics are introduced with the help of stick-breaking construction. In this context, both Gaussian stick-breaking construction and recurrent stick-breaking construction are able to provide non-parametrics into deep generative models. As the recurrent stick-breaking construction applies recurrent neural networks that dynamically produce new logits to break the stick *ad infinitum*, it is able to model the countably infinite topics amongst the documents in a truncation-free fashion. While for Gaussian stick-breaking construction, we have to predefine a large enough number to be the threshold of generated topics during inference process, which is a truncated variational inference. Therefore, the neural unbounded topic model discussed in this chapter refers to the model implemented by the recurrent stick-breaking construction with truncation-free neural variational inference.

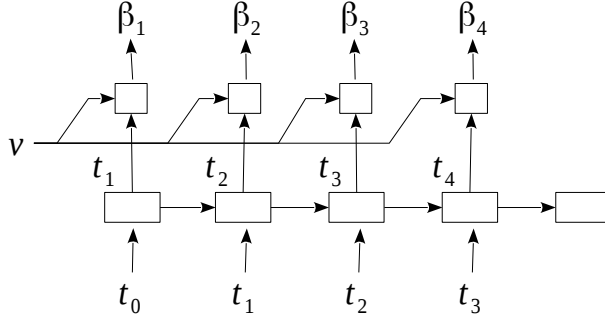


Figure 4.2: The unrolled RNN that produces the topic-word distributions β .

4.3.1 Unbounded Topics

The recurrent stick breaking process employs a recurrent neural network, conditioned on the Gaussian draw, to progressively break the stick, yielding a neural analog of a Dirichlet Process topic model [102]. Hence, in addition to the RNN_{SB} that generates the topic proportions $\theta \in \mathbb{R}^\infty$ for each document, we must introduce another neural network $\text{RNN}_{\text{Topic}}$ to produce the topics $t \in \mathbb{R}^{\infty \times H}$ dynamically, so as to avoid the need to truncate the variational inference.

For comparison, in finite neural topic models we have topic vectors $t \in \mathbb{R}^{K \times H}$, while in unbounded neural topic models the topics $t \in \mathbb{R}^{\infty \times H}$ are dynamically generated by $\text{RNN}_{\text{Topic}}$ and the order of the topics corresponds to the order of the states in RNN_{SB} . The generation of β follows:

$$\begin{aligned} t_k &= \text{RNN}_{\text{Topic}}(t_{k-1}), \\ \beta_k &= \text{softmax}(v \cdot t_k^T), \end{aligned}$$

where $v \in \mathbb{R}^{V \times H}$ represents the word vectors, t_k is the k th topic generated by $\text{RNN}_{\text{Topic}}$ and $k < \infty$. Figure 4.2 illustrates the neural structure of $\text{RNN}_{\text{Topic}}$.

4.3.2 Truncation-free Neural Variational Inference

For the unbounded topic models we introduce a truncation-free neural variational inference method which enables the model to dynamically decide the number of active topics. Assume the current active number of topics is i , $\text{RNN}_{\text{Topic}}$ generates $t^i \in \mathbb{R}^{i \times H}$ by an $i - 1$ step stick-breaking process (the logit for the n th topic is the remaining stick after $i - 1$ breaks). The variational lower bound for a document d is:

$$\mathcal{L}_d^i \approx \sum_{n=1}^N \left[\log p(w_n | \beta^i, \hat{\theta}^i) \right] - D_{KL} [q(x|d) || p(x)],$$

Algorithm 1 Unbounded Recurrent Neural Topic Model

```
0: Initialise  $\Theta$  and  $\Phi$ ; Set active topic number  $i$ 
1: repeat
2:   for  $s \in$  minibatches  $S$  do
3:     for  $k \in [1, i]$  do
4:       Compute topic vector  $t_k = \text{RNN}_{\text{Topic}}(t_{k-1})$ 
5:       Compute topic distribution  $\beta_k = \text{softmax}(v \cdot t_k^T)$ 
6:     end for
7:     for  $d \in D_s$  do
8:       Sample topic proportion  $\hat{\theta} \sim G_{\text{RSB}}(\theta | \mu(d), \sigma^2(d))$ 
9:       for  $w \in$  document  $d$  do
10:        Compute log-likelihood  $\log p(w | \hat{\theta}, \beta)$ 
11:      end for
12:      Compute lower bound  $\mathcal{L}_d^{i-1}$  and  $\mathcal{L}_d^i$ 
13:      Compute gradients  $\nabla_{\Theta, \Phi} \mathcal{L}_d^i$  and update
14:    end for
15:    Compute likelihood increase  $\mathcal{I}$ 
16:    if  $\mathcal{I} > \gamma$  then
17:      Increase active topic number  $i = i + 1$ 
18:    end if
19:  end for
20: until Convergence
```

where $\hat{\theta}^i$ corresponds to the topic distribution over words β^t . In order to dynamically increase the number of topics, the model proposes the i th break on the stick to split the $(i + 1)$ th topic. In this case, $\text{RNN}_{\text{Topic}}$ proceeds to the next state and generates topic $t^{(i+1)}$ for $\beta^{(i+1)}$ and the RNN_{SB} generates $\hat{\theta}^{(i+1)}$ by an extra break of the stick. Firstly, we compute the likelihood increase brought by topic i across the documents D :

$$\mathcal{I} = \sum_d^D [\mathcal{L}_d^i - \mathcal{L}_d^{i-1}] / \sum_d^D [\mathcal{L}_d^i]$$

Then, we employ an acceptance hyper-parameter γ to decide whether to generate a new topic. If $\mathcal{I} > \gamma$, the previous proposed new topic (the i th topic) contributes to the generation of words and we increase the active number of topics i by 1, otherwise we keep the current i unchanged. Thus γ controls the rate at which the model generates new topics. In practice, the increase of the lower bound is computed over mini-batches so that the model is able to generate new topics before the current epoch is finished. The details of the algorithm are described in Algorithm 1.

4.4 Experiments

We perform an experimental evaluation employing three datasets: *MXM*¹ song lyrics, *20NewsGroups*² and Reuters *RCV1-v2*³ news. *MXM* is the official lyrics collection of the Million Song Dataset with 210,519 training and 27,143 testing datapoints respectively. The *20NewsGroups* corpus is divided into 11,314 training and 7,531 testing documents, while the *RCV1-v2* corpus is a larger collection with 794,414 training and 10,000 test cases from Reuters newswire stories. We employ the original 5,000 vocabulary provided for *MXM*, while the other two datasets are processed by stemming, filtering stopwords and taking the most frequent 2,000⁴ and 10,000 words as the vocabularies.

The hidden dimension of the MLP for constructing $q(\theta|d)$ is 256 for all the neural topic models and the benchmarks that apply neural variational inference (e.g. NVDM, proLDA, NVLDA), and 0.8 dropout is applied on the output of the MLP before parameterising the diagonal Gaussian distribution. Grid search is carried out on learning rate and batch size for achieving the held-out perplexity. For the recurrent stick breaking construction we use a one layer LSTM cell (256 hidden units) for constructing the recurrent neural network. For the finite topic models we set the maximum number of topics K as 50 and 200. The models are trained by Adam [53] and only one sample is used for neural variational inference. We follow the optimisation strategy of NVDM by alternately updating the model parameters and the inference network. To alleviate the redundant topics issue, we also apply topic diversity regularisation [120] during the inference process.

We use Perplexity as the main metric for assessing the generalisation ability of our generative models. Here we use the variational lower bound to estimate the document perplexity: $\exp(-\frac{1}{D} \sum_d \frac{1}{N_d} \log p(d))$ following the setting of NVDM in Chapter 3.

Table 4.1 presents the test document perplexities of the topic models on the three datasets. The upper section of the table lists the results for finite neural topic models, with 50 or 200 topics, on the *MXM*, *20NewsGroups* and *RCV1* datasets. We compare our neural topic models with the Gaussian Softmax (GSM), Gaussian Stick Breaking (GSB) and Recurrent Stick Breaking (RSB) constructions to the online variational LDA (onlineLDA) [44] and neural variational inference LDA (NVLDA) [96] models. The lower section shows the results for the unbounded topic models, including our

¹<http://labrosa.ee.columbia.edu/millionsong/musixmatch> [8]

²<http://qwone.com/~jason/20Newsgroups>

³<http://trec.nist.gov/data/reuters/reuters.html>

⁴We use the vocabulary provided by Srivastava and Sutton [96] for direct comparison.

Finite Topic Model	MXM		20News		RCV1	
	50	200	50	200	50	200
GSM	306	272	822	830	717	602
GSB	309	296	838	826	788	634
RSB	311	297	835	822	750	628
OnlineLDA [44]	312	342	893	1015	1062	1058
NVLDA [96]	330	357	1073	993	791	797
Unbounded Topic Model	MXM		20News		RCV1	
RSB-TF	303		825		622	
HDP [108]	370		937		918	

Table 4.1: Perplexities of the topic models on the test datasets.

Finite Document Model	MXM		20News		RCV1	
	50	200	50	200	50	200
GSM	270	267	787	829	653	521
GSB	285	275	816	815	712	544
RSB	286	283	785	792	662	534
NVDM [75]	345	345	837	873	717	588
ProdLDA [96]	319	326	1009	989	780	788
Unbounded Document Model	MXM		20News		RCV1	
RSB-TF	285		788		532	

Table 4.2: Perplexities of document models on the test datasets.

truncation-free RSB (RSB-TF) and the online HDP topic model [108]. Amongst the finite topic models, the Gaussian softmax construction (GSM) achieves the lowest perplexity in most cases, while all of the GSM, GSB and RSB models are significantly better than the benchmark LDA and NVLDA models. Amongst our selection of unbounded topic models, we compare our truncation-free RSB model, which applies an RNN to dynamically increase the active topics (γ is set as $5e^{-5}$ by tuning on the development set), with the traditional non-parametric HDP topic model [102]. Here we see that the recurrent neural topic model performs significantly better than the HDP topic model on perplexity.

Next we evaluate our neural network parameterisations as document models with the implicit topic distribution introduced in Section 4.2.4. Table 4.2 compares the proposed neural document models with the benchmarks. The table compares the

results for a fixed dimension latent variable, 50 or 200, achieved by our neural document models to Product of Experts LDA (prodLDA) [96] and the Neural Variational Document Model (NVDM) [75]. According to our experimental results, the generalisation abilities of the GSM, GSB and RSB models are all improved by switching from the mixture decoder (Equation 4.5) to the softmax decoder (Equation 4.6), and their performance is also significantly better than the NVDM and ProdLDA. We hypothesise that this effect is due to the models not needing to infer the topic-word assignments, which makes optimisation much easier. Interestingly, the RSB model performs slightly better than the GSM and GSB on *20NewsGroups* in both the 50 and 200 topic settings. As mentioned in Section 4.2.4, GSM is a variant of NVDM that applies topic and word vectors to construct the topic distribution over words instead of directly modelling a multinomial distribution by a softmax function, which further simplifies optimisation. If it is not necessary to model the explicit topic distribution over words, using an implicit topic distribution may lead to better generalisation.

To further demonstrate the effectiveness of the stick-breaking construction, Figure 4.3 and Figure 4.4 present the average probability of each topic of GSM model and GSB model respectively, which are estimated by the posterior probability $q(z|d)$ of each document from *20NewsGroups*. Here we set the number of topics to 400 that is large enough for this dataset. Figure 4.3a shows that the topics with higher probability are evenly distributed. While in Figure 4.4a the higher probability ones are placed in the front, and we can see a small tail on the topics after 300. Moreover, Figure 4.3b and Figure 4.4b sort the average probability measures by decreasing order, and as can be seen the overall distribution of GSB has slightly heavier tail than GSM. Due to the sparsity inducing property of the stick-breaking construction, the topics on the tail are less likely to be sampled. This is also the advantage of stick-breaking construction when we apply the RSB-TF as a non-parametric topic model, since the model activates the topics according to the knowledge learned from data and it becomes less sensitive to the hyperparameter controlling the initial number of topics.

Figure 4.5 shows the impact on test perplexity for the neural topic models when the maximum number of topics is increased. The truncation-free RSB (RSB-TF) dynamically increases the active topics, we use a dashed line to represent its test perplexity for reference in the figure. We can see that the performance of the GSM model gets worse if the maximum number of topics exceeds 400, but the GSB and RSB are stable even though the number of topics far outstrips that which the model requires. In addition, the RSB model performs better than GSB when the number of

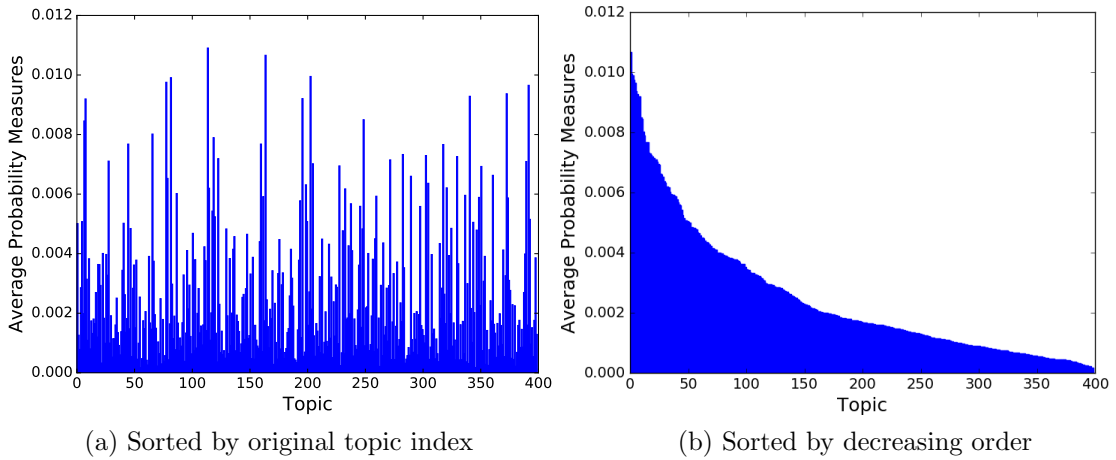


Figure 4.3: Corpus level topic probability distributions of GSM.

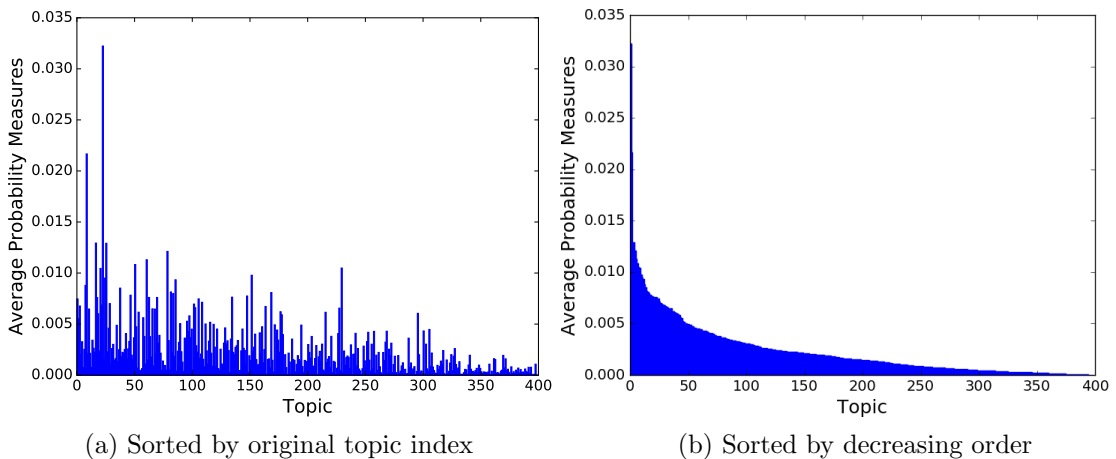


Figure 4.4: Corpus level topic probability distributions of GSB.

topics is under 200, but it becomes slightly worse than GSB when the number exceeds 400, possibly due to the difficulty of learning long sequences with RNNs.

Figure 4.6 shows the convergence process of the truncation-free RSB (RSB-TF) model on the *20NewsGroups*. With different initial number of topics, 10, 30, and 50. Dash lines represent the corresponding active topics. The RSB-TF dynamically increases the number of active topics to achieve a better variational lower bound. We can see the training perplexity keeps decreasing while the RSB-TF activates more topics. The numbers of active topics will stabilise when the convergence point is approaching (normally between 200 and 300 active topics on the *20NewsGroups*). Hence, as a non-parametric model, RSB-TF is not sensitive to the initial number of

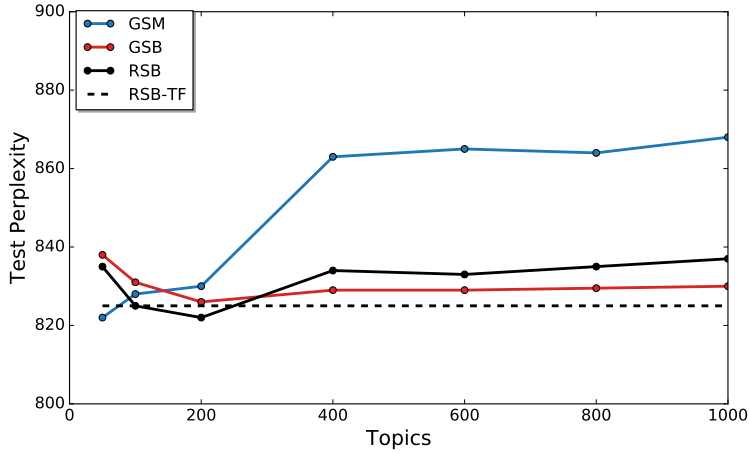


Figure 4.5: Test perplexities of the neural topic models with a varying maximum number of topics on the 20NewsGroups dataset.

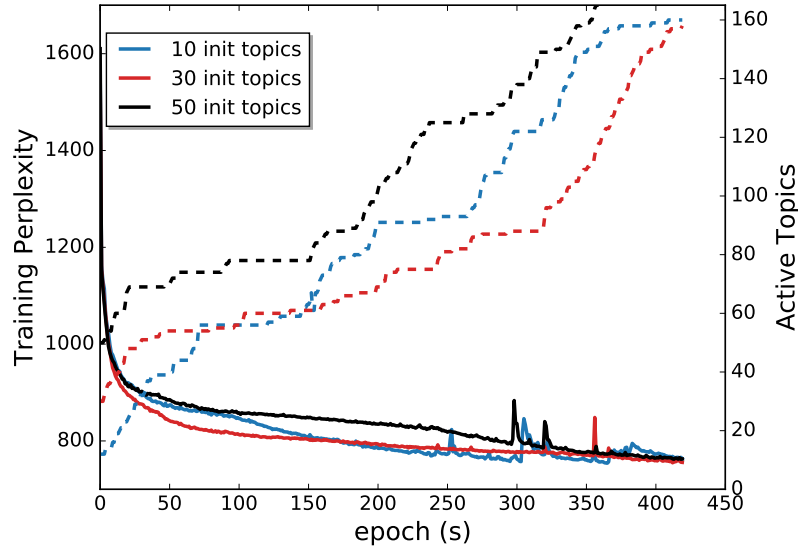


Figure 4.6: The convergence behavior of the truncation-free RSB model (RSB-TF) with different initial active topics on *20NewsGroups*.

active topics.

In addition, since the quality of the discovered topics is not directly reflected by perplexity (i.e. a function of log-likelihood), we evaluate the **topic observed coherence** by normalised point-wise mutual information (NPMI) [62]. Table 4.3 shows the topic observed coherence achieved by the finite neural topic models. We compute coherence over the top-5 words and top-10 words for all topics and then take the mean of both values. According to these results, there does not appear to be a significant difference in topic coherence amongst the neural topic models. We observe that in both the GSB and RSB, the NPMI scores of the former topics in

Topic Model	Topics	
	50	200
GSM	0.121	0.110
GSB	0.095	0.081
RSB	0.111	0.097
OnlineLDA	0.131	0.112
NVLDA	0.110	0.110
Document Model	Latent Dimension	
	50	200
GSM	0.223	0.186
GSB	0.217	0.171
RSB	0.224	0.177
NVDM	0.186	0.157
ProdLDA	0.240	0.190

Table 4.3: Topic coherence on *20NewsGroups* (higher is better).

the stick breaking order are higher than the latter ones. It is plausible as the stick-breaking construction implicitly assumes the order of the topics, the former topics obtain more sufficient gradients to update the topic distributions. Likewise we present the results obtained by the neural document models with implicit topic distributions. Though the topic probability distribution over words does not exist, we could rank the words by the positiveness of the connections between the words and each dimension of the latent variable. Interestingly the performance of these document models are significantly better than their topic model counterparts on topic coherence. The results of RSB-TF and HDP are not presented due to the fact that the number of active topics is dynamic, which makes these two models not directly comparable to the others. To further demonstrate the quality of the topics, we produce a t-SNE projection for the estimated topic proportions of each document in Figure C.1 (Appendix C).

4.5 Summary

This chapter presents alternative neural approaches to topic modelling by providing parameterisable distributions over topics which permit training by backpropagation in the framework of neural variational inference. We also introduce a family of neural topic models using the Gaussian softmax, Gaussian stick-breaking and recurrent stick-breaking constructions for parameterising the latent multinomial topic distributions of each document. With the help of the stick-breaking construction, we are

able to build neural topic models which exhibit similar sparse topic distributions as found with traditional Dirichlet-Multinomial models. By exploiting the ability of recurrent neural networks to model sequences of unbounded length, we further present a truncation-free variational inference method that allows the number of topics to dynamically increase, which is notionally neural unbounded topic model that is analogous to Bayesian non-parametric topic model. The evaluation results show that neural topic models achieve state-of-the-art performance on a range of standard document corpora, which demonstrate more flexibility and effectiveness of deep generative models than traditional probabilistic topic models. The experiments also compare the neural topic models (with explicit topic distribution) and the neural document models (with implicit topic distribution) using different parameterisation networks. The results show that if it is not necessary to model the explicit topic distribution over words, using an implicit topic distribution may lead to better generalisation.

Chapter 5

Neural Answer Selection Model

Different from the deep generative models for unsupervised document and topic modelling introduced in the previous contents, this chapter focuses on the generative model for supervised question answer selection. One major difference of this task from document/topic modelling is that there are a few labels available indicating whether an answer is correct to a given question. Here, we are interested in how is the performance of deep generative models applied on supervised learning tasks, and whether the neural variational inference is effective on carrying out sequence modelling.

5.1 Introduction

Question answer sentence selection is a task that selects the correct sentences answering a factoid question from a set of candidate sentences. The key point is to learn good representations of observed question and answer sequences. More specifically, the representations need to contain both semantic and syntactic information so that the model is able to figure out what information the question seeks to answer and whether the answer contains correct information. Previous works have focused on using only syntactic information to match questions and answers, for example the generative models on aligning the dependency trees of question answer pairs [110, 109, 33]. Then, the features are fed into a classifier for predicting correctness. However, only syntactic information is not good enough to solve such a difficult NLP task. Yih et al. [123] incorporates semantic information by using a combination of lexical semantic resources such as WordNet. Severyn and Moschitti [92] employs tree kernel SVMs to generate highly discriminative structural features that combine syntactic and semantic information encoded in the input trees. Recently, Yu et al. [125] directly applies a convolutional neural network over bigrams so that the model

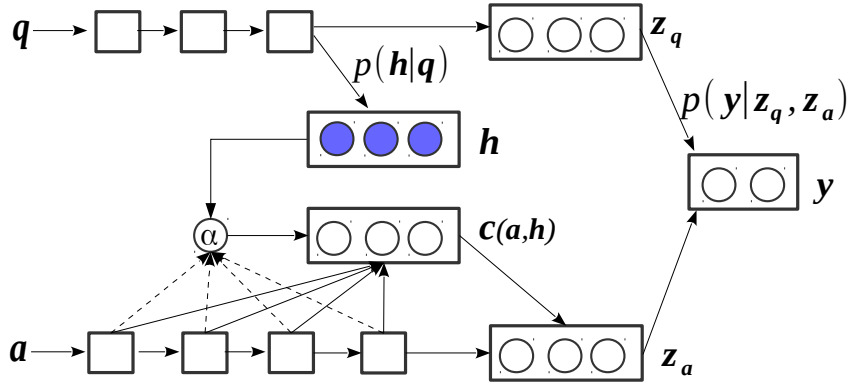


Figure 5.1: Neural Answer Selection Model.

automatically learns the representations containing both semantic and syntactic information for predicting the matchness of question answer pairs. Other neural models [117, 100, 59] further exploit memory networks, where long-term memories act as dynamic knowledge bases. A more complicated deep model [36] applies an attentive mechanism to help read and comprehend long articles and answer the corresponding questions.

In this chapter, we introduce the Neural Answer Selection Model (NASM) which imbues LSTMs with a latent stochastic attention mechanism to model the semantics of question-answer pairs and predict their relatedness. This mechanism allows the model to deal with the ambiguity inherent in the task and learns pair-specific representations that are more effective at predicting answer matches, rather than independent embeddings of question and answer sentences. Bayesian inference provides a natural safeguard against overfitting, especially as the training sets available for this task are small. The experiments show that the LSTM with a latent stochastic attention mechanism learns an effective attention model and outperforms both previously published results, and our own strong non-stochastic attention baselines.

5.2 Neural Answer Selection Model (NASM)

The Neural Answer Selection Model (Figure 5.1) is a supervised model that learns the question and answer representations and predicts their relatedness. Assume a question q is associated with a set of answer sentences $\{a_1, a_2, \dots, a_n\}$, together with their judgements $\{y_1, y_2, \dots, y_n\}$, where $y_m = 1$ if the answer a_m is correct and $y_m = 0$ otherwise. This is a classification task where we treat each training data

point as a triple $(\mathbf{q}, \mathbf{a}, \mathbf{y})$ while predicting \mathbf{y} for the unlabelled question-answer pair (\mathbf{q}, \mathbf{a}) .

NASM employs two different LSTMs to embed raw question inputs \mathbf{q} and answer inputs \mathbf{a} . Let $\mathbf{s}_q(j)$ and $\mathbf{s}_a(i)$ be the state outputs of the two LSTMs, and i, j be the positions of the states. Conventionally, the last state outputs $\mathbf{s}_q(|\mathbf{q}|)$ and $\mathbf{s}_a(|\mathbf{a}|)$, as the independent question and answer representations, can be used for relatedness prediction. In NASM, however, we aim to learn pair-specific representations through a latent attention mechanism, which is more effective for pair relatedness prediction.

An attention model is applied in NASM to focus on the words in the answer sentence that are prominent for predicting the answer matched to the current question. Instead of using a deterministic question vector, such as $\mathbf{s}_q(|\mathbf{q}|)$, NASM employs a latent distribution $p_\theta(\mathbf{h}|\mathbf{q})$ to model the question semantics, which is a parameterised diagonal Gaussian $\mathcal{N}(\mathbf{h}|\boldsymbol{\mu}(\mathbf{q}), \boldsymbol{\sigma}^2(\mathbf{q}))$. Therefore, the attention model extracts a context vector $\mathbf{c}(\mathbf{a}, \mathbf{h})$ by iteratively attending to the answer tokens based on the stochastic vector $\mathbf{h} \sim p_\theta(\mathbf{h}|\mathbf{q})$. In doing so the model is able to adapt to the ambiguity inherent in questions and obtain salient information through attention. Compared to its deterministic counterpart (applying $\mathbf{s}_q(|\mathbf{q}|)$ as the question semantics), the stochastic units incorporated into NASM allow multi-modal attention distributions. Further, NASM is more robust against overfitting due to that stochastic representations also model the uncertainties. Hence, if we marginalise over the latent variable, the model actually feeds a distribution to next layer instead of a deterministic vector. This is analogous to the application of dropout in deep neural networks. [25] and [55] provide more discussions about the connection between dropout and variational inference.

In this model, the conditional distribution $p_\theta(\mathbf{h}|\mathbf{q})$ is:

$$\boldsymbol{\pi}_\theta = g_\theta(f_q^{\text{LSTM}}(\mathbf{q})) = g_\theta(\mathbf{s}_q(|\mathbf{q}|)) \quad (5.1)$$

$$\boldsymbol{\mu}_\theta = l_1(\boldsymbol{\pi}_\theta), \log \boldsymbol{\sigma}_\theta = l_2(\boldsymbol{\pi}_\theta) \quad (5.2)$$

For each question \mathbf{q} , the neural network generates the corresponding parameters $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ that parameterise the latent distribution over question semantics \mathbf{h} . Following [6], the attention model is defined as:

$$\alpha(i) \propto \exp(\mathbf{W}_\alpha^T \tanh(\mathbf{W}_h \mathbf{h} + \mathbf{W}_s \mathbf{s}_a(i))) \quad (5.3)$$

$$\mathbf{c}(\mathbf{a}, \mathbf{h}) = \sum_i \mathbf{s}_a(i) \alpha(i) \quad (5.4)$$

$$\mathbf{z}_a(\mathbf{a}, \mathbf{h}) = \tanh(\mathbf{W}_a \mathbf{c}(\mathbf{a}, \mathbf{h}) + \mathbf{W}_n \mathbf{s}_a(|\mathbf{a}|)) \quad (5.5)$$

where $\alpha(i)$ is the normalised attention score at answer token i , and the context vector $\mathbf{c}(\mathbf{a}, \mathbf{h})$ is the weighted sum of all the state outputs $\mathbf{s}_a(i)$. We adopt $\mathbf{z}_q(\mathbf{q}), \mathbf{z}_a(\mathbf{a}, \mathbf{h})$

as the question and answer representations for predicting their relatedness \mathbf{y} . $\mathbf{z}_q(\mathbf{q})$ is a deterministic vector that is equal to $\mathbf{s}_q(|\mathbf{q}|)$, while $\mathbf{z}_a(\mathbf{a}, \mathbf{h})$ is a combination of the sequence output $\mathbf{s}_a(|\mathbf{a}|)$ and the context vector $\mathbf{c}(\mathbf{a}, \mathbf{h})$ (Eq. 5.5). For the prediction of pair relatedness \mathbf{y} , we model the conditional probability distribution $p_\theta(\mathbf{y}|\mathbf{z}_q, \mathbf{z}_a)$ by sigmoid function:

$$p_\theta(\mathbf{y} = 1|\mathbf{z}_q, \mathbf{z}_a) = \sigma(\mathbf{z}_q^T \mathbf{M} \mathbf{z}_a + b) \quad (5.6)$$

To maximise the log-likelihood $\log p(\mathbf{y}|\mathbf{q}, \mathbf{a})$ we use the variational lower bound:

$$\begin{aligned} \mathcal{L} &= \mathbb{E}_{q_\phi(\mathbf{h})}[\log p_\theta(\mathbf{y}|\mathbf{z}_q(\mathbf{q}), \mathbf{z}_a(\mathbf{a}, \mathbf{h}))] - D_{\text{KL}}(q_\phi(\mathbf{h})||p_\theta(\mathbf{h}|\mathbf{q})) \\ &\leq \log \int p_\theta(\mathbf{y}|\mathbf{z}_q(\mathbf{q}), \mathbf{z}_a(\mathbf{a}, \mathbf{h})) p_\theta(\mathbf{h}|\mathbf{q}) d\mathbf{h} \\ &= \log p(\mathbf{y}|\mathbf{q}, \mathbf{a}) \end{aligned} \quad (5.7)$$

Following the neural variational inference framework, we construct a deep neural network as the inference network $q_\phi(\mathbf{h}|\mathbf{q}, \mathbf{a}, \mathbf{y}) = \mathcal{N}(\mathbf{h}|\boldsymbol{\mu}_\phi(\mathbf{q}, \mathbf{a}, \mathbf{y}), \boldsymbol{\sigma}_\phi^2(\mathbf{q}, \mathbf{a}, \mathbf{y}))$:

$$\begin{aligned} \boldsymbol{\pi}_\phi &= g_\phi(f_q^{\text{LSTM}}(\mathbf{q}), f_a^{\text{LSTM}}(\mathbf{a}), f_y(\mathbf{y})) \\ &= g_\phi(\mathbf{s}_q(|\mathbf{q}|), \mathbf{s}_a(|\mathbf{a}|), \mathbf{s}_y) \end{aligned} \quad (5.8)$$

$$\boldsymbol{\mu}_\phi = l_3(\boldsymbol{\pi}_\phi), \log \boldsymbol{\sigma}_\phi = l_4(\boldsymbol{\pi}_\phi) \quad (5.9)$$

where \mathbf{q} and \mathbf{a} are also modelled by LSTMs¹, and the relatedness label \mathbf{y} is modelled by a simple linear transformation into the vector \mathbf{s}_y . According to the joint representation $\boldsymbol{\pi}_\phi$, we then generate the parameters $\boldsymbol{\mu}_\phi$ and $\boldsymbol{\sigma}_\phi$, which parameterise the variational distribution over the question semantics \mathbf{h} . To emphasise, though both $p_\theta(\mathbf{h}|\mathbf{q})$ and $q_\phi(\mathbf{h}|\mathbf{q}, \mathbf{a}, \mathbf{y})$ are modelled as parameterised Gaussian distributions, $q_\phi(\mathbf{h}|\mathbf{q}, \mathbf{a}, \mathbf{y})$ as an approximation only functions during inference by producing samples to compute the stochastic gradients, while $p_\theta(\mathbf{h}|\mathbf{q})$ is the generative distribution that generates the samples for predicting the question-answer relatedness \mathbf{y} .

Based on the samples $\mathbf{h} \sim q_\phi(\mathbf{h}|\mathbf{q}, \mathbf{a}, \mathbf{y})$, we use SGVB to optimise the lower bound (Eq.5.7). The model parameters θ and the inference network parameters ϕ are updated jointly using their stochastic gradients. In this case, similar to the NVDM, the Gaussian KL divergence $D_{\text{KL}}[q_\phi(\mathbf{h}|\mathbf{q}, \mathbf{a}, \mathbf{y})||p_\theta(\mathbf{h}|\mathbf{q})]$ can be analytically computed during training process.

Source	Set	Questions	QA Pairs	Judgement
QASent	Train	1,229	53,417	automatic
	Dev	82	1,148	manual
	Test	100	1,517	manual
WikiQA	Train	2,118	20,360	manual
	Dev	296	2,733	manual
	Test	633	6,165	manual

Table 5.1: Statistics of *QASent* and *WikiQA*. Judgement denotes whether correctness was determined automatically or by human annotators.

5.3 Experiments

5.3.1 Dataset & Setup

We experiment on two answer selection datasets, the *QASent* and the *WikiQA* datasets. *QASent* [110] is created from the TREC QA track, and the *WikiQA* [122] is constructed from Wikipedia, which is less noisy and less biased towards lexical overlap². Table 5.1 summarises the statistics of the two datasets.

In order to investigate the effectiveness of our NASM model we also implemented two strong baseline models — a vanilla LSTM model (LSTM) and an LSTM model with a deterministic attention mechanism (LSTM+Att). The former directly applies the QA matching function (Eq. 5.6) on the independent question and answer representations which are the last state outputs $\mathbf{s}_q(|\mathbf{q}|)$ and $\mathbf{s}_a(|\mathbf{a}|)$ from the question and answer LSTM models. The latter adds an attention model to learn pair-specific representation for prediction on the basis of the vanilla LSTM. Moreover, LSTM+Att is the deterministic counterpart of NASM, which has the same neural network architecture as NASM. The only difference is that it replaces the stochastic units \mathbf{h} with deterministic ones, and no inference network is required for stochastic estimation. Following previous work, for each of our models we also add a lexical overlap feature by combining a co-occurrence word count feature with the probability generated from the neural model. MAP and MRR are adopted as the evaluation metrics for this task.

To facilitate direct comparison with previous work we follow the same experimental setup as [125] and [91]. The word embeddings ($K = 50$) are obtained by running the `word2vec` tool [76] on the English Wikipedia dump and the *AQUAINT*³ corpus.

¹In this case, the LSTMs for \mathbf{q} and \mathbf{a} are shared by the inference network and the generative model, but there is no restriction on using different LSTMs in the inference network (more parameters will be used).

²Yang et al [122] provide detailed explanation of the differences between the two datasets.

³<https://catalog.ldc.upenn.edu/LDC2002T31>

Model	QASent		WikiQA	
	MAP	MRR	MAP	MRR
Published Models				
PV	0.6762	0.7514	0.5976	0.6058
WA	0.7063	0.7740	—	—
LCLR	0.7092	0.7700	0.5993	0.6068
Bigram-CNN	0.7113	0.7846	0.6520	0.6652
Deep CNN	0.7186	0.7826	—	—
Our Models				
LSTM	0.7228	0.7986	0.6820	0.6988
LSTM + Att	0.7289	0.8072	0.6855	0.7041
NASM	0.7339	0.8117	0.6886	0.7069

Table 5.2: Results of our models (LSTM, LSTM + Att, NASM) in comparison with other state of the art models on the *QASent* and *WikiQA* dataset.

Q1	how old was sue lyon when she made lolita
A_{NASM}	the actress who played lolita , sue lyon , was fourteen at the time of filming .
A_{LSTM}	the actress who played lolita , sue lyon , was fourteen at the time of filming .
Q2	how much is centavos in mexico
A_{NASM}	the peso is subdivided into 100 centavos , represented by " _UNK_ "
A_{LSTM}	the peso is subdivided into 100 centavos , represented by " _UNK_ "
Q3	what does a liquid oxygen plant look like
A_{NASM}	the blue color of liquid oxygen in a dewar flask
A_{LSTM}	the blue color of liquid oxygen in a dewar flask

Figure 5.2: A visualisation of attention scores on answer sentences.

We use LSTMs with 3 layers and 50 hidden units, and apply 40% dropout after the embedding layer. For the construction of the inference network, we use an MLP (Eq. 5.1) with 2 layers and tanh units of 50 dimension, and an MLP (Eq. 5.8) with 2 layers and tanh units of 150 dimension for modelling the joint representation. During training we apply stochastic estimation by taking one sample for computing the gradients, while in prediction we use 20 samples to calculate the expectation of the lower bound. Figure 5.3 presents the standard deviation of NASM’s MAP scores while using different numbers of samples. Considering the trade-off between computational cost and variance, we chose 20 samples for prediction in all the experiments. The models are trained using Adam [53], with hyperparameters selected by optimising the MAP score on the development set.

5.3.2 Experimental Results

Table 5.2 compares the results of our models with current state-of-the-art models on both answer selection datasets. PV is the paragraph vector [64]. Bigram-CNN is the simple convolutional model reported in [125]. Deep CNN is the deep convolutional model from [91]. WA is a model based on word alignment [112]. LCLR is the SVM-based classifier trained using a set of features. Model + Cnt means that the result is obtained from a combination of a lexical overlap feature and the output from the distributional model. On the *QASent* dataset, our vanilla LSTM model outperforms the deep CNN⁴ model by approximately 7% on MAP and 6% on MRR. The LSTM+Att performs slightly better than the vanilla LSTM model, and our NASM improves the results further. Since the *QASent* dataset is biased towards lexical overlapping features, after combining with a co-occurrence word count feature, our best model NASM outperforms all the previous models, including both neural network based models and classifiers with a set of hand-crafted features (e.g. LCLR). Similarly, on the *WikiQA* dataset, all of our models outperform the previous distributional models by a large margin. By including a word count feature, our models improve further and achieve the state-of-the-art. Notably, on both datasets, our two LSTM-based models have set strong baselines and NASM works even better, which demonstrates the effectiveness of introducing stochastic units to model question semantics in this answer sentence selection task.

In Figure 5.2, we compare the effectiveness of the latent attention mechanism (NASM) and its deterministic counterpart (LSTM+Att) by visualising the attention scores on the answer sentences. For most of the negative answer sentences, neither of the two attention models can attend to reasonable words that are beneficial for predicting relatedness. But for the correct answer sentences, such as the ones in Figure 5.2, both attention models are able to capture crucial information by attending to different parts of the sentence based on the question semantics. Interestingly, compared to the deterministic counterpart LSTM+Att, our NASM assigns higher attention scores on the prominent words that are relevant to the question, which forms a more peaked distribution and in turn helps the model achieve better performance.

⁴As stated in [123] that the evaluation scripts used by previous work are noisy — 4 out of 72 questions in the test set are treated answered incorrectly. This makes the MAP and MRR scores $\sim 4\%$ lower than the *true* scores. Since [91] and [112] use a cleaned-up evaluation scripts, we apply the original noisy scripts to re-evaluate their outputs in order to make the results directly comparable with previous work.

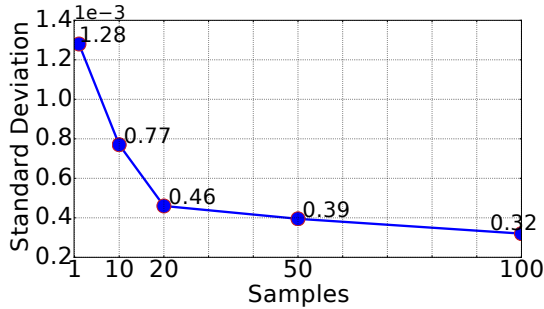


Figure 5.3: The standard deviations of MAP scores computed by running 10 NASM models on WikiQA with different numbers of samples.

5.3.3 Discussion

The basic intuition of applying deep generative models for learning language is that the latent distributions grant the ability to sum over all the possibilities in terms of semantics. As shown in the experiments, neural variational inference brings consistent improvements on the performance of the NLP tasks including document/topic modelling and question answer selection. Compared to the unsupervised learning model NVDM, the supervised learning model NASM in this context enjoys more benefits from the perspective of optimisation, since Bayesian learning naturally guards against overfitting.

According to the variational lower bound (Eq. 3.1) of NVDM, we adopt $p(\mathbf{h})$ as a standard Gaussian prior, the KL divergence term $D_{\text{KL}}[q_{\phi}(\mathbf{h}|\mathbf{X})||p(\mathbf{h})]$ can be analytically computed as $\frac{1}{2}(K - \|\boldsymbol{\mu}\|^2 - \|\boldsymbol{\sigma}\|^2 + \log |\boldsymbol{\sigma}^2|)$. It is not difficult to find that it actually acts as L2 regulariser when we update the $\boldsymbol{\mu}$. Similarly, in NASM (Eq. 5.7), we also have the KL divergence term $D_{\text{KL}}[q_{\phi}(\mathbf{h}|\mathbf{q}, \mathbf{a}, \mathbf{y})||p_{\theta}(\mathbf{h}|\mathbf{q})]$. Different from NVDM, it attempts to minimise the distance between $q_{\phi}(\mathbf{h}|\mathbf{q}, \mathbf{a}, \mathbf{y})$ and $p_{\theta}(\mathbf{h}|\mathbf{q})$ that are both conditional distributions. Because $p_{\theta}(\mathbf{h}|\mathbf{q})$ as well as $q_{\phi}(\mathbf{h}|\mathbf{q}, \mathbf{a}, \mathbf{y})$ are learned during training, the two distributions are mutually restrained while being updated. Therefore, NVDM simply penalises the large $\boldsymbol{\mu}$ and encourages $q_{\phi}(\mathbf{h}|\mathbf{X})$ to approach the prior $p(\mathbf{h})$ for every document \mathbf{X} , but in NASM, $p_{\theta}(\mathbf{h}|\mathbf{q})$ acts like a moving baseline distribution which regularises the update of $q_{\phi}(\mathbf{h}|\mathbf{q}, \mathbf{a}, \mathbf{y})$ for every different condition. In practice, we use early stopping by observing the prediction performance on development dataset for the question answer selection task. Using the same learning rate and neural network structure, LSTM+Att reaches optimal performance and starts to overfit on training dataset generally at the 20th iteration, while NASM starts to overfit around the 35th iteration.

More interestingly, in the question answer selection experiments, NASM learns more peaked attention scores than its deterministic counterpart LSTM+Att. For the update process of LSTM+Att, we find there exists a relatively big variance in the gradients w.r.t. question semantics (LSTM+Att applies deterministic $\mathbf{s}_q(|\mathbf{q}|)$ while NASM applies stochastic \mathbf{h}). This is because the training dataset is small and contains many negative answer sentences that brings no benefit but noise to the learning of the attention model. In contrast, for the update process of NASM, we observe more stable gradients w.r.t. the parameters of latent distributions. The optimisation of the lower bound on one hand maximises the conditional log-likelihood (that the deterministic counterpart cares about) and on the other hand minimises the KL-divergence (that regularises the gradients). Hence, each update of the lower bound actually keeps the gradients w.r.t. $\boldsymbol{\mu}$ from swinging heavily. Besides, since the values of $\boldsymbol{\sigma}$ are not very significant in this case, the distribution of attention scores mainly depends on $\boldsymbol{\mu}$. Therefore, the learning of the attention model benefits from the regularisation as well, and it explains the fact that NASM learns more peaked attention scores which in turn helps achieve a better prediction performance.

Certainly, the improvements of the performance in this question answer selection task is more like icing on the cake. The fact of lacking training data makes the deterministic counterpart easily overfit, which gives the opportunity to latent variable models. However, if the labelled question answer pairs are sufficient enough, the deterministic counterpart should be able to achieve adequate accuracy on predicting question answer matchness.

5.4 Summary

This chapter discusses the neural answer selection model (NASM) which imbues LSTMs with a latent stochastic attention mechanism to model the semantics of question-answer pairs and predict their relatedness. Different from the unsupervised generative models for document/topic modelling introduced in the previous chapters, NASM is a supervised conditional model but its inference can also be carried out in the same neural variational inference framework. These models are simple, expressive and can be trained efficiently with the highly scalable stochastic gradient back-propagation. Hence, we show the effectiveness of the continuous latent variable models for NLP, and the neural variational inference is capable of generalising to incorporate any type of neural network for different application scenarios.

Part III

Discrete Latent Variable Models

Chapter 6

Latent Sentence Compression Models

Compared to the continuous latent variables that commonly applied for learning stochastic representations, discrete latent variables are more extensive on applications, for example structure prediction, planning and clustering. Since discrete representation is a natural fit for predictive systems with categorical properties, it has more advantages on developing the potential of deep neural networks. Language is a straightforward case where each sentence is represented by a sequence of discrete word symbols. However, variational inference on discrete latent variables is relatively difficult due to intractable marginalisation and the high variance issue of the sampling-based gradient estimators. Chapter 2 has introduced a few techniques to mitigate the problem when the continuous reparameterisation trick is not applicable. In this chapter, we are going to discuss the discrete latent variable models for modelling language. We formulate a discrete variational auto-encoder and apply it to the task of compressing long sentences into short ones.

6.1 Introduction

The recurrent sequence-to-sequence paradigm for natural language generation [48, 101] has achieved remarkable recent success and is now the approach of choice for applications such as machine translation [6], caption generation [121] and speech recognition [18]. While these models have developed sophisticated conditioning mechanisms, e.g. attention, fundamentally they are discriminative models trained only to approximate the conditional output distribution of strings. In this chapter we explore modelling the language sentences by a discrete variational auto-encoder (VAE).

Auto-encoder [86] is a typical neural network architecture for learning compact data representations, with the general aim of performing dimensionality reduction on embeddings [41]. In this context, rather than seeking to embed sentences as points in a vector space, we model the representations with explicit natural language sentences. This approach is a natural fit for summarisation tasks such as sentence compression. According to this, we propose a generative **auto-encoding sentence compression** (ASC) model, where we introduce a latent language model to provide the variable-length compact summary. The objective is to perform Bayesian inference for the posterior distribution of summaries conditioned on the observed utterances. Hence, in the framework of VAE, we construct an inference network as the variational approximation of the posterior, which generates compression samples to optimise the variational lower bound.

The most common family of variational auto-encoders [56] relies on the reparameterisation trick, which is not applicable for our discrete latent language model. Instead, we employ the REINFORCE algorithm [79, 77] introduced in Chapter 2 to mitigate the problem of high variance during sampling-based variational inference. Nevertheless, when directly applying the RNN encoder-decoder to model the variational distribution it is very difficult to generate reasonable compression samples in the early stages of training, since each hidden state of the sequence would have $|V|$ possible words to be sampled from. To combat this we employ pointer networks [105] to construct the variational distribution. This biases the latent space to sequences composed of words only appearing in the source sentence (i.e. the size of softmax output for each state becomes the length of current source sentence), which amounts to applying an extractive compression model for the variational approximation.

In order to further boost the performance on sentence compression, we employ a supervised **forced-attention sentence compression** model (FSC) trained on labelled data to teach the ASC model to generate compression sentences. The FSC model shares the pointer network of the ASC model and combines a softmax output layer over the whole vocabulary. Therefore, while training on the sentence-compression pairs, it is able to balance copying a word from the source sentence with generating it from the background distribution. More importantly, by jointly training on the labelled and unlabelled datasets, this shared pointer network enables the model to work in a semi-supervised scenario. In this case, the FSC teaches the ASC to generate reasonable samples, while the pointer network trained on a large unlabelled data set helps the FSC model to perform better abstractive summarisation.

In the experiment section, we evaluate the proposed model by jointly training the generative (ASC) and discriminative (FSC) models on the standard Gigaword sentence compression task with varying amounts of labelled and unlabelled data. The results demonstrate that by introducing a latent language variable we are able to match the previous benchmarkers with small amount of the supervised data. When we employ our mixed discriminative and generative objective with all of the supervised data the model significantly outperforms all previously published results.

6.2 Related Work

As one of the typical sequence-to-sequence tasks, sentence-level summarisation has been explored by a series of discriminative encoder-decoder neural models. Filippova et al [24] carries out extractive summarisation via deletion with LSTMs, while Rush et al [87] applies a convolutional encoder and an attentional feed-forward decoder to generate abstractive summarises, which provides the benchmark for the *Gigaword* dataset. Nallapati et al [80] further improves the performance by exploring multiple variants of RNN encoder-decoder models. The recent works [32], [70], [80] and [30] also apply the similar idea of combining pointer networks and softmax output. However, different from all these discriminative models above, we explore generative models for sentence compression. Instead of training the discriminative model on a big labelled dataset, our original intuition of introducing a combined pointer networks is to bridge the unsupervised generative model (ASC) and supervised model (FSC) so that we could utilise a large additional dataset, either labelled or unlabelled, to boost the compression performance. Dai and Le [19] also explored semi-supervised sequence learning, but in a pure deterministic model focused on learning better vector representations.

Recently variational auto-encoders have been applied in a variety of fields as deep generative models. In computer vision Kingma and Welling [56], Rezende et al [84], and Gregor et al [28] have demonstrated strong performance on the task of image generation and Eslami et al [23] proposes variable-sized variational auto-encoders to identify multiple objects in images. While in natural language processing, there are variants of VAEs on modelling documents [75] (NVDM introduced in Chapter 3), sentences [14] and discovery of relations [72]. Apart from the typical initiations of VAEs, there are also a series of works that employs generative models for supervised learning tasks. For instance, Ba et al [5] learns visual attention for multiple objects by optimising a variational lower bound, Kingma et al [54] implements a semi-supervised

framework for image classification and Miao et al [75] (NASM introduced in Chapter 5) applies a conditional variational approximation in the task of factoid question answering. Dyer et al [22] proposes a generative model that explicitly extracts syntactic relationships among words and phrases which further supports the argument that generative models can be a statistically efficient method for learning neural networks from small data.

6.3 Auto-encoding Sentence Compression Model (ASC)

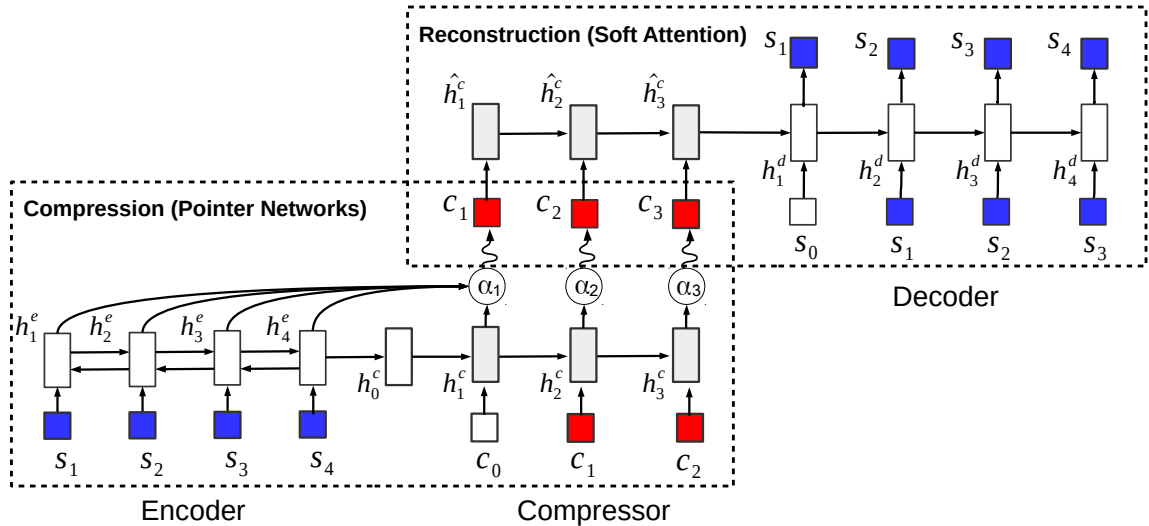


Figure 6.1: Auto-encoding Sentence Compression Model

In this section, we introduce the auto-encoding sentence compression model (Figure 6.1)¹ in the framework of variational auto-encoders. The ASC model consists of four recurrent neural networks – an encoder, a compressor, a decoder and a language model. Let \mathbf{s} be the source sentence, and \mathbf{c} be the compression sentence. The **compression** model (encoder-compressor) is the inference network $q_{\phi}(\mathbf{c}|\mathbf{s})$ that takes source sentences \mathbf{s} as inputs and generates extractive compressions \mathbf{c} . The **reconstruction** model (compressor-decoder) is the generative network $p_{\theta}(\mathbf{s}|\mathbf{c})$ that reconstructs source sentences \mathbf{s} based on the latent compressions \mathbf{c} . Hence, the forward pass starts from the encoder to the compressor and ends at the decoder. As

¹The language model, layer connections and decoder soft attentions are omitted in Figure 6.1 for clarity.

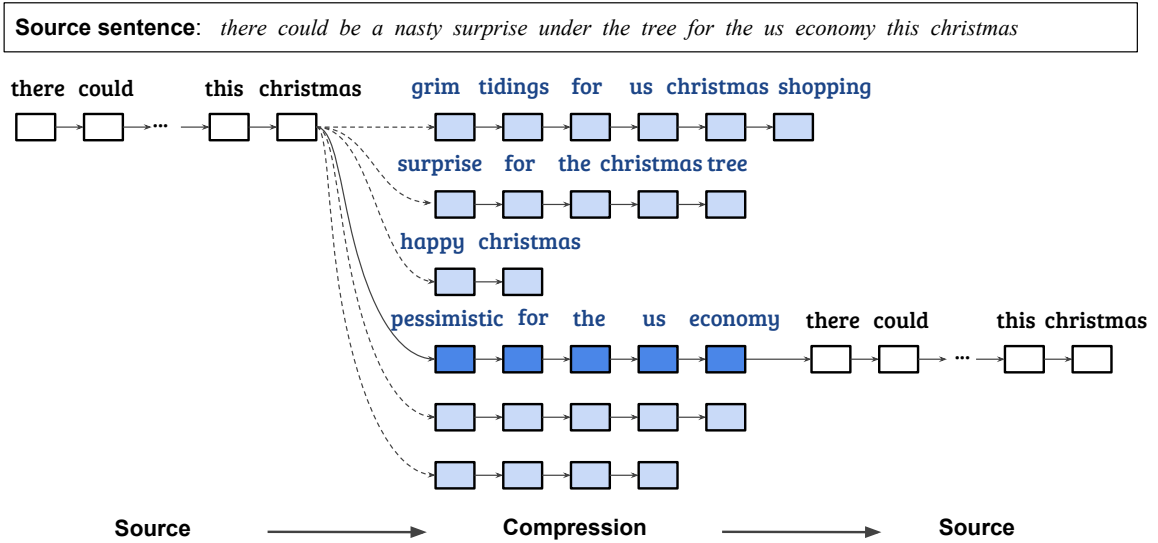


Figure 6.2: Example of auto-encoding sentence compression.

the prior distribution, a language model $p(\mathbf{c})$ is pre-trained to regularise the latent compressions so that the samples drawn from the compression model are likely to be reasonable natural language sentences.

For example, in Figure 6.2, the compression model generates multiple possible compressions for the source sentence “*there could be a nasty surprise under the tree for the us economy this christmas*”, and the generated compression sentences will be used to reconstruct the source sentence in the reconstruction model. All of the possible sentence compressions are evaluated by the variational lower bound.

6.3.1 Compression

For the compression model (encoder-compressor), $q_\phi(\mathbf{c}|\mathbf{s})$, we employ a pointer network consisting of a bidirectional LSTM encoder that processes the source sentences, and an LSTM compressor that generates compressed sentences by attending to the encoded source words.

Let \mathbf{s}_i be the words in the source sentences, \mathbf{h}_i^e be the corresponding state outputs of the encoder. $\tilde{\mathbf{h}}_i^e$ are the concatenated hidden states from each direction:

$$\mathbf{h}_i^e = f_{\text{enc}}^{\leftarrow}(\tilde{\mathbf{h}}_{i-1}^e, \mathbf{s}_i) || f_{\text{enc}}^{\rightarrow}(\tilde{\mathbf{h}}_{i+1}^e, \mathbf{s}_i) \quad (6.1)$$

Further, let \mathbf{c}_j be the words in the compressed sentences, \mathbf{h}_j^c be the state outputs of the compressor. We construct the predictive distribution by attending to the words

in the source sentences:

$$\mathbf{h}_j^c = f_{\text{com}}(\mathbf{h}_{j-1}^c, \mathbf{c}_{j-1}) \quad (6.2)$$

$$\mathbf{u}_j(i) = \mathbf{w}_3^T \tanh(\mathbf{W}_1 \mathbf{h}_j^c + \mathbf{W}_2 \mathbf{h}_i^e) \quad (6.3)$$

$$q_\phi(\mathbf{c}_j | \mathbf{c}_{1:j-1}, \mathbf{s}) = \text{softmax}(\mathbf{u}_j) \quad (6.4)$$

where \mathbf{c}_0 is the start symbol for each compressed sentence and \mathbf{h}_0^c is initialised by the source sentence vector of $\mathbf{h}_{|s|}^e$. In this case, all the words \mathbf{c}_j sampled from $q_\phi(\mathbf{c}_j | \mathbf{c}_{1:j-1}, \mathbf{s})$ are the subset of the words appeared in the source sentence (i.e. $\mathbf{c}_j \in \mathbf{s}$).

6.3.2 Reconstruction

For the reconstruction model (compressor-decoder) $p_\theta(\mathbf{s} | \mathbf{c})$, we apply a soft attention sequence-to-sequence model to generate the source sentence \mathbf{s} based on the compression samples $\mathbf{c} \sim q_\phi(\mathbf{c} | \mathbf{s})$.

Let \mathbf{s}_k be the words in the reconstructed sentences and \mathbf{h}_k^d be the corresponding state outputs of the decoder:

$$\mathbf{h}_k^d = f_{\text{dec}}(\mathbf{h}_{k-1}^d, \mathbf{s}_{k-1}) \quad (6.5)$$

In this model, we directly use the recurrent cell of the compressor to encode the compression samples²:

$$\hat{\mathbf{h}}_j^c = f_{\text{com}}(\hat{\mathbf{h}}_{j-1}^c, \mathbf{c}_j) \quad (6.6)$$

where the state outputs $\hat{\mathbf{h}}_j^c$ corresponding to the word inputs \mathbf{c}_j are different from the outputs \mathbf{h}_j^c in the compression model, since we block the information from the source sentences. We also introduce a start symbol \mathbf{s}_0 for the reconstructed sentence and \mathbf{h}_0^d is initialised by the last state output $\hat{\mathbf{h}}_{|\mathbf{c}|}^c$. The soft attention model is defined as:

$$v_k(j) = \mathbf{w}_6^T \tanh(\mathbf{W}_4 \mathbf{h}_k^d + \mathbf{W}_5 \hat{\mathbf{h}}_j^c) \quad (6.7)$$

$$\gamma_k(j) = \text{softmax}(v_k(j)) \quad (6.8)$$

$$\mathbf{d}_k = \sum_j^{|\mathbf{c}|} \gamma_k(j) \hat{\mathbf{h}}_j^c(v_k(j)) \quad (6.9)$$

We then construct the predictive probability distribution over reconstructed words using a softmax:

$$p_\theta(\mathbf{s}_k | \mathbf{s}_{1:k-1}, \mathbf{c}) = \text{softmax}(\mathbf{W}_7 \mathbf{d}_k) \quad (6.10)$$

²The recurrent parameters of the compressor are not updated by the gradients from the reconstruction model.

6.3.3 Inference

In the ASC model there are two sets of parameters, ϕ and θ , that need to be updated during inference. Due to the non-differentiability of the model, the reparameterisation trick of the VAE is not applicable in this case. Thus, we use the REINFORCE algorithm [79, 77] to reduce the variance of the gradient estimator.

The variational lower bound of the ASC model is:

$$\begin{aligned} L &= \mathbb{E}_{q_\phi(\mathbf{c}|\mathbf{s})}[\log p_\theta(\mathbf{s}|\mathbf{c})] - D_{KL}[q_\phi(\mathbf{c}|\mathbf{s})||p(\mathbf{c})] \\ &\leq \log \int \frac{q_\phi(\mathbf{c}|\mathbf{s})}{q_\phi(\mathbf{c}|\mathbf{s})} p_\theta(\mathbf{s}|\mathbf{c}) p(\mathbf{c}) d\mathbf{c} = \log p(\mathbf{s}) \end{aligned} \quad (6.11)$$

Therefore, by optimising the lower bound (Eq. 6.11), the model balances the selection of keywords for the summaries and the efficacy of the composed compressions, corresponding to the reconstruction error and KL divergence respectively.

In practice, the pre-trained language model prior $p(\mathbf{c})$ prefers short sentences for compressions. As one of the drawbacks of VAEs, the KL divergence term in the lower bound pushes every sample drawn from the variational distribution towards the prior. Thus acting to regularise the posterior, but also to restrict the learning of the encoder. If the estimator keeps sampling short compressions during inference, the LSTM decoder would gradually rely on the contexts from the decoded words instead of the information provided by the compressions, which does not yield the best performance on sentence compression.

Here, we introduce a co-efficient λ to scale the learning signal of the KL divergence:

$$L = \mathbb{E}_{q_\phi(\mathbf{c}|\mathbf{s})}[\log p_\theta(\mathbf{s}|\mathbf{c})] - \lambda D_{KL}[q_\phi(\mathbf{c}|\mathbf{s})||p(\mathbf{c})] \quad (6.12)$$

Although we are not optimising the exact variational lower bound, the ultimate goal of learning an effective compression model is mostly up to the reconstruction error. In Section 6, we apply $\lambda = 0.1$ for all the experiments on ASC model. Interestingly, λ controls the compression rate of the sentences which can be a good point to be explored in future work.

During the inference, we have different strategies for updating the parameters of ϕ and θ . For the parameters θ in the reconstruction model, we directly update them by the gradients:

$$\begin{aligned} \frac{\partial L}{\partial \theta} &= \mathbb{E}_{q_\phi(\mathbf{c}|\mathbf{s})} \left[\frac{\partial \log p_\theta(\mathbf{s}|\mathbf{c})}{\partial \theta} \right] \\ &\approx \frac{1}{M} \sum_m \frac{\partial \log p_\theta(\mathbf{s}|\mathbf{c}^{(m)})}{\partial \theta} \end{aligned} \quad (6.13)$$

where we draw M samples $\mathbf{c}^{(m)} \sim q_\phi(\mathbf{c}|\mathbf{s})$ independently for computing the stochastic gradients.

For the parameters ϕ in the compression model, we firstly define the learning signal,

$$l(\mathbf{s}, \mathbf{c}) = \log p_\theta(\mathbf{s}|\mathbf{c}) - \lambda(\log q_\phi(\mathbf{c}|\mathbf{s}) - \log p(\mathbf{c})).$$

Then, we update the parameters ϕ by:

$$\begin{aligned} \frac{\partial L}{\partial \phi} &= \mathbb{E}_{q_\phi(\mathbf{c}|\mathbf{s})} [l(\mathbf{s}, \mathbf{c}) \frac{\partial \log q_\phi(\mathbf{c}|\mathbf{s})}{\partial \phi}] \\ &\approx \frac{1}{M} \sum_m [l(\mathbf{s}, \mathbf{c}^{(m)}) \frac{\partial \log q_\phi(\mathbf{c}^{(m)}|\mathbf{s})}{\partial \phi}] \end{aligned} \quad (6.14)$$

However, this gradient estimator has a big variance because the learning signal $l(\mathbf{s}, \mathbf{c}^{(m)})$ relies on the samples from $q_\phi(\mathbf{c}|\mathbf{s})$. Therefore, following the REINFORCE algorithm, we introduce two baselines b and $b(\mathbf{s})$, the centred learning signal and input-dependent baseline respectively, to help reduce the variance.

Here, we build an MLP to implement the input-dependent baseline $b(\mathbf{s})$. During training, we learn the two baselines by minimising the expectation:

$$\mathbb{E}_{q_\phi(\mathbf{c}|\mathbf{s})} [(l(\mathbf{s}, \mathbf{c}) - b - b(\mathbf{s}))^2]. \quad (6.15)$$

Hence, the gradients w.r.t. ϕ are derived as,

$$\frac{\partial L}{\partial \phi} \approx \frac{1}{M} \sum_m (l(\mathbf{s}, \mathbf{c}^{(m)}) - b - b(\mathbf{s})) \frac{\partial \log q_\phi(\mathbf{c}^{(m)}|\mathbf{s})}{\partial \phi} \quad (6.16)$$

which is basically a likelihood-ratio estimator.

6.4 Forced Attention Sentence Compression Model (FSC)

In order to further boost the performance on sentence compression, we employ a supervised **forced-attention sentence compression** model (FSC) trained on labelled data to teach the ASC model to generate compression sentences. The FSC model shares the pointer network of the ASC model and combines a softmax output layer over the whole vocabulary. Therefore, while training on the sentence-compression pairs, it is able to balance copying a word from the source sentence with generating it from the background distribution. More importantly, by jointly training on the

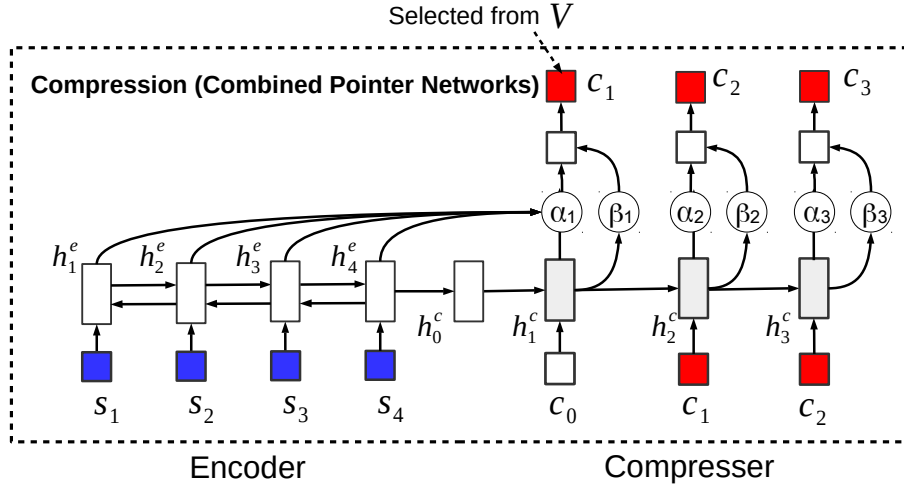


Figure 6.3: Forced Attention Sentence Compression Model

labelled and unlabelled datasets, this shared pointer network enables the model to work in a semi-supervised scenario. In this case, the FSC teaches the ASC to generate reasonable samples, while the pointer network trained on a large unlabelled data set helps the FSC model to perform better abstractive summarisation.

In neural variational inference, the effectiveness of training largely depends on the quality of the inference network gradient estimator. Although we introduce a biased estimator by using pointer networks, it is still very difficult for the compression model to generate reasonable natural language sentences at the early stage of learning, which results in high-variance for the gradient estimator. Here, we introduce our supervised forced-attention sentence compression (FSC) model to teach the compression model to generate coherent compressed sentences.

Neither directly replicating the pointer network of ASC model, nor using a typical sequence-to-sequence model, the FSC model employs a forced-attention strategy (Figure 6.3) that encourages the compressor to select words appearing in the source sentence but keeps the original full output vocabulary V . The forced-attention strategy is basically a combined pointer network that chooses whether to select a word from the source sentence \mathbf{s} or to predict a word from V at each recurrent state. Hence, the combined pointer network learns to copy the source words while predicting the word sequences of compressions. By sharing the pointer networks between the ASC and FSC model, the biased estimator obtains further positive biases by training on a small set of labelled source-compression pairs.

Here, the FSC model makes use of the compression model (Eq. 6.1 to 6.4) in the

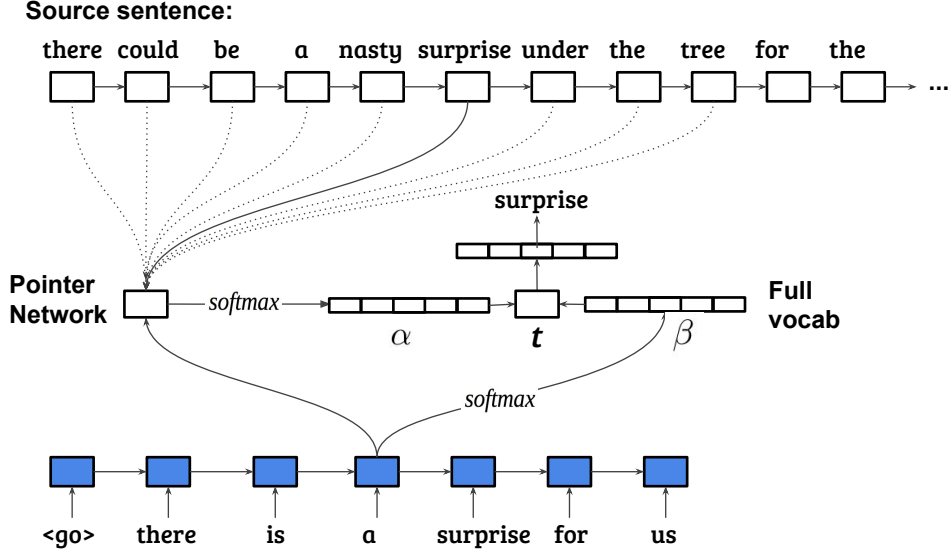


Figure 6.4: Example of forced-attention sentence compression.

ASC model,

$$\alpha_j = \text{softmax}(\mathbf{u}_j), \quad (6.17)$$

where $\alpha_j(i)$, $i \in (1, \dots, |\mathbf{s}|)$ denotes the probability of selecting \mathbf{s}_i as the prediction for \mathbf{c}_j .

On the basis of the pointer network, we further introduce the probability of predicting \mathbf{c}_j that is selected from the full vocabulary,

$$\beta_j = \text{softmax}(\mathbf{W}\mathbf{h}_j^c), \quad (6.18)$$

where $\beta_j(w)$, $w \in (1, \dots, |V|)$ denotes the probability of selecting the w th from V as the prediction for \mathbf{c}_j . To combine these two probabilities in the RNN, we define a selection factor \mathbf{t} for each state output, which computes the semantic similarities between the current state and the attention vector,

$$\boldsymbol{\eta}_j = \sum_i^{|\mathbf{s}|} \alpha_j(i) \mathbf{h}_i^e \quad (6.19)$$

$$\mathbf{t}_j = \sigma(\boldsymbol{\eta}_j^T \mathbf{M}\mathbf{h}_j^c). \quad (6.20)$$

Hence, the probability distribution over compressed words is defined as,

$$p(\mathbf{c}_j | \mathbf{c}_{1:j-1}, \mathbf{s}) = \begin{cases} \mathbf{t}_j \alpha_j(i) + (1 - \mathbf{t}_j) \beta_j(\mathbf{c}_j), & \mathbf{c}_j = \mathbf{s}_i \\ (1 - \mathbf{t}_j) \beta_j(\mathbf{c}_j), & \mathbf{c}_j \in \mathbf{s} \end{cases} \quad (6.21)$$

Essentially, the FSC model is the extended compression model of ASC by incorporating the pointer network with a softmax output layer over the full vocabulary. So

we employ ϕ to denote the parameters of the FSC model $p_\phi(\mathbf{c}|\mathbf{s})$, which covers the parameters of the variational distribution $q_\phi(\mathbf{c}|\mathbf{s})$.

For example, in Figure 6.4, every time when the model generates a word for the sentence compression, the selection factor \mathbf{t} balance the probability of directly choosing a word that appeared in the source sentence and the probability of generating the word from the full vocabulary.

6.4.1 Semi-supervised Learning

As the auto-encoding sentence compression (ASC) model grants the ability to make use of an unlabelled dataset, we explore a semi-supervised training framework for the ASC and FSC models. In this scenario we have a labelled dataset that contains source-compression parallel sentences, $(\mathbf{s}, \mathbf{c}) \in \mathbb{L}$, and an unlabelled dataset that contains only source sentences $\mathbf{s} \in \mathbb{U}$. The FSC model is trained on \mathbb{L} so that we are able to learn the compression model by maximising the log-probability,

$$F = \sum_{(\mathbf{c}, \mathbf{s}) \in \mathbb{L}} \log p_\phi(\mathbf{c}|\mathbf{s}). \quad (6.22)$$

While the ASC model is trained on \mathbb{U} , where we maximise the modified variational lower bound,

$$L = \sum_{\mathbf{s} \in \mathbb{U}} (\mathbb{E}_{q_\phi(\mathbf{c}|\mathbf{s})} [\log p_\theta(\mathbf{s}|\mathbf{c})] - \lambda D_{KL}[q_\phi(\mathbf{c}|\mathbf{s})||p(\mathbf{c})]). \quad (6.23)$$

The joint objective function of the semi-supervised learning is,

$$\begin{aligned} J = & \sum_{\mathbf{s} \in \mathbb{U}} (\mathbb{E}_{q_\phi(\mathbf{c}|\mathbf{s})} [\log p_\theta(\mathbf{s}|\mathbf{c})] - \lambda D_{KL}[q_\phi(\mathbf{c}|\mathbf{s})||p(\mathbf{c})]) \\ & + \sum_{(\mathbf{c}, \mathbf{s}) \in \mathbb{L}} \log p_\phi(\mathbf{c}|\mathbf{s}). \end{aligned} \quad (6.24)$$

Hence, the pointer network is trained on both unlabelled data, \mathbb{U} , and labelled data, \mathbb{L} , by a mixed criterion of REINFORCE and cross-entropy.

6.4.2 Variance Reduction

From the perspective of generative models, a significant contribution of our work is a process for reducing variance for discrete sampling-based variational inference. The first step is to introduce two baselines in the control variates method due to the fact that the reparameterisation trick is not applicable for discrete latent variables. However it is the second step of using a pointer network as the biased estimator that

makes the key contribution. This results in a much smaller state space, bounded by the length of the source sentence (mostly between 20 and 50 tokens), compared to the full vocabulary. The final step is to apply the FSC model to transfer the knowledge learned from the supervised data to the pointer network. This further reduces the sampling variance by acting as a sort of bootstrap or constraint on the unsupervised latent space which could encode almost anything but which thus becomes biased towards matching the supervised distribution. By using these variance reduction methods, the ASC model is able to carry out effective variational inference for the latent language model so that it learns to summarise the sentences from the large unlabelled training data.

In a different vein, according to the reinforcement learning interpretation of sequence level training [83], the compression model of the ASC model acts as an agent which iteratively generates words (takes actions) to compose the compression sentence and the reconstruction model acts as the reward function evaluating the quality of the compressed sentence which is provided as a reward signal. [83] presents a thorough empirical evaluation on three different NLP tasks by using additional sequence-level reward (BLEU and Rouge-2) to train the models. In the context of this chapter, we apply a variational lower bound (mixed reconstruction error and KL divergence regularisation) instead of the explicit Rouge score. Thus the ASC model is granted the ability to explore unlimited unlabelled data resources. In addition we introduce a supervised FSC model to teach the compression model to generate stable sequences instead of starting with a random policy. In this case, the pointer network that bridges the supervised and unsupervised model is trained by a mixed criterion of REINFORCE and cross-entropy in an incremental learning framework. Eventually, according to the experimental results, the joint ASC and FSC model is able to learn a robust compression model by exploring both labelled and unlabelled data, which outperforms the other single discriminative compression models that are only trained by cross-entropy reward signal.

6.5 Experiments

The proposed models are evaluated on the standard *Gigaword*³ sentence compression dataset. This dataset was generated by pairing the headline of each article with its first sentence to create a source-compression pair. [87] provided scripts⁴ to filter out

³<https://catalog.ldc.upenn.edu/LDC2012T21>

⁴<https://github.com/facebook/NAMAS>

Model	Data		Recall			Precision			F-1		
	L	U	R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L
FSC	500K	-	30.817	10.861	28.263	22.357	7.998	20.520	23.415	8.156	21.468
ASC+FSC ₁	500K	500K	29.117	10.643	26.811	28.558	10.575	26.344	26.987	9.741	24.874
ASC+FSC ₂	500K	3.8M	28.236	10.359	26.218	30.112	11.131	27.896	27.453	9.902	25.452
FSC	1M	-	30.889	11.645	28.257	27.169	10.266	24.916	26.984	10.028	24.711
ASC+FSC ₁	1M	1M	30.490	11.443	28.097	28.109	10.799	25.943	27.258	10.189	25.148
ASC+FSC ₂	1M	3.8M	29.034	10.780	26.801	31.037	11.521	28.658	28.336	10.313	26.145
FSC	3.8M	-	30.112	12.436	27.889	34.135	13.813	31.704	30.225	12.258	28.035
ASC+FSC ₂	3.8M	3.8M	29.946	12.558	27.805	35.538	14.699	32.972	30.568	12.553	28.366

Table 6.1: Extractive Summarisation Performance.

outliers, resulting in roughly 3.8M training pairs, a 400K validation set, and a 400K test set. In the following experiments all models are trained on the training set with different data sizes⁵ and tested on a 2K subset, which is identical to the test set used by [87] and [80]. We decode the sentences by $k = 5$ Beam search and test with full-length Rouge score.

For the ASC and FSC models, we use 256 for the dimension of both hidden units and lookup tables. In the ASC model, we apply a 3-layer bidirectional RNN with skip connections as the encoder, a 3-layer RNN pointer network with skip connections as the compressor, and a 1-layer vanilla RNN with soft attention as the decoder. The language model prior is trained on the article sentences of the full training set using a 3-layer vanilla RNN with 0.5 dropout. To lower the computational cost, we apply different vocabulary sizes for encoder and compressor (119,506 and 68,897) which corresponds to the settings of [87]. Specifically, the vocabulary of the decoder is filtered by taking the most frequent 10,000 words from the vocabulary of the encoder, where the rest of the words are tagged as ‘<unk>’. In further consideration of efficiency, we use only one sample for the gradient estimator. We optimise the model by Adam [53] with a 0.0002 learning rate and 64 sentences per batch. The model converges in 5 epochs. Except for the pre-trained language model, we do not use dropout or embedding initialisation for ASC and FSC models.

6.5.1 Extractive Summarisation

The first set of experiments evaluate the models on extractive summarisation. Here, we denote the joint models by ASC+FSC₁ and ASC+FSC₂ where ASC is trained on

⁵The hyperparameters were tuned on the validation set to maximise the perplexity of the summaries rather than the reconstructed source sentences.

Model	Data		Recall			Precision			F-1		
	L	U	R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L
FSC	500K	-	27.147	10.039	25.197	33.781	13.019	31.288	29.074	10.842	26.955
ASC+FSC ₁	500K	500K	27.067	10.717	25.239	33.893	13.678	31.585	29.027	11.461	27.072
ASC+FSC ₂	500K	3.8M	27.662	11.102	25.703	35.756	14.537	33.212	30.140	12.051	27.99
FSC	1M	-	28.521	11.308	26.478	33.132	13.422	30.741	29.580	11.807	27.439
ASC+FSC ₁	1M	1M	28.333	11.814	26.367	35.860	15.243	33.306	30.569	12.743	28.431
ASC+FSC ₂	1M	3.8M	29.017	12.007	27.067	36.128	14.988	33.626	31.089	12.785	28.967
FSC	3.8M	-	31.148	13.553	28.954	36.917	16.127	34.405	32.327	14.000	30.087
ASC+FSC ₂	3.8M	3.8M	32.385	15.155	30.246	39.224	18.382	36.662	34.156	15.935	31.915

Table 6.2: Abstractive Summarisation Performance.

unlabelled data and FSC is trained on labelled data. The ASC+FSC₁ model employs equivalent sized labelled and unlabelled datasets, where the article sentences of the unlabelled data are the same article sentences in the labelled data, so there is no additional unlabelled data applied in this case. The ASC+FSC₂ model employs the full unlabelled dataset in addition to the existing labelled dataset, which is the true semi-supervised setting.

Table 6.1 presents the test Rouge score on extractive compression. The extractive summaries of these models are decoded by the pointer network (i.e the shared component of the ASC and FSC models). R-1, R-2 and R-L represent the Rouge-1, Rouge-2 and Rouge-L score respectively. We can see that the ASC+FSC₁ model achieves significant improvements on F-1 scores when compared to the supervised FSC model only trained on labelled data. Moreover, fixing the labelled data size, the ASC+FSC₂ model achieves better performance by using additional unlabelled data than the ASC+FSC₁ model, which means the semi-supervised learning works in this scenario. Interestingly, learning on the unlabelled data largely increases the precisions (though the recalls do not benefit from it) which leads to significant improvements on the F-1 Rouge scores. And surprisingly, the extractive ASC+FSC₁ model trained on full labelled data outperforms the abstractive NABS [87] baseline model (in Table 6.4).

6.5.2 Abstractive Summarisation

The second set of experiments evaluate performance on abstractive summarisation (Table 6.2). The abstractive summaries of these models are decoded by the combined pointer network (i.e. the shared pointer network together with the softmax output

Model	Labelled Data	Perplexity
Bag-of-Word (BoW)	3.8M	43.6
Convolutional (TDNN)	3.8M	35.9
Attention-Based (NABS) [87]	3.8M	27.1
Forced-Attention (FSC)	3.8M	18.6
Auto-encoding (ASC+FSC ₂)	3.8M	16.6

Table 6.3: Comparison on validation perplexity.

Model	Labelled Data	R-1	R-2	R-L
NABS [87]	3.8M	29.78	11.89	26.97
S2S [80]	3.8M	33.17	16.02	30.98
ASC + FSC ₂	500K	30.14	12.05	27.99
ASC + FSC ₂	1M	31.09	12.79	28.97
ASC + FSC ₂	3.8M	34.17	15.94	31.92

Table 6.4: Comparison on test Rouge scores.

layer over the full vocabulary). Consistently, we see that adding the generative objective to the discriminative model (ASC+FSC₁) results in a notable boost on all the Rouge scores, while employing extra unlabelled data increase performance further (ASC+FSC₂). This validates the effectiveness of transferring the knowledge learned on unlabelled data to the supervised abstractive summarisation.

In Figure 6.5, we present the validation perplexity to compare the abilities of the three models to learn the compression languages. BoW, TDNN and NABS are the baseline neural compression models with different encoders in [87]. The ASC+FSC₁(red) employs the same dataset for unlabelled and labelled training, while the ASC+FSC₂(black) employs the full unlabelled dataset. Here, the joint ASC+FSC₁ model obtains better perplexities than the single discriminative FSC model, but there is not much difference between ASC+FSC₁ and ASC+FSC₂ when the size of the labelled dataset grows. From the perspective of language modelling, the generative ASC model indeed helps the discriminative model learn to generate good summary sentences. Table 6.3 displays the validation perplexities of the benchmark models, where the joint ASC+FSC₁ model trained on the full labelled and unlabelled datasets performs the best on modelling compression languages.

Table 6.4 compares the test Rouge score on abstractive summarisation. Encouragingly, the semi-supervised model ASC+FSC₂ outperforms the baseline model NABS when trained on 500K supervised pairs, which is only about an eighth of the supervised data. In [80], the authors exploit the full limits of discriminative RNN

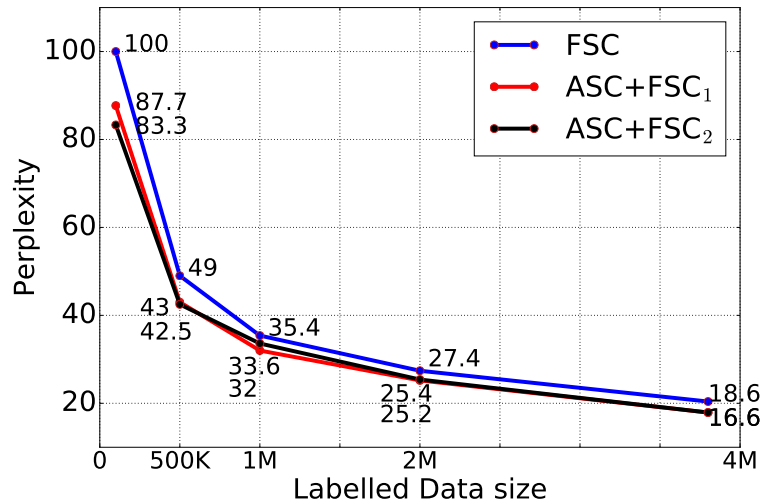


Figure 6.5: Perplexity on validation dataset.

encoder-decoder models by incorporating a sampled softmax, expanded vocabulary, additional lexical features, and combined pointer networks⁶, which yields the best performance listed in Table 6.4. However, when all the data is employed with the mixed objective ASC+FSC₁ model, the result is significantly better than this previous state-of-the-art. As the semi-supervised ASC+FSC₂ model can be trained on unlimited unlabelled data, there is still significant space left for further performance improvements.

Table D.2 presents the examples of the compression sentences decoded by the joint model ASC+FSC₁ and the FSC model trained on the full dataset. **src** and **ref** are the source and reference sentences provided in the test set. **asc_a** and **asc_e** are the abstractive and extractive compression sentences decoded by the joint model ASC+FSC₁, and **fsc_a** denotes the abstractive compression obtained by the FSC model.

6.5.3 Discussion

From the perspective of generative models, a significant contribution of our work is the development of a process for reducing variance for discrete sampling-based variational inference. The first step is to introduce two baselines in the control variates method due to the fact that the reparameterisation trick is not applicable for discrete latent variables. However it is the second step of using a pointer network as the biased estimator that makes the key contribution. This results in a much smaller state space, bounded by the length of the source sentence (mostly between 20 and

⁶The idea of the combined pointer networks is similar to the FSC model, but the implementations are slightly different.

50 tokens), compared to the full vocabulary. The final step is to apply the FSC model to transfer the knowledge learned from the supervised data to the pointer network. This further reduces the sampling variance by acting as a sort of bootstrap or constraint on the unsupervised latent space which could encode almost anything but which thus becomes biased towards matching the supervised distribution. By using these variance reduction methods, the ASC model is able to carry out effective variational inference for the latent language model so that it learns to summarise the sentences from the large unlabelled training data. However, without incorporating the FSC model that provides explicit learning signals, the ASC model is unable to learn effective summarisation by just learning in an unsupervised fashion, which is basically due to high variance issue caused by the large vocabulary.

In a different vein, according to the reinforcement learning interpretation of sequence level training [83], the compression model of the ASC model acts as an agent which iteratively generates words (takes actions) to compose the compression sentence and the reconstruction model acts as the reward function evaluating the quality of the compressed sentence which is provided as a reward signal. [83] presents a thorough empirical evaluation on three different NLP tasks by using additional sequence-level reward (BLEU and Rouge-2) to train the models. In the context of this paper, we apply a variational lower bound (mixed reconstruction error and KL divergence regularisation) instead of the explicit Rouge score. Thus the ASC model is granted the ability to explore unlimited unlabelled data resources. In addition we introduce a supervised FSC model to teach the compression model to generate stable sequences instead of starting with a random policy. In this case, the pointer network that bridges the supervised and unsupervised model is trained by a mixed criterion of REINFORCE and cross-entropy in an incremental learning framework. Eventually, according to the experimental results, the joint ASC and FSC model is able to learn a robust compression model by exploring both labelled and unlabelled data, which outperforms the other single discriminative compression models that are only trained by cross-entropy reward signal.

6.6 Summary

In this chapter, a generative model is introduced for jointly modelling pairs of sequences and evaluated its efficacy on the task of sentence compression. The neural variational inference framework provided an effective inference algorithm for this approach and also allowed us to explore combinations of discriminative (FSC) and

generative (ASC) compression models. The evaluation results show that supervised training of the combination of these models improves upon the state-of-the-art performance for the Gigaword compression dataset. When we train the supervised FSC model on a small amount of labelled data and the unsupervised ASC model on a large set of unlabelled data the combined model is able to outperform previously reported benchmarks trained on a great deal more supervised data. These results demonstrate that we are able to model language as a discrete latent variable in the framework of variational auto-encoders and that the resultant generative model is able to effectively exploit both supervised and unsupervised data in sequence-to-sequence tasks.

Chapter 7

Latent Intention Dialogue Models

In this chapter, we present a Latent Intention Dialogue Model (LIDM) that employs a discrete latent variable to learn underlying dialogue intentions in the framework of neural variational inference. Different from the latent sentence compression model introduced in the previous chapter, where the discrete latent variables represent explicit words from the full vocabulary, the discrete latent intention in this context is interpreted as actions guiding the generation of machine responses and can be further refined autonomously by reinforcement learning. Hence, by combining this latent intention with the RNN decoder, we expect to decompose the learning of generating natural language and the internal dialogue decision-making. The experimental evaluation of LIDM shows that the model out-performs published benchmarks for both corpus-based and human evaluation, demonstrating the effectiveness of discrete latent variable models for learning goal-oriented dialogues.

7.1 Introduction

Recurrent neural networks (RNNs) have shown impressive results in modeling generation tasks that have a sequential structured output form, such as machine translation [101, 6], caption generation [49, 121], and natural language generation [114, 52]. These discriminative models are trained to learn only a conditional output distribution over strings and despite the sophisticated architectures and conditioning mechanisms used to ensure salience, they are not able to model the underlying actions needed to generate natural dialogues, which are the latent intentions in our case. As a consequence, these sequence-to-sequence models are limited in their ability to exhibit the intrinsic variability and stochasticity of natural dialogue. For example both goal-oriented dialogue systems [116, 12] and sequence-to-sequence learning chatbots [106, 93, 89] struggle to generate diverse yet causal responses [66, 88]. Here, we

introduce a latent variable model – Latent Intention Dialogue Model (LIDM) – for learning the complex distribution of communicative intentions in goal-oriented dialogues. The latent intentions are modelled as discrete latent variable, which can be interpreted by using the content words in the generated machine responses (Table 7.1). In addition, an identical intention is able to generate multiple different machine responses conditioned on different conversational contexts.

In this model, the latent variable representing dialogue intention can be considered as the autonomous decision-making center of a dialogue agent for composing appropriate machine responses, which is inferred from user input utterances. Based on the dialogue context, the agent draws a sample as the intention which then guides the natural language response generation. Firstly, in the framework of neural variational inference, we construct an inference network to approximate the posterior distribution over the latent intention. Then, by sampling the intentions for each response, we are able to directly learn a basic intention distribution on a human-human dialogue corpus by optimising the variational lower bound. To further reduce the variance, we utilize a labeled subset of the corpus in which the labels of intentions are automatically generated by clustering. Then, the latent intention distribution can be learned in a semi-supervised fashion, where the learning signals are either from the direct supervision (labeled set) or the variational lower bound (unlabeled set).

From the perspective of reinforcement learning, the latent intention distribution can be interpreted as the intrinsic policy that reflects human decision-making under a particular conversational scenario. Based on the initial policy (latent intention distribution) learned from the semi-supervised variational inference framework, the model can refine its strategy easily against alternative objectives using policy gradient-based reinforcement learning. In some way, this is analagous to the training process used in AlphaGo [94] for the game of Go. Based on LIDM, we show that different learning paradigms can be brought together under the same framework to bootstrap the development of a dialogue agent [69, 118].

7.2 Related Work

Modeling chat-based dialogues [89, 93] as a sequence-to-sequence learning [101] problem is a common theme in the deep learning community. Vinyals and Le [106] has demonstrated a seq2seq-based model trained on a huge amount of conversation corpora which learns interesting replies conditioned on different user queries. However, due to an inability to model dialogue context, these models generally suffer from the

generic response problem [66, 88]. Several approaches have been proposed to mitigate this issue, such as modeling the persona [67], reinforcement learning [69], and introducing continuous latent variables [88, 16]. While in our case, we not only make use of the latent variable to inject stochasticity for generating natural and diverse machine responses but also model the hidden dialogue intentions explicitly. This combines the merits of reinforcement learning and generative models.

In another direction, goal-oriented dialogue systems typically adopt the POMDP framework [124] and break down the development of the dialogue systems into a pipeline of modules: natural language understanding [34], dialogue management [26], and natural language generation [114]. These system modules communicate through a dialogue act formalism [104], which in effect constitute a fixed set of handcrafted intentions. This limits the ability of such systems to scale to more complex tasks. In contrast, the LIDM directly infers all underlying dialogue intentions from data and can handle intention distributions with long tails by measuring similarities against the existing ones during variational inference. Modeling of end-to-end goal-oriented dialogue systems has also been studied recently [113, 116, 12], however, these models are typically deterministic and rely on decoder supervision signals to fine-tune a large set of model parameters.

A lot of research works have focused on combining different learning paradigms and signals to bootstrap performance. For example, semi-supervised learning [54] has been applied in the sample-based neural variational inference framework as a way to reduce sample variance. In practice, this relies on a discrete latent variable as the vehicle for the supervision labels, for example the sentence compression models introduced in Chapter 6 and the semantic parsing VAE model [58]. As in reinforcement learning, which has been a very common learning paradigm in dialogue systems [26, 98, 68], the policy is also parameterised by a discrete set of actions. As a consequence, the LIDM, which parameterises the intention space via a discrete latent variable, can automatically enjoy the benefit of bootstrapping from signals coming from different learning paradigms. In addition, self-supervised learning [95] (or distant, weak supervision) as a simple way to generate automatic labels by heuristics is popular in many NLP tasks and has been applied to memory networks [38] and neural dialogue systems [113] recently. Since there is no additional effort required in labeling, it can also be viewed as a method for bootstrapping.

Goal-oriented dialogue¹ [124] aims at building models that can help users to complete certain tasks via natural language interaction. Given a user input utterance

¹Like most of the goal-oriented dialogue research, we focus on information seek type dialogues.

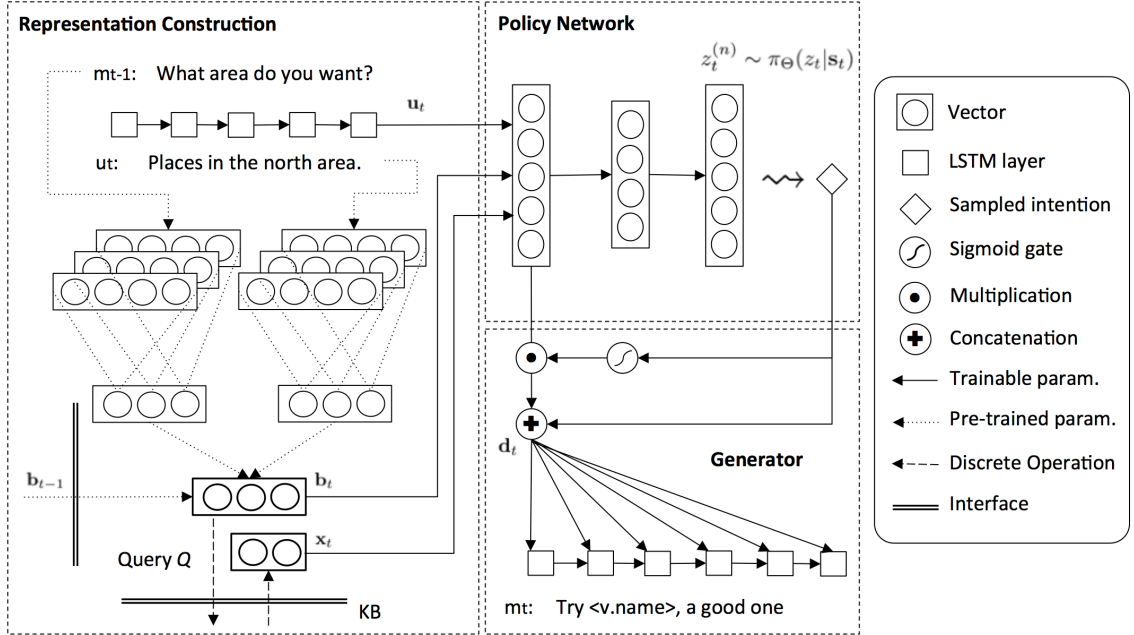


Figure 7.1: LIDM for Goal-oriented Dialogue Modeling

u_t at turn t and a knowledge base (KB), the model needs to parse the input into actionable commands Q and access the KB to search for useful information in order to answer the query. Based on the search result, the model needs to summarise its findings and reply with an appropriate response m_t in natural language.

7.3 Latent Intention Dialogue Model (LIDM)

7.3.1 Model Description

The LIDM is based on the end-to-end system architecture described in [116]. It comprises three components: (1) Representation Construction; (2) Policy Network; and (3) Generator, as shown in Figure 7.1. To capture the user’s intent and match it against the system’s knowledge, a concatenated dialogue state vector $\mathbf{s}_t = \mathbf{u}_t \oplus \mathbf{b}_t \oplus \mathbf{x}_t$ is derived from the user input u_t and the knowledge base KB: \mathbf{u}_t is the distributed utterance representation, which is formed by encoding the user utterance² u_t with a bidirectional LSTM [43] and concatenating the final stage hidden states together,

$$\mathbf{u}_t = \text{biLSTM}_{\Theta}(u_t). \quad (7.1)$$

²All sentences are pre-processed by delexicalisation [35] where slot-value specific words are replaced with their corresponding generic tokens based on an ontology.

The belief vector \mathbf{b}_t , which is a concatenation of a set of probability distributions over domain specific slot-value pairs, is extracted by a set of pre-trained RNN-CNN belief trackers [116, 35], in which u_t and m_{t-1} are processed by two different CNNs as shown in Figure 7.1,

$$\mathbf{b}_t = \text{RNN-CNN}(u_t, m_{t-1}, \mathbf{b}_{t-1}) \quad (7.2)$$

where m_{t-1} is the preceding machine response and \mathbf{b}_{t-1} is the preceding belief vector. They are included to model the current turn of the discourse and the long-term dialogue context, respectively. Based on the belief vector, a query Q is formed by taking the union of the maximum values of each slot. Q is then used to search the internal KB and return a vector \mathbf{x}_t representing the degree of matching in the KB. This is produced by counting all the matching values and re-structuring it into a six-bin one-hot vector. Among the three vectors that comprise the dialogue state \mathbf{s}_t , \mathbf{u}_t is completely trainable from data, \mathbf{b}_t is pre-trained using a separate objective function, and \mathbf{x}_t is produced by a discrete database accessing operation. For more details about the belief trackers and database operation refer to Wen et al [113, 116].

Conditioning on the state \mathbf{s}_t , the policy network parameterises the latent intention z_t by a single layer MLP,

$$\pi_{\Theta}(z_t|\mathbf{s}_t) = \text{softmax}(\mathbf{W}_2^T \cdot \tanh(\mathbf{W}_1^T \mathbf{s}_t + \mathbf{b}_1) + \mathbf{b}_2) \quad (7.3)$$

where \mathbf{W}_1 , \mathbf{b}_1 , \mathbf{W}_2 , \mathbf{b}_2 are model parameters. Since $\pi_{\Theta}(z_t|\mathbf{s}_t)$ is a discrete conditional probability distribution based on dialogue state, we can also interpret the policy network here as a latent dialogue management component in the traditional POMDP-based framework [124, 26]. A latent intention $z_t^{(n)}$ (or an action in the reinforcement learning literature) can then be sampled from the conditional distribution,

$$z_t^{(n)} \sim \pi_{\Theta}(z_t|\mathbf{s}_t). \quad (7.4)$$

This sampled intention (or action) $z_t^{(n)}$ and the state vector \mathbf{s}_t can then be combined into a control vector \mathbf{d}_t , which is used to govern the generation of the system response based on a conditional LSTM language model,

$$\mathbf{d}_t = \mathbf{W}_4^T \mathbf{z}_t \oplus [\text{sigmoid}(\mathbf{W}_3^T \mathbf{z}_t + \mathbf{b}_3) \cdot \mathbf{W}_5^T \mathbf{s}_t] \quad (7.5)$$

$$p_{\Theta}(m_t|\mathbf{s}_t, z_t) = \prod_j p(w_{j+1}^t | w_j^t, \mathbf{h}_{j-1}^t, \mathbf{d}_t) \quad (7.6)$$

where \mathbf{b}_3 and $\mathbf{W}_{3\sim 5}$ are parameters, \mathbf{z}_t is the 1-hot representation of $z_t^{(n)}$, w_j^t is the last output token (i.e. a word, a delexicalised² slot name or a delexicalised² slot

value), and \mathbf{h}_{j-1}^t is the decoder’s last hidden state. Note in Equation 7.5 the degree of information flow from the state vector is controlled by a sigmoid gate whose input signal is the sampled intention $z_t^{(n)}$. This prevents the decoder from over-fitting to the deterministic state information and forces it to take the sampled stochastic intention into account. The LIDM can then be formally written down in its parameterised form with parameter set Θ ,

$$p_{\Theta}(m_t|\mathbf{s}_t) = \sum_{z_t} p_{\Theta}(m_t|z_t, \mathbf{s}_t)\pi_{\Theta}(z_t|\mathbf{s}_t). \quad (7.7)$$

7.3.2 Inference

To carry out inference for the LIDM, we apply an inference network $q_{\Phi}(z_t|\mathbf{s}_t, m_t)$ to approximate the posterior distribution $p(z_t|\mathbf{s}_t, m_t)$ so that we can optimise the variational lower bound of the joint probability in the neural variational inference framework introduced in Chapter 2. We can then derive the modified variational lower bound as,

$$\begin{aligned} \mathcal{L} &= \mathbb{E}_{q_{\Phi}(z_t)}[\log p_{\Theta}(m_t|z_t, \mathbf{s}_t)] - \lambda D_{KL}(q_{\Phi}(z_t)||\pi_{\Theta}(z_t|\mathbf{s}_t)) \\ &\leq \log \sum_{z_t} p_{\Theta}(m_t|z_t, \mathbf{s}_t)\pi_{\Theta}(z_t|\mathbf{s}_t) \\ &= \log p_{\Theta}(m_t|\mathbf{s}_t) \end{aligned} \quad (7.8)$$

where $q_{\Phi}(z_t)$ is a shorthand for $q_{\Phi}(z_t|\mathbf{s}_t, m_t)$. Note that we use a modified version of the lower bound here by incorporating a trade-off factor λ [37]. The inference network $q_{\Phi}(z_t|\mathbf{s}_t, m_t)$ is then constructed by

$$q_{\Phi}(z_t|\mathbf{s}_t, m_t) = \text{Multi}(\mathbf{o}_t) = \text{softmax}(\mathbf{W}_6\mathbf{o}_t) \quad (7.9)$$

$$\mathbf{o}_t = \text{MLP}_{\Phi}(\mathbf{b}_t, \mathbf{x}_t, \mathbf{u}_t, \mathbf{m}_t) \quad (7.10)$$

$$\mathbf{u}_t = \text{biLSTM}_{\Phi}(u_t), \mathbf{m}_t = \text{biLSTM}_{\Phi}(m_t) \quad (7.11)$$

where \mathbf{o}_t is the joint representation, and both \mathbf{u}_t and \mathbf{m}_t are modeled by a bidirectional LSTM network. Although both $q_{\Phi}(z_t|\mathbf{s}_t, m_t)$ and $\pi_{\Theta}(z_t|\mathbf{s}_t)$ are modelled as parameterised multinomial distributions, the approximation $q_{\Phi}(z_t|\mathbf{s}_t, m_t)$ only functions during inference by producing samples to compute the stochastic gradients, while $\pi_{\Theta}(z_t|\mathbf{s}_t)$ is the generative distribution that generates the required samples for composing the machine response.

Based on the samples $z_t^{(n)} \sim q_\Phi(z_t|\mathbf{s}_t, m_t)$, we use different strategies to alternately optimise the parameters Θ and Φ against the variational lower bound (Equation 7.8). To do this, we further divide Θ into two sets $\Theta = \{\Theta_1, \Theta_2\}$. Parameters Θ_1 on the decoder side are directly updated by back-propagating the gradients,

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \Theta_1} &= \mathbb{E}_{q_\Phi(z_t|\mathbf{s}_t, m_t)} \left[\frac{\partial \log p_{\Theta_1}(m_t|z_t, \mathbf{s}_t)}{\partial \Theta_1} \right] \\ &\approx \frac{1}{N} \sum_n \frac{\partial \log p_{\Theta_1}(m_t|z_t^{(n)}, \mathbf{s}_t)}{\partial \Theta_1}. \end{aligned} \quad (7.12)$$

Parameters Θ_2 in the generative model, however, are updated by minimising the KL divergence,

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \Theta_2} &= - \frac{\partial \lambda D_{KL}(q_\Phi(z_t|\mathbf{s}_t, m_t) \parallel \pi_{\Theta_2}(z_t|\mathbf{s}_t))}{\partial \Theta_2} \\ &= -\lambda \sum_{z_t} q_\Phi(z_t|\mathbf{s}_t, m_t) \frac{\partial \log \pi_{\Theta_2}(z_t|\mathbf{s}_t)}{\partial \Theta_2} \end{aligned} \quad (7.13)$$

where the entropy derivative $\partial H[q_\Phi(z_t|\mathbf{s}_t, m_t)]/\partial \Theta_2 = 0$ and therefore can be ignored. Finally, for the parameters Φ in the inference network, we firstly define the learning signal $r(m_t, z_t^{(n)}, \mathbf{s}_t)$,

$$\begin{aligned} r(m_t, z_t^{(n)}, \mathbf{s}_t) &= \log p_{\Theta_1}(m_t|z_t^{(n)}, \mathbf{s}_t) - \\ &\quad \lambda (\log q_\Phi(z_t^{(n)}|\mathbf{s}_t, m_t) - \log \pi_{\Theta_2}(z_t^{(n)}|\mathbf{s}_t)). \end{aligned} \quad (7.14)$$

Then the parameters Φ are updated by,

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \Phi} &= \mathbb{E}_{q_\Phi(a_t|\mathbf{s}_t, m_t)} \left[r(m_t, a_t, \mathbf{s}_t) \frac{\partial \log q_\Phi(a_t|\mathbf{s}_t, m_t)}{\partial \Phi} \right] \\ &\approx \frac{1}{N} \sum_n r(m_t, z_t^{(n)}, \mathbf{s}_t) \frac{\partial \log q_\Phi(z_t^{(n)}|\mathbf{s}_t, m_t)}{\partial \Phi}. \end{aligned} \quad (7.15)$$

However, this gradient estimator has a large variance because the learning signal $r(m_t, z_t^{(n)}, \mathbf{s}_t)$ relies on samples from the proposal distribution $q_\Phi(z_t|\mathbf{s}_t, m_t)$. To reduce the variance during inference, we follow the REINFORCE algorithm introduced in Chapter 2 and apply two baselines b and $b(\mathbf{s}_t)$, the centered learning signal and input dependent baseline respectively to help reduce the variance. b is a learnable constant and $b(\mathbf{s}_t) = \text{MLP}(\mathbf{s}_t)$. During training, the two baselines are updated by minimising the distance,

$$\mathcal{L}_b = \left[r(m_t, z_t^{(n)}, \mathbf{s}_t) - b - b(\mathbf{s}_t) \right]^2 \quad (7.16)$$

and the gradient w.r.t. Φ can be rewritten as

$$\frac{\partial \mathcal{L}}{\partial \Phi} \approx \frac{1}{N} \sum_n \left[r(m_t, z_t^{(n)}, \mathbf{s}_t) - b - b(\mathbf{s}_t) \right] \frac{\partial \log q_\Phi(z_t^{(n)}|\mathbf{s}_t, m_t)}{\partial \Phi}. \quad (7.17)$$

7.3.3 Semi-Supervision

Despite the steps described above for reducing the variance, there remain two major difficulties in learning latent intentions in a completely unsupervised manner: (1) the high variance of the inference network prevents it from generating sensible intention samples in the early stages of training, and (2) the overly strong discriminative power of the LSTM language model is prone to the disconnection phenomenon between the LSTM decoder and the rest of the components whereby the decoder learns to ignore the samples and focuses solely on optimising the language model. To ensure more stable training and prevent disconnection, a semi-supervised learning technique is introduced.

Inferring the latent intentions underlying utterances is similar to an unsupervised clustering task. Standard clustering algorithms can therefore be used to pre-process the corpus and generate automatic labels \hat{z}_t for part of the training examples $(m_t, \mathbf{s}_t, \hat{z}_t) \in \mathbb{L}$. Then when the model is trained on the unlabeled examples $(m_t, \mathbf{s}_t) \in \mathbb{U}$, we optimise it against the modified variational lower bound given in Equation 7.8,

$$\begin{aligned} \mathcal{L}_1 = \sum_{(m_t, \mathbf{s}_t) \in \mathbb{U}} \mathbb{E}_{q_\phi(z_t | \mathbf{s}_t, m_t)} [\log p_\theta(m_t | z_t, \mathbf{s}_t)] \\ - \lambda D_{KL}(q_\phi(z_t | \mathbf{s}_t, m_t) || \pi_\theta(z_t | \mathbf{s}_t)) \end{aligned} \quad (7.18)$$

However, when the model is updated based on examples from the labeled set $(m_t, \mathbf{s}_t, \hat{z}_t) \in \mathbb{L}$, we treat the labeled intention \hat{z}_t as an observed variable and train the model by maximising the joint log-likelihood,

$$\mathcal{L}_2 = \sum_{(m_t, \hat{z}_t, \mathbf{s}_t) \in \mathbb{L}} \log [p_\theta(m_t | \hat{z}_t, \mathbf{s}_t) \pi_\theta(\hat{z}_t | \mathbf{s}_t)] \quad (7.19)$$

The final joint objective function can then be written as $\mathcal{L}' = \alpha \mathcal{L}_1 + \mathcal{L}_2$, where α controls the trade-off between the supervised and unsupervised examples.

7.3.4 Reinforcement Learning

One of the main purposes of learning interpretable, discrete latent intention inside a dialogue system is to be able to control and refine the model’s behaviour with operational experience. The learnt generative network $\pi_\theta(z_t | \mathbf{s}_t)$ encodes the policy discovered from the underlying data distribution but this is not necessarily optimal for any specific task. Since $\pi_\theta(z_t | \mathbf{s}_t)$ is a parameterised policy network itself, any

policy gradient-based reinforcement learning algorithm [119, 57] can be used to fine-tune the initial policy against other objective functions that we are more interested in.

Based on the initial policy $\pi_{\Theta}(z_t|\mathbf{s}_t)$, we revisit the training dialogues and update parameters based on the following strategy: when encountering unlabeled examples \mathbb{U} at turn t the system samples an action from the learnt policy $z_t^{(n)} \sim \pi_{\Theta}(z_t|\mathbf{s}_t)$ and receives a reward $r_t^{(n)}$. Conditioning on these, we can directly fine-tune a subset of the model parameters Θ' by the policy gradient method,

$$\frac{\partial \mathcal{J}}{\partial \Theta'} \approx \frac{1}{N} \sum_n r_t^{(n)} \frac{\partial \log \pi_{\Theta}(z_t^{(n)}|\mathbf{s}_t)}{\partial \Theta'} \quad (7.20)$$

where $\Theta' = \{\mathbf{W}_1, \mathbf{b}_1, \mathbf{W}_2, \mathbf{b}_2\}$ is the MLP that parameterises the policy network (Equation 7.3). However, when a labeled example $\in \mathbb{L}$ is encountered we force the model to take the labeled action $z_t^{(n)} = \hat{z}_t$ and update the parameters by Equation 7.20 as well. Unlike Li et al [69] where the whole model is refined end-to-end using RL, updating only Θ' effectively allows us to refine only the decision-making of the system and avoid the problem of over-fitting.

7.4 Experiments

7.4.1 Dataset & Setup

We explored the properties of the LIDM model³ using the CamRest676 corpus⁴ collected by Wen et al [116], in which the task of the system is to assist users to find a restaurant in the Cambridge, UK area. The corpus was collected based on a modified Wizard of Oz [50] online data collection. Workers were recruited on Amazon Mechanical Turk and asked to complete a task by carrying out a conversation, alternating roles between a user and a wizard. There are three informable slots (*food*, *pricerange*, *area*) that users can use to constrain the search and six requestable slots (*address*, *phone*, *postcode* plus the three informable slots) that the user can ask a value for once a restaurant has been offered. There are 676 dialogues in the dataset (including both finished and unfinished dialogues) and approximately 2750 conversational turns in total. The database contains 99 unique restaurants.

To make a direct comparison with prior work we follow the same experimental setup as in Wen et al [113, 116]. The corpus was partitioned into training, validation,

³Will be available at <https://github.com/shawnwun/NNDIAL>

⁴<https://www.repository.cam.ac.uk/handle/1810/260970>

ID	#	content words
0	138	thank, goodbye
1	91	welcome, goodbye
3	42	phone, address, [v.phone],[v.address]
14	17	address, [v.address]
31	9	located, area, [v.area]
34	9	area, would, like
46	7	food, serving, restaurant, [v.food]
85	4	help, anything, else

Table 7.1: An example of the automatically labeled response seed set for semi-supervised learning during variational inference.

and test sets in the ratio 3:1:1. The LSTM hidden layer sizes were set to 50, and the vocabulary size is around 500 after pre-processing, to remove rare words and words that can be delexicalised². All the system components were trained jointly by fixing the pre-trained belief trackers and the discrete database operator with the model’s latent intention size I set to 50, 70, and 100, respectively. The trade-off constants λ and α were both set to 0.1. To produce self-labeled response clusters for semi-supervised learning of the intentions, we firstly removed function words from all the responses and clustered them according to their content words. We then assigned the responses in the i -th frequent cluster to the i -th latent dimension as its supervised set. This results in about 35% ($I = 50$) to 43% ($I = 100$) labeled responses across the whole dataset. An example of the resulting seed set is shown in Table 7.1. During inference we carried out stochastic estimation by taking one sample for estimating the stochastic gradients. The model is trained by Adam [53] and tuned (early stopping, hyper-parameters) on the held-out validation set. We alternately optimised the generative model and the inference network by fixing the parameters of one while updating the parameters of the other.

During reinforcement fine-tuning, we generated a sentence m_t from the model to replace the ground truth \hat{m}_t at each turn and define an immediate reward as whether m_t can improve the dialogue success [99] relative to \hat{m}_t , plus the sentence BLEU score [3],

$$r_t = \eta \cdot \text{sBLEU}(m_t, \hat{m}_t) + \begin{cases} 1 & m_t \text{ improves} \\ -1 & m_t \text{ degrades} \\ 0 & \text{otherwise} \end{cases} \quad (7.21)$$

where the constant η was set to 0.5. We fine-tuned the model parameters using RL for only 3 epochs. During testing, we greedily selected the most probable intention

Model	Success (%)	BLEU
Ground Truth		
Ground Truth	91.6	1.000
Published Models [113]		
NDM	76.1	0.212
NDM+Att	79.0	0.224
NDM+Att+SS	81.8	0.240
LIDM Models		
LIDM, $I = 50$	66.9	0.238
LIDM, $I = 70$	61.0	0.246
LIDM, $I = 100$	63.2	0.242
LIDM Models + RL		
LIDM, $I = 50$, +RL	82.4	0.231
LIDM, $I = 70$, +RL	81.6	0.230
LIDM, $I = 100$, +RL	84.6	0.240

Table 7.2: Corpus-based Evaluation.

and applied beam search with the beamwidth set to 10 when decoding the response. The decoding criterion was the average log-probability of tokens in the response. We then evaluated our model on task success rate [99] and BLEU score [82] as in Wen et al [113, 116] in which the model is used to predict each system response in the held-out test set.

7.4.2 Experimental Results

Table 7.2 presents the results of the corpus-based evaluation. The Ground Truth block shows the two metrics when we compute them on the human-authored responses. This sets a gold standard for the task. In the block of Published Models, there are three baseline modules developed from Wen et al [113]: (1) the vanilla neural dialogue model (NDM), (2) NDM plus an attention mechanism on the belief trackers, and (3) the attentive NDM with self-supervised sub-task neurons. The results of the LIDM model with and without RL fine-tuning are shown in the LIDM Models and the LIDM Models + RL blocks, respectively. As can be seen, the initial policy learned by fitting the latent intention to the underlying data distribution yielded reasonably good results on BLEU but did not perform well on task success when compared to their deterministic counterparts (block 2 v.s. 3). This may be due to the fact that the variational lower bound of the dataset was optimised rather than task success during variational inference. However, once RL was applied to optimise the success

Metrics	NDM	LIDM	LIDM+RL
Success	91.5%	92.0%	93.0%
Comprehension	4.21	4.40*	4.40
Naturalness	4.08	4.29*	4.28*
# of Turns	4.45	4.54	4.29

* p < 0.05

Table 7.3: Human evaluation. The significance test is based on a two-tailed student-t test, between NDM and LIDMs.

rate as part of the reward function (Equation 7.21) during the fine-tuning phase, the resulting LIDM+RL models outperformed the three baselines in terms of task success without significantly sacrificing BLEU (block 2 v.s. 4)⁵.

In order to assess the human perceived performance, we evaluated the three models (1) NDM, (2) LIDM, and (3) LIDM+RL by recruiting paid subjects on Amazon Mechanical Turk. Each judge was asked to follow a task and carried out a conversation with the machine. At the end of each conversation the judges were asked to rate and compare the model’s performance. We assessed the subjective success rate, the perceived comprehension ability and the naturalness of responses on a scale of 1 to 5. For each model, we collected 200 dialogues and averaged the scores. During human evaluation, we sampled from the top-5 intentions of the LIDM models and decoded a response based on the sample. The result is shown in Table 7.3. One interesting fact to note is that although the LIDM did not perform well on the corpus-based task success metric, the human judges rated its subjective success almost indistinguishably from the others. This discrepancy between the two experiments arises mainly from a flaw in the corpus-based success metric in that it favors greedy policies because the user side behaviours are fixed rather than interactional⁶. Despite the fact that LIDMs are considered only marginally better than NDM on subjective success, the LIDMs do outperform NDM on both comprehension and naturalness scores. This is because the proposed LIDM models can better capture multiple modes in the communicative intention and thereby respond more naturally by sampling from the latent intention variable.

Three example conversations are shown between a human judge and a machine, one from LIDM in Table E.1 and two from LIDM+RL in Table E.2, respectively. The results are displayed one exchange per block. Each induced latent intention is

⁵Note that both NDM+Att+SS and LIDM use self-supervised information

⁶The system tries to provide as much information as possible in the early turns, in case the fixed user side behaviours a few turns later do not fit the scenario the system originally planned.

shown by a tuple (index, probability) followed by a decoded response, and the sample dialogues were produced by following the responses highlighted in bold. As can be seen, the LIDM shown in Table E.1 clearly has multiple modes in the distribution over the learned intention latent variable, and what it represents can be easily interpreted by the response generated. However, some intentions (such as intent 0) can result in very different responses under different dialogue states even though they were supervised by a small response set as shown in Table 7.1. This is mainly because of the variance introduced during variational inference. Finally, when comparing Table E.1 and Table E.2, we can observe the difference between the two dialogue strategies: the LIDM, by inferring its policy from the supervised dataset, reflects the diverse set of modes in the underlying distribution; whereas the LIDM+RL, which refined its strategy using RL, exhibits a much greedier behavior in achieving task success (e.g. in Table E.2 in block 2 & 4 the LIDM+RL agent provides the address and phone number even before the user asks). This is also supported by the human evaluation in Table 7.3 where LIDM+RL has much shorter dialogues on average compared to the other two models.

7.5 Summary

In this chapter, we propose a framework for learning dialogue intentions via discrete latent variable models and introduced the Latent Intention Dialogue Model (LIDM) for goal-oriented dialogue modeling. According to the experiments, the LIDM is able to discover an effective initial policy from the underlying data distribution and is capable of revising its strategy based on an external reward using reinforcement learning. We believe this is a promising step forward for building autonomous dialogue agents since the learned discrete latent variable interface enables the agent to perform learning using several differing paradigms. The experiments showed that the proposed LIDM is able to communicate with human subjects and outperforms previous published results.

Chapter 8

Conclusions and Further Work

This thesis has investigated deep generative models for NLP with neural variational inference. The experiments on several different scenarios including document/topic modelling, question answer selection, sentence compression and goal oriented dialogue modelling have demonstrated the effectiveness and efficiency of our models on learning stochastic representations, discovering topics and modelling latent intentions among human utterances. We show that deep generative models are able to learn knowledge from language and carry out a range of NLP tasks with unlabelled data or small amounts of labelled data.

In this thesis, the deep generative models are divided into two folds by the settings of the latent variables (continuous or discrete). However, the generic neural variational inference framework is suitable for both continuous and discrete latent variables, and is able to incorporate different kinds of deep neural networks. More importantly, the inference networks acting as black-box functions are able to approximate multiplicate and complex distributions that can be jointly trained with generative models, which provides efficient inference and grant the possibilities for more interesting and sophisticated generative models.

In practice, both continuous and discrete latent variables have their own advantages and disadvantages. Continuous variable is good at learning dense stochastic representations. It is easy to generalise and fits the encoder-decoder structure very well, especially in the unsupervised learning situation where the model can be built as a continuous variational auto-encoder and easily trained with the help of continuous reparameterisation trick. However, the KL divergence between the prior distribution and the inference network acts as a strong regularisation for the learning of generative model. If a powerful autoregressive neural network (e.g. LSTM) is employed as the decoder, it has a chance to ignore the latent variable and generate the observations as an unconditional model, which ends up with a trivial encoder. Hence, continuous

latent variables are preferred due to the simplicity of inference, but the decoders are suggested to be built by simple and expressive neural networks to avoid the possible failure from the perspective of optimisation.

Discrete latent variable, in contrast, is difficult to carry out inference due to the high variance issue. In addition, as it lacks of smooth prior distribution, the discrete latent variables are hard to be used directly in unsupervised learning models. Nevertheless, discrete variables are good at constructing predictive models. By using discrete latent distribution in the policy network, we are able to easily incorporate semi-supervised and reinforcement learning to further improve the performance of generative models. Moreover, in the context of language, discrete latent distribution is more natural for the reasoning about complex semantics, incorporating structure information and modelling interactive actions (e.g. dialogues). It generally has better interpretability and easier to incorporate linguistic inductive biases. Therefore, it has more potential in the field of natural language processing.

Introducing Bayesian non-parametrics in deep generative models is another interesting part of this thesis. The complicated constructive process of Dirichlet process yields to difficult inference in traditional probabilistic models. In the framework of neural variational inference, however, we employ stick-breaking constructions parameterised by continuous latent variables to model categorical distributions. In this case, the parameterisation networks can be easily updated together with the generative models by stochastic gradient back-propagation. With the help of recurrent stick-breaking construction (RSB), a truncation-free variational inference can be implemented to dynamically choose the dimension of latent space during the inference process. This unbounded property is able to further extend the scalability of deep generative models.

As of now, we have explored a few different kinds of deep generative models applied for NLP and achieved good performance compared to other benchmark systems, but there is still big space for further improvement due to the pressing requirement of exploiting unlabelled data. Looking forward, the following topics can be further studied in order to go deeper in the context of deep generative models:

1. Developing advanced inference algorithm and optimisation method for the neural variational inference framework. Variational auto-encoder (VAE) is an efficient framework of modeling natural images, but it is still not satisfiable on generating natural language sentences due to the strong reliance on powerful autoregressive neural networks. To develop more advanced inference algorithm is able to further extend the capabilities of deep generative models for NLP.

2. Extending the proposed topic models into contextual neural topic models. Traditional probabilistic topic models achieved huge success owing to their flexibility of incorporating extra document information. Therefore time, authorship, geo-location and categorical labels could all be integrated into the neural topic models to further extend the vanilla neural topic models into contextual neural topic models.
3. Discovering latent structures in language. The powerful recurrent neural networks are especially good at generating natural language sentences, generally by the default left-to-right order. However, by using discrete variables representing latent structures, we are able to incorporate grammars and logics in the natural language generation rather than simply modelling language as plain sequences. Hence, it is promising to develop more sophisticated and grammar-guided language generation models with structural information.
4. Unsupervised language learning from multimodal resources. Human beings are able to learn and develop language ability without supervision, since language is grounded in daily life with the help of multiple input resources including vision, voice, text and symbols. It is a challenging but worthwhile attempt to jointly model multiple observations in the same deep generative model. In this context, we expect the machine is able to learn knowledge from language, make sense from surroundings and evolve without supervisions.

Appendix A

Details of Inference

A.1 Neural Variational Document Model

(1) Inference Network $q_\phi(\mathbf{h}|\mathbf{X})$:

$$\boldsymbol{\lambda} = \text{ReLU}(\mathbf{W}_1\mathbf{X} + \mathbf{b}_1) \quad (\text{A.1})$$

$$\boldsymbol{\pi} = \text{ReLU}(\mathbf{W}_2\boldsymbol{\lambda} + \mathbf{b}_2) \quad (\text{A.2})$$

$$\boldsymbol{\mu} = \mathbf{W}_3\boldsymbol{\pi} + \mathbf{b}_3 \quad (\text{A.3})$$

$$\log \boldsymbol{\sigma} = \mathbf{W}_4\boldsymbol{\pi} + \mathbf{b}_4 \quad (\text{A.4})$$

$$\mathbf{h} \sim \mathcal{N}(\boldsymbol{\mu}(\mathbf{X}), \boldsymbol{\sigma}^2(\mathbf{X})) \quad (\text{A.5})$$

(2) Generative Model $p_\theta(\mathbf{X}|\mathbf{h})$:

$$\mathbf{e}_i = \exp(-\mathbf{h}^T \mathbf{R}\mathbf{x}_i + \mathbf{b}_{x_i}) \quad (\text{A.6})$$

$$p_\theta(\mathbf{x}_i|\mathbf{h}) = \frac{\mathbf{e}_i}{\sum_j |\mathbf{V}| \mathbf{e}_j} \quad (\text{A.7})$$

$$p_\theta(\mathbf{X}|\mathbf{h}) = \prod_i^N p_\theta(\mathbf{x}_i|\mathbf{h}) \quad (\text{A.8})$$

(3) KL Divergence $D_{\text{KL}}[q_\phi(\mathbf{h}|\mathbf{X})||p(\mathbf{h})]$:

$$D_{\text{KL}} = -\frac{1}{2}(K - \|\boldsymbol{\mu}\|^2 - \|\boldsymbol{\sigma}\|^2 + \log |\text{diag}(\boldsymbol{\sigma}^2)|) \quad (\text{A.9})$$

The variational lower bound to be optimised:

$$\begin{aligned} \mathcal{L} = & \mathbb{E}_{q_\phi(\mathbf{h}|\mathbf{X})} \left[\sum_{i=1}^N \log p_\theta(\mathbf{x}_i|\mathbf{h}) \right] \\ & - D_{\text{KL}}[q_\phi(\mathbf{h}|\mathbf{X})||p(\mathbf{h})] \end{aligned} \quad (\text{A.10})$$

$$\begin{aligned} \approx & \sum_{l=1}^L \sum_{i=1}^N \log p_\theta(\mathbf{x}_i|\mathbf{h}^{(l)}) \\ & + \frac{1}{2}(K - \|\boldsymbol{\mu}\|^2 - \|\boldsymbol{\sigma}\|^2 + \log |\text{diag}(\boldsymbol{\sigma}^2)|) \end{aligned} \quad (\text{A.11})$$

A.2 Neural Answer Selection Model

(1) Inference Network $q_\phi(\mathbf{h}|\mathbf{q}, \mathbf{a}, \mathbf{y})$:

$$\mathbf{s}_q(|\mathbf{q}|) = f_q^{\text{LSTM}}(\mathbf{q}) \quad (\text{A.12})$$

$$\mathbf{s}_a(|\mathbf{a}|) = f_a^{\text{LSTM}}(\mathbf{a}) \quad (\text{A.13})$$

$$\mathbf{s}_y = \mathbf{W}_5 \mathbf{y} + \mathbf{b}_5 \quad (\text{A.14})$$

$$\boldsymbol{\gamma} = \mathbf{s}_q(|\mathbf{q}|) \parallel \mathbf{s}_a(|\mathbf{a}|) \parallel \mathbf{s}_y \quad (\text{A.15})$$

$$\boldsymbol{\lambda}_\phi = \tanh(\mathbf{W}_6 \boldsymbol{\gamma} + \mathbf{b}_6) \quad (\text{A.16})$$

$$\boldsymbol{\pi}_\phi = \tanh(\mathbf{W}_7 \boldsymbol{\lambda}_\phi + \mathbf{b}_7) \quad (\text{A.17})$$

$$\boldsymbol{\mu}_\phi = \mathbf{W}_8 \boldsymbol{\pi}_\phi + \mathbf{b}_8 \quad (\text{A.18})$$

$$\log \boldsymbol{\sigma}_\phi = \mathbf{W}_9 \boldsymbol{\pi}_\phi + \mathbf{b}_9 \quad (\text{A.19})$$

$$\mathbf{h} \sim \mathcal{N}(\boldsymbol{\mu}_\phi(\mathbf{q}, \mathbf{a}, \mathbf{y}), \boldsymbol{\sigma}_\phi^2(\mathbf{q}, \mathbf{a}, \mathbf{y})) \quad (\text{A.20})$$

(2) Generative Model

$p_\theta(\mathbf{h}|\mathbf{q})$:

$$\boldsymbol{\lambda}_\theta = \tanh(\mathbf{W}_1 \mathbf{s}_q(|\mathbf{q}|) + \mathbf{b}_1) \quad (\text{A.21})$$

$$\boldsymbol{\pi}_\theta = \tanh(\mathbf{W}_2 \boldsymbol{\lambda}_\theta + \mathbf{b}_2) \quad (\text{A.22})$$

$$\boldsymbol{\mu}_\theta = \mathbf{W}_3 \boldsymbol{\pi}_\theta + \mathbf{b}_3 \quad (\text{A.23})$$

$$\log \boldsymbol{\sigma}_\theta = \mathbf{W}_4 \boldsymbol{\pi}_\theta + \mathbf{b}_4 \quad (\text{A.24})$$

$p_\theta(\mathbf{y}|\mathbf{q}, \mathbf{a}, \mathbf{h})$:

$$\mathbf{e}(i) = \mathbf{W}_\alpha^T \tanh(\mathbf{W}_h \mathbf{h} + \mathbf{W}_s \mathbf{s}_a(i)) \quad (\text{A.25})$$

$$\alpha(i) = \frac{e(i)}{\sum_j e(j)} \quad (\text{A.26})$$

$$\mathbf{c}(\mathbf{a}, \mathbf{h}) = \sum_i \mathbf{s}_a(i) \alpha(i) \quad (\text{A.27})$$

$$\mathbf{z}_a(\mathbf{a}, \mathbf{h}) = \tanh(\mathbf{W}_a \mathbf{c}(\mathbf{a}, \mathbf{h}) + \mathbf{W}_n \mathbf{s}_a(|\mathbf{a}|)) \quad (\text{A.28})$$

$$\mathbf{z}_q(\mathbf{q}) = \mathbf{s}_q(|\mathbf{q}|) \quad (\text{A.29})$$

$$p_\theta(\mathbf{y} = 1|\mathbf{q}, \mathbf{a}, \mathbf{h}) = \sigma(\mathbf{z}_q^T \mathbf{M} \mathbf{z}_a + b) \quad (\text{A.30})$$

(3) KL Divergence $D_{\text{KL}}[q_\phi(\mathbf{h}|\mathbf{q}, \mathbf{a}, \mathbf{y})||p_\theta(\mathbf{h}|\mathbf{q})]$:

$$\begin{aligned} D_{\text{KL}} = & -\frac{1}{2}(K + \log |\text{diag}(\boldsymbol{\sigma}_\phi^2)| - \log |\text{diag}(\boldsymbol{\sigma}_\theta^2)| - \text{Tr}(\text{diag}(\boldsymbol{\sigma}_\phi^2) \text{diag}^{-1}(\boldsymbol{\sigma}_\theta^2)) \\ & - (\boldsymbol{\mu}_\phi - \boldsymbol{\mu}_\theta)^T \text{diag}^{-1}(\boldsymbol{\sigma}_\theta^2)(\boldsymbol{\mu}_\phi - \boldsymbol{\mu}_\theta)) \end{aligned} \quad (\text{A.31})$$

The variational lower bound to be optimised:

$$\begin{aligned}
\mathcal{L} &= \mathbb{E}_{q_\phi(\mathbf{h}|\mathbf{q},\mathbf{a},\mathbf{y})}[\log p_\theta(\mathbf{y}|\mathbf{q},\mathbf{a},\mathbf{h})] - D_{\text{KL}}[q_\phi(\mathbf{h}|\mathbf{q},\mathbf{a},\mathbf{y})||p_\theta(\mathbf{h}|\mathbf{q})] & (\text{A.32}) \\
&\approx \sum_{l=1}^L \mathbf{y} \log \sigma(\mathbf{z}_q^T \mathbf{M} \mathbf{z}_a^{(l)} + b) + (1 - \mathbf{y}) \log(1 - \sigma(\mathbf{z}_q^T \mathbf{M} \mathbf{z}_a^{(l)} + b)) \\
&\quad + \frac{1}{2}(K + \log |\text{diag}(\boldsymbol{\sigma}_\phi^2)| - \log |\text{diag}(\boldsymbol{\sigma}_\theta^2)| - \text{Tr}(\text{diag}(\boldsymbol{\sigma}_\phi^2) \text{diag}^{-1}(\boldsymbol{\sigma}_\theta^2))) \\
&\quad - (\boldsymbol{\mu}_\phi - \boldsymbol{\mu}_\theta)^T \text{diag}^{-1}(\boldsymbol{\sigma}_\theta^2) (\boldsymbol{\mu}_\phi - \boldsymbol{\mu}_\theta) & (\text{A.33})
\end{aligned}$$

Appendix B

Discovered Topics

<i>Space</i>	<i>Religion</i>	<i>Encryption</i>	<i>Sport</i>	<i>Science</i>
space	god	encryption	player	science
satellite	atheism	device	hall	theory
april	exist	technology	defensive	scientific
sequence	atheist	protect	team	universe
launch	moral	americans	average	experiment
president	existence	chip	career	observation
station	marriage	use	league	evidence
radar	system	privacy	play	exist
training	parent	industry	bob	god
committee	murder	enforcement	year	mistake

Table B.1: Topics learned by GSM on 20NewsGroups dataset.

<i>Space</i>	<i>Religion</i>	<i>Lawsuit</i>	<i>Vehicle</i>	<i>Science</i>
moon	atheist	homicide	bike	theory
lunar	life	gun	motorcycle	science
orbit	eternal	rate	dod	gary
spacecraft	christianity	handgun	insurance	scientific
billion	hell	crime	bmw	sun
launch	god	firearm	ride	orbit
space	christian	weapon	dealer	energy
hockey	atheism	knife	oo	experiment
cost	religion	study	car	mechanism
nasa	brian	death	buy	star

Table B.2: Topics learned by GSB on 20NewsGroups dataset.

<i>Aerospace</i>	<i>Crime</i>	<i>Hardware</i>	<i>Technology</i>	<i>Science</i>
instruction	gun	drive	technology	science
spacecraft	weapon	scsi	americans	hell
amp	crime	ide	pit	scientific
pat	firearm	scsus	encryption	evidence
wing	criminal	hd	policy	physical
plane	use	go	industry	eternal
algorithm	control	controller	protect	universe
db	handgun	tape	privacy	experiment
reduce	law	datum	product	reason
orbit	kill	isa	approach	death

Table B.3: Topics learned by RSB on 20NewsGroups dataset.

Appendix C

Visualisation

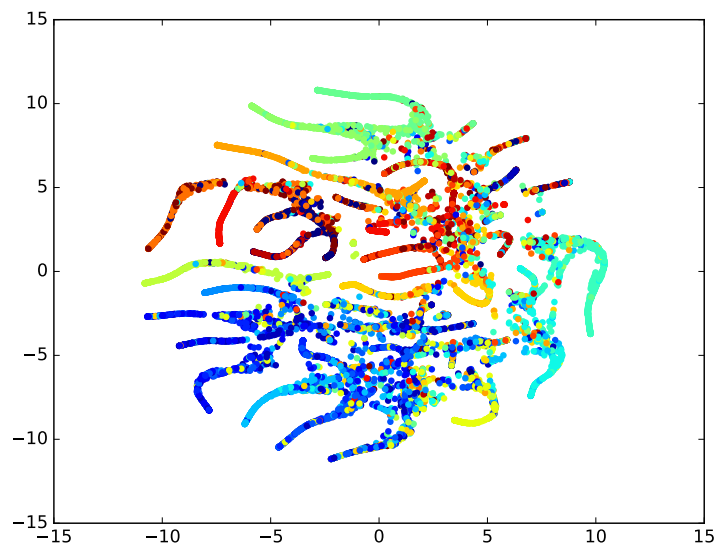


Figure C.1: t-SNE projection of the estimated topic proportions of each document (i.e. $q(\theta|d)$) from *20NewsGroups*. The vectors are learned by the GSM model with 50 topics and each color represents one group from the 20 different groups of the dataset.

Appendix D

Sentence Compression Examples

src	the sri lankan government on wednesday announced the closure of government schools with immediate effect as a military campaign against tamil separatists escalated in the north of the country .
ref	sri lanka closes schools as war escalates
asc_a	sri lanka closes government schools
asc_e	sri lankan government closure schools escalated
fsc_a	sri lankan government closure with tamil rebels closure
src	factory orders for manufactured goods rose #.# percent in september , the commerce department said here thursday .
ref	us september factory orders up #.# percent
asc_a	us factory orders up #.# percent in september
asc_e	factory orders rose #.# percent in september
fsc_a	factory orders #.# percent in september
src	hong kong signed a breakthrough air services agreement with the united states on friday that will allow us airlines to carry freight to asian destinations via the territory .
ref	hong kong us sign breakthrough aviation pact
asc_a	us hong kong sign air services agreement
asc_e	hong kong signed air services agreement with united states
fsc_a	hong kong signed air services pact with united states
src	a swedish un soldier in bosnia was shot and killed by a stray bullet on tuesday in an incident authorities are calling an accident , military officials in stockholm said tuesday .
ref	swedish un soldier in bosnia killed by stray bullet
asc_a	swedish un soldier killed in bosnia
asc_e	swedish un soldier shot and killed
fsc_a	swedish soldier shot and killed in bosnia

Table D.1: Examples of the compression sentences (Part I).

src	tea scores on the fourth day of the second test between australia and pakistan here monday .
ref	australia vs pakistan tea scorecard
asc_a	australia v pakistan tea scores
asc_e	australia tea scores
fsc_a	tea scores on #th day of #nd test
src	india won the toss and chose to bat on the opening day in the opening test against west indies at the antigua recreation ground on friday .
ref	india win toss and elect to bat in first test
asc_a	india win toss and bat against west indies
asc_e	india won toss on opening day against west indies
fsc_a	india chose to bat on opening day against west indies
src	a powerful bomb exploded outside a navy base near the sri lankan capital colombo tuesday , seriously wounding at least one person , military officials said .
ref	bomb attack outside srilanka navy base
asc_a	bomb explodes outside sri lanka navy base
asc_e	bomb outside sri lankan navy base wounding one
fsc_a	bomb exploded outside sri lankan navy base
src	press freedom in algeria remains at risk despite the release on wednesday of prominent newspaper editor mohamed UNK after a two-year prison sentence , human rights organizations said .
ref	algerian press freedom at risk despite editor 's release UNK picture
asc_a	algeria press freedom remains at risk
asc_e	algeria press freedom remains at risk
fsc_a	press freedom in algeria at risk

Table D.2: Examples of the compression sentences (Part II).

Appendix E

Goal Oriented Dialogue Examples

Human : Hi, I'm hungry for some Indonesian

Machine: (0 0.71) **i am sorry , there are no indonesian restaurants in the area .
would you like something else ?**

(46 0.19) there are no restaurants serving indonesian kind of food .

(34 0.07) what location would you like ?

Human : Oh no. How about Indian?

Machine: (0 0.59) there are 9 restaurants in the east side of town . would you like their
address ?

(30 0.15) curry prince serves indian cuisine .

(34 0.10) **what area would you like ?**

Human : East side please

Machine: (0 0.62) **curry prince is a indian restaurant in the east side of town .
would you like their telephone number ?**

(84 0.19) curry prince is an indian restaurant in the east of town and the number
is 01223 566388 .

(81 0.08) curry prince serves indian type of cuisine and is located in the east part
of town .

Human : yes , the address and phone number please

Machine: (3 0.46) the address is 451 newmarket road fen ditton and the phone is
01223 566388 .

(11 0.16) **the address for curry prince is 451 newmarket road fen ditton
and the phone number is 01223 566388 .**

(0 0.15) curry prince is located at 451 newmarket road fen ditton and their
phone is 01223 566388 . is there anything else i can help you with ?

Human : Okay, thank you goodbye

Machine: (0 0.57) thank you , goodbye .

(2 0.13) goodbye .

(1 0.11) **you are welcome . goodbye .**

(6 0.10) thank you for using the cambridge restaurant system . goodbye .

Table E.1: A sample dialogue from the LIDM, I=100 model, one exchange per block. Each latent intention is shown by a tuple (index, probability), followed by a decoded response. The sample dialogue was produced by following the responses highlighted in bold.

Human : Hi, I'm hungry for some Indonesian .
Machine: (0 0.61) **i am sorry , there are no indonesian restaurants in the area . would you like something else ?**
(46 0.17) no restaurants in the part of town .
(34 0.14) what part of town would you like ?

Human : Oh no. How about Indian?
Machine: (93 0.51) **the gandhi is located at 72 regent street city centre and their telephone number is 01223 353942 .**
(72 0.21) the gandhi serves indian kind of food and is in the centre of town and is in the cheap price range .
(12 0.12) the gandhi is a indian restaurant in the centre location and is in the cheap price range .

Human : Okay, thank you goodbye
Machine: (6 0.28) **thank you for using the cambridge restaurant system . goodbye .**
(1 0.24) you are welcome . goodbye .
(85 0.19) is there anything else i can help you with ?

Human : hey can i have some chinese cuisine in the centre ? and a cheap one please
Machine: (15 0.42) golden house is located at 12 lensfield road city centre and their number is 01842 753771 .
(5 0.38) **golden house is located at 12 lensfield road city centre and the number is 01842 753771 . is there anything else i can help you with ?**
(93 0.10) golden house is located at 12 lensfield road city centre and their phone is 01842 753771 .

Human : that is all
Machine: (6 0.57) **thank you for using the cambridge restaurant system . goodbye .**
(1 0.20) you are welcome . goodbye .
(17 0.11) have a nice day !

Table E.2: Two sample dialogues from the LIDM+RL, I=100 model, one exchange per block. Comparing to Table E.1, the RL agent demonstrates a much greedier behavior toward task success. This can be seen in block 2 & block 4 in which the agent provides the address and phone number even before the user asks.

Bibliography

- [1] Christophe Andrieu, Nando De Freitas, Arnaud Doucet, and Michael I Jordan. An introduction to mcmc for machine learning. *Machine learning*, 50(1-2):5–43, 2003.
- [2] Hagai Attias. A variational bayesian framework for graphical models. In *Proceedings of NIPS*, 2000.
- [3] Michael Auli and Jianfeng Gao. Decoder integration and expected bleu training for recurrent neural network language models. In *ACL*. Association for Computational Linguistics, 2014.
- [4] Jimmy Ba, Roger Grosse, Ruslan Salakhutdinov, and Brendan Frey. Learning wake-sleep recurrent attention models. In *Proceedings of NIPS*, 2015.
- [5] Jimmy Ba, Volodymyr Mnih, and Koray Kavukcuoglu. Multiple object recognition with visual attention. In *Proceedings of ICLR*, 2015.
- [6] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *Proceedings of ICLR*, 2015.
- [7] Matthew James Beal. *Variational algorithms for approximate Bayesian inference*. PhD thesis, University of London, 2003.
- [8] Thierry Bertin-Mahieux, Daniel P.W. Ellis, Brian Whitman, and Paul Lamere. The million song dataset. In *Proceedings of ISMIR*, 2011.
- [9] David M Blei and John D Lafferty. Dynamic topic models. In *Proceedings of ICML*, pages 113–120. ACM, 2006.
- [10] David M Blei and John D Lafferty. A correlated topic model of science. *The Annals of Applied Statistics*, 2007.

- [11] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *The Journal of Machine Learning Research*, 3:993–1022, 2003.
- [12] Antoine Bordes and Jason Weston. Learning end-to-end goal-oriented dialog. In *Proceedings of ICLR*, 2017.
- [13] Jörg Bornschein and Yoshua Bengio. Reweighted wake-sleep. In *Proceedings of ICLR*, 2015.
- [14] Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew M Dai, Rafal Jozefowicz, and Samy Bengio. Generating sentences from a continuous space. *arXiv preprint arXiv:1511.06349*, 2015.
- [15] Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. Importance weighted autoencoders. In *Proceedings of ICLR*, 2016.
- [16] Kris Cao and Stephen Clark. Latent variable dialogue models and their diversity. In *Proceedings of EACL*, 2017.
- [17] Bradley P Carlin and Nicholas G Polson. Inference for nonconjugate bayesian models using the gibbs sampler. *Canadian Journal of statistics*, 19(4):399–405, 1991.
- [18] Jan K Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, Kyunghyun Cho, and Yoshua Bengio. Attention-based models for speech recognition. In *Proceedings of NIPS*, pages 577–585, 2015.
- [19] Andrew M Dai and Quoc V Le. Semi-supervised sequence learning. In *Proceedings of NIPS*, pages 3061–3069, 2015.
- [20] Rajarshi Das, Manzil Zaheer, and Chris Dyer. Gaussian lda for topic models with word embeddings. In *Proceedings of ACL*, volume 1, pages 795–804, 2015.
- [21] Peter Dayan and Geoffrey E Hinton. Varieties of helmholtz machine. *Neural Networks*, 9(8):1385–1403, 1996.
- [22] Chris Dyer, Adhiguna Kuncoro, Miguel Ballesteros, and Noah A Smith. Recurrent neural network grammars. In *Proceedings of NAACL*, 2016.
- [23] SM Eslami, Nicolas Heess, Theophane Weber, Yuval Tassa, Koray Kavukcuoglu, and Geoffrey E Hinton. Attend, infer, repeat: Fast scene understanding with generative models. *arXiv preprint arXiv:1603.08575*, 2016.

- [24] Katja Filippova, Enrique Alfonseca, Carlos A Colmenares, Lukasz Kaiser, and Oriol Vinyals. Sentence compression by deletion with lstms. In *Proceedings of EMNLP*, 2015.
- [25] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of ICML*, pages 1050–1059, 2016.
- [26] Milica Gašić, Catherine Breslin, Matthew Henderson, Dongho Kim, Martin Szummer, Blaise Thomson, Pirros Tsiakoulis, and Steve Young. On-line policy optimisation of bayesian spoken dialogue systems via human interaction. In *Proceedings of ICASSP*, 2013.
- [27] Zoubin Ghahramani. Nonparametric bayesian methods. In *Tutorial presentation at the UAI Conference*, 2005.
- [28] Karol Gregor, Ivo Danihelka, Alex Graves, and Daan Wierstra. Draw: A recurrent neural network for image generation. In *Proceedings of ICML*, 2015.
- [29] Karol Gregor, Andriy Mnih, and Daan Wierstra. Deep autoregressive networks. In *Proceedings of ICML*, 2014.
- [30] Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. Incorporating copying mechanism in sequence-to-sequence learning. *arXiv preprint arXiv:1603.06393*, 2016.
- [31] Shixiang Gu, Sergey Levine, Ilya Sutskever, and Andriy Mnih. Muprop: Unbiased backpropagation for stochastic neural networks. In *Proceedings of ICLR*, 2016.
- [32] Caglar Gulcehre, Sungjin Ahn, Ramesh Nallapati, Bowen Zhou, and Yoshua Bengio. Pointing the unknown words. *arXiv preprint arXiv:1603.08148*, 2016.
- [33] Michael Heilman and Noah A. Smith. Tree edit models for recognizing textual entailments, paraphrases, and answers to questions. In *Proceedings of NAACL*, 2010.
- [34] Matthew Henderson. Machine learning for dialog state tracking: A review. In *Machine Learning in Spoken Language Processing Workshop*, 2015.

- [35] Matthew Henderson, Blaise Thomson, and Steve Young. Word-based dialog state tracking with recurrent neural networks. In *Proceedings of SigDial*, 2014.
- [36] Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching machines to read and comprehend. In *Proceedings of NIPS*, 2015.
- [37] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *Proceedings of ICLR*, 2017.
- [38] Felix Hill, Antoine Bordes, Sumit Chopra, and Jason Weston. The goldilocks principle: Reading children’s books with explicit memory representations. In *Proceedings of ICLR*, 2016.
- [39] Geoffrey E Hinton, Peter Dayan, Brendan J Frey, and Radford M Neal. The wake-sleep algorithm for unsupervised neural networks. *Science*, 268(5214):1158–1161, 1995.
- [40] Geoffrey E Hinton and Ruslan Salakhutdinov. Replicated softmax: an undirected topic model. In *Proceedings of NIPS*, 2009.
- [41] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.
- [42] Geoffrey E Hinton and Richard S Zemel. Autoencoders, minimum description length, and helmholtz free energy. In *Proceedings of NIPS*, 1994.
- [43] Sepp Hochreiter and Jurgen Schmidhuber. Long short-term memory. *Neural Computation*, 1997.
- [44] Matthew Hoffman, Francis R Bach, and David M Blei. Online learning for latent dirichlet allocation. In *Proceedings of NIPS*, pages 856–864, 2010.
- [45] Thomas Hofmann. Probabilistic latent semantic indexing. In *Proceedings of SIGIR*, 1999.
- [46] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

- [47] Michael I Jordan, Zoubin Ghahramani, Tommi S Jaakkola, and Lawrence K Saul. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233, 1999.
- [48] Nal Kalchbrenner and Phil Blunsom. Recurrent continuous translation models. In *Proceedings of EMNLP*, 2013.
- [49] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of CVPR*, 2015.
- [50] John F. Kelley. An iterative design methodology for user-friendly natural language office information applications. *ACM Transaction on Information Systems*, 1984.
- [51] Mohammad Emtiyaz Khan, Shakir Mohamed, Benjamin M Marlin, and Kevin P Murphy. A stick-breaking likelihood for categorical data analysis with latent gaussian models. In *Proceedings of AISTATS*, 2012.
- [52] Chloé Kiddon, Luke Zettlemoyer, and Yejin Choi. Globally coherent text generation with neural checklist models. In *Proceedings of EMNLP*, 2016.
- [53] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proceedings of ICLR*, 2015.
- [54] Diederik P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. In *Proceedings of NIPS*, 2014.
- [55] Diederik P Kingma, Tim Salimans, and Max Welling. Variational dropout and the local reparameterization trick. In *Proceedings of NIPS*, pages 2575–2583, 2015.
- [56] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *Proceedings of ICLR*, 2014.
- [57] Vijay R. Konda and John N. Tsitsiklis. On actor-critic algorithms. *SIAM J. Control Optim.*, 42(4):1143–1166, April 2003.
- [58] Tomáš Kočiský, Gábor Melis, Edward Grefenstette, Chris Dyer, Wang Ling, Phil Blunsom, and Karl Moritz Hermann. Semantic parsing with semi-supervised sequential autoencoders. In *Proceedings of EMNLP*, 2016.

- [59] Ankit Kumar, Ozan Irsoy, Jonathan Su, James Bradbury, Robert English, Brian Pierce, Peter Ondruska, Ishaan Gulrajani, and Richard Socher. Ask me anything: Dynamic memory networks for natural language processing. *arXiv preprint arXiv:1506.07285*, 2015.
- [60] Thomas K Landauer, Peter W Foltz, and Darrell Laham. An introduction to latent semantic analysis. *Discourse processes*, 25(2-3):259–284, 1998.
- [61] Hugo Larochelle and Stanislas Lauly. A neural autoregressive topic model. In *Proceedings of NIPS*, 2012.
- [62] Jey Han Lau, David Newman, and Timothy Baldwin. Machine reading tea leaves: Automatically evaluating topic coherence and topic model quality. In *Proceedings of EACL*, 2014.
- [63] Quoc V Le and Tomas Mikolov. Distributed representations of sentences and documents. In *Proceedings of ICML*, 2014.
- [64] Quoc V. Le and Tomas Mikolov. Distributed representations of sentences and documents. In *Proceedings of ICML*, 2014.
- [65] Yann LeCun, Sumit Chopra, and Raia Hadsell. A tutorial on energy-based learning. *Predicting structured data*, 2006.
- [66] Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. A diversity-promoting objective function for neural conversation models. In *Proceedings of NAACL-HLT*, 2016.
- [67] Jiwei Li, Michel Galley, Chris Brockett, Georgios Spithourakis, Jianfeng Gao, and Bill Dolan. A persona-based neural conversation model. In *Proceedings of ACL*, 2016.
- [68] Jiwei Li, Alexander H. Miller, Sumit Chopra, Marc’Aurelio Ranzato, and Jason Weston. Dialogue learning with human-in-the-loop. In *Proceedings of ICLR*, 2017.
- [69] Jiwei Li, Will Monroe, Alan Ritter, Dan Jurafsky, Michel Galley, and Jianfeng Gao. Deep reinforcement learning for dialogue generation. In *Proceedings of EMNLP*, 2016.

- [70] Wang Ling, Edward Grefenstette, Karl Moritz Hermann, Tomas Kocisky, Andrew Senior, Fumin Wang, and Phil Blunsom. Latent predictor networks for code generation. In *Proceedings of ACL*, 2016.
- [71] Chris J Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. *arXiv preprint arXiv:1611.00712*, 2016.
- [72] Diego Marcheggiani and Ivan Titov. Discrete-state variational autoencoders for joint discovery and factorization of relations. *Transactions of the Association for Computational Linguistics*, 4, 2016.
- [73] Jon D McAuliffe and David M Blei. Supervised topic models. In *Proceedings of NIPS*, pages 121–128, 2008.
- [74] Yishu Miao and Phil Blunsom. Language as a latent variable: Discrete generative models for sentence compression. In *Proceedings of EMNLP*, 2016.
- [75] Yishu Miao, Lei Yu, and Phil Blunsom. Neural variational inference for text processing. In *Proceedings of ICML*, 2016.
- [76] Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In *Proceedings of NIPS*, 2013.
- [77] Andriy Mnih and Karol Gregor. Neural variational inference and learning in belief networks. In *Proceedings of ICML*, 2014.
- [78] Andriy Mnih and Danilo Rezende. Variational inference for monte carlo objectives. In *Proceedings of ICML*, 2016.
- [79] Volodymyr Mnih, Nicolas Heess, and Alex Graves. Recurrent models of visual attention. In *Proceedings of NIPS*, 2014.
- [80] Ramesh Nallapati, Bowen Zhou, Çağlar Gulçehre, and Bing Xiang. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023*, 2016.
- [81] Radford M Neal. Probabilistic inference using markov chain monte carlo methods. *Technical report: CRG-TR-93-1*, 1993.

- [82] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of ACL*.
- [83] Marc’Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. Sequence level training with recurrent neural networks. In *Proceedings of ICLR*, 2016.
- [84] Danilo J Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of ICML*, 2014.
- [85] Michal Rosen-Zvi, Thomas Griffiths, Mark Steyvers, and Padhraic Smyth. The author-topic model for authors and documents. In *Proceedings of UAI*, 2004.
- [86] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. Technical report, DTIC Document, 1985.
- [87] Alexander M Rush, Sumit Chopra, and Jason Weston. A neural attention model for abstractive sentence summarization. In *Proceedings of EMNLP*, 2015.
- [88] Iulian V. Serban, Alessandro Sordoni, Ryan Lowe, Laurent Charlin, Joelle Pineau, Aaron Courville, and Yoshua Bengio. A hierarchical latent variable encoder-decoder model for generating dialogues. *arXiv preprint: 1605.06069*, 2016.
- [89] Iulian Vlad Serban, Alessandro Sordoni, Yoshua Bengio, Aaron C. Courville, and Joelle Pineau. Hierarchical neural network generative models for movie dialogues. In *Proceedings of AAAI*, 2015.
- [90] Jayaram Sethuraman. A constructive definition of dirichlet priors. *Statistica sinica*, pages 639–650, 1994.
- [91] Aliaksei Severyn. *Modelling input texts: from Tree Kernels to Deep Learning*. PhD thesis, University of Trento, 2015.
- [92] Aliaksei Severyn and Alessandro Moschitti. Automatic feature engineering for answer selection and extraction. In *Proceedings of EMNLP*, 2013.

- [93] Lifeng Shang, Zhengdong Lu, and Hang Li. Neural responding machine for short-text conversation. In *Proceedings of ACL*, 2015.
- [94] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.
- [95] Rion Snow, Daniel Jurafsky, and Andrew Y. Ng. Learning syntactic patterns for automatic hypernym discovery. In *Proceedings of NIPS*, 2004.
- [96] Akash Srivastava and Charles Sutton. Neural variational inference for topic models. *Bayesian deep learning workshop, NIPS*, 2016.
- [97] Nitish Srivastava, RR Salakhutdinov, and Geoffrey Hinton. Modeling documents with deep boltzmann machines. In *Proceedings of UAI*, 2013.
- [98] Pei-Hao Su, Milica Gasic, Nikola Mrkšić, Lina M. Rojas Barahona, Stefan Ultes, David Vandyke, Tsung-Hsien Wen, and Steve Young. On-line active reward learning for policy optimisation in spoken dialogue systems. In *Proceedings of ACL*, 2016.
- [99] Pei-Hao Su, David Vandyke, Milica Gasic, Dongho Kim, Nikola Mrksic, Tsung-Hsien Wen, and Steve Young. Learning from real users: Rating dialogue success with neural networks for reinforcement learning in spoken dialogue systems. In *Proceedings of Interspeech*, 2015.
- [100] Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. End-to-end memory networks. In *Proceedings of NIPS*, 2015.
- [101] Ilya Sutskever, Oriol Vinyals, and Quoc VV Le. Sequence to sequence learning with neural networks. In *Proceedings of NIPS*, 2014.
- [102] Yee Whye Teh, Michael I Jordan, Matthew J Beal, and David M Blei. Hierarchical dirichlet processes. *Journal of the American Statistical Association*, 101(476), 2006.
- [103] Yee Whye Teh, Michael I Jordan, Matthew J Beal, and David M Blei. Hierarchical dirichlet processes. *Journal of the american statistical association*, 2012.

- [104] David R. Traum. *Foundations of Rational Agency*, chapter Speech Acts for Dialogue Agents. Springer, 1999.
- [105] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks. In *Proceedings of NIPS*, pages 2674–2682, 2015.
- [106] Oriol Vinyals and Quoc V. Le. A neural conversational model. In *ICML Deep Learning Workshop*, 2015.
- [107] Chong Wang and David M Blei. Variational inference in nonconjugate models. *Journal of Machine Learning Research*, 14(Apr):1005–1031, 2013.
- [108] Chong Wang, John William Paisley, and David M Blei. Online variational inference for the hierarchical dirichlet process. In *Proceedings of AISTATS*, 2011.
- [109] Mengqiu Wang and Christopher D Manning. Probabilistic tree-edit models with structured latent variables for textual entailment and question answering. In *Proceedings of COLING*, 2010.
- [110] Mengqiu Wang, Noah A Smith, and Teruko Mitamura. What is the jeopardy model? a quasi-synchronous grammar for qa. In *Proceedings of EMNLP-CoNLL*, 2007.
- [111] Xuerui Wang and Andrew McCallum. Topics over time: a non-markov continuous-time model of topical trends. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2006.
- [112] Zhiguo Wang and Abraham Ittycheriah. Faq-based question answering via word alignment. *arXiv preprint arXiv:1507.02628*, 2015.
- [113] Tsung-Hsien Wen, Milica Gasic, Nikola Mrkšić, Lina M. Rojas Barahona, Pei-Hao Su, Stefan Ultes, David Vandyke, and Steve Young. Conditional generation and snapshot learning in neural dialogue systems. In *Proceedings of EMNLP*, November 2016.
- [114] Tsung-Hsien Wen, Milica Gasic, Nikola Mrkšić, Pei-Hao Su, David Vandyke, and Steve Young. Semantically conditioned lstm-based natural language generation for spoken dialogue systems. In *Proceedings of EMNLP*, September 2015.

- [115] Tsung-Hsien Wen, Yishu Miao, Phil Blunsom, and Steve Young. Latent intention dialogue models. In *Proceedings of ICML*, 2017.
- [116] Tsung-Hsien Wen, David Vandyke, Nikola Mrkšić, Milica Gašić, Lina M. Rojas-Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young. A network-based end-to-end trainable task-oriented dialogue system. In *Proceedings of EACL*, 2017.
- [117] Jason Weston, Sumit Chopra, and Antoine Bordes. Memory networks. In *Proceedings of ICLR*, 2015.
- [118] Jason E Weston. Dialog-based language learning. In *Proceedings of NIPS*, pages 829–837. Curran Associates, Inc., 2016.
- [119] Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3-4):229–256, May 1992.
- [120] Pengtao Xie, Yuntian Deng, and Eric Xing. Diversifying restricted boltzmann machine for document modeling. In *Proceedings of KDD*, pages 1315–1324. ACM, 2015.
- [121] Kelvin Xu, Jimmy Ba, Ryan Kiros, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *Proceedings of ICML*, 2015.
- [122] Yi Yang, Wen-tau Yih, and Christopher Meek. Wikiqa: A challenge dataset for open-domain question answering. In *Proceedings of EMNLP*, 2015.
- [123] Wen-tau Yih, Ming-Wei Chang, Christopher Meek, and Andrzej Pastusiak. Question answering using enhanced lexical semantic models. In *Proceedings of ACL*, 2013.
- [124] Steve Young, Milica Gašić, Blaise Thomson, and Jason D. Williams. Pomdp-based statistical spoken dialog systems: A review. In *Proceedings of the IEEE*, 2013.
- [125] Lei Yu, Karl Moritz Hermann, Phil Blunsom, and Stephen Pulman. Deep Learning for Answer Sentence Selection. In *NIPS Deep Learning Workshop*, 2014.