Probabilistic Modelling of Morphologically Rich Languages

Jan Abraham Botha

Lincoln College

University of Oxford

A thesis submitted for the degree of

Doctor of Philosophy

Hilary Term 2014

Acknowledgements

Thanks go to my main supervisor, Phil Blunsom, and to Stephen Pulman, who remained an available secondary supervisor. To Phil, I am especially grateful for two things he provided: 1) Continued patience in laying out subtle intricacies of machine learning, mixed with nudges towards interesting applications in language processing; 2) Generous exposure to his own collaborators during the early days when I was new to the NLP field. In this regard, I am also thankful to the input Chris Dyer provided at an early stage and as co-author, which helped transform my thitherto generic explorations into something more concrete.

This project was funded in part by the Rhodes Trust, and I benefited from travel grants from Lincoln College, which aided conference attendance. I would like to acknowledge three administrators in particular who made the papermaze of Oxford's bureaucracy much more tolerable through the dedicated and pleasant way in which they supported my efforts: Julie Sheppard, Carmella Elan-Gaston and Mary Eaton.

Thanks to my friends and various housemates in Oxford who made the time there so enriching and often unbelievable. In particular, thanks to Bronwyn and Alysia for frequently providing a place to stay in Oxford during the last six months after I moved away.

I give special thanks to my family for their continued support of my endeavours, and for nurturing such a special appreciation of language and languages at home. Thanks to Saskia for her love and support throughout, across continents and Channels, and not across them.

Abstract

This thesis investigates how the sub-structure of words can be accounted for in probabilistic models of language. Such models play an important role in natural language processing tasks such as translation or speech recognition, but often rely on the simplistic assumption that words are opaque symbols. This assumption does not fit morphologically complex language well, where words can have rich internal structure and sub-word elements are shared across distinct word forms.

Our approach is to encode basic notions of morphology into the assumptions of three different types of language models, with the intention that leveraging shared sub-word structure can improve model performance and help overcome data sparsity that arises from morphological processes.

In the context of n-gram language modelling, we formulate a new Bayesian model that relies on the decomposition of compound words to attain better smoothing, and we develop a new distributed language model that learns vector representations of morphemes and leverages them to link together morphologically related words. In both cases, we show that accounting for word sub-structure improves the models' intrinsic performance and provides benefits when applied to other tasks, including machine translation.

We then shift the focus beyond the modelling of word sequences and consider models that automatically learn *what* the sub-word elements of a given language are, given an unannotated list of words. We formulate a novel model that can learn discontiguous morphemes in addition to the more conventional contiguous morphemes that most previous models are limited to. This approach is demonstrated on Semitic languages, and we find that modelling discontiguous sub-word structures leads to improvements in the task of segmenting words into their contiguous morphemes.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

People can understand sentences and words they have never heard before. We do this by combining the interpretations of parts of the sentence, such as individual words or phrases, or parts of words, into an interpretation of the whole. Two key parts of this compositional view of language is that the smaller meaningful entities are easily identifiable and that their meanings are likely already known. To illustrate this, consider how immediately the following sentence is understandable to an English speaker:

> *The king finally abdicated after years of unkingly conduct.*

This sentence is understandable despite the unlikeliness that one will have previously encountered the word *unkingly*. So not only is the sentence interpretable as a function of the words it contains, but the word *unkingly* is itself interpretable as a function of the morphemes it contains—one can compose the meaning of *king* with the *-ly* suffix to derive the adjective, while the prefix *un-* supplies the negation.

In the field of natural language processing (NLP), a general approach is also to decompose problems as the manipulation of elements smaller than whole sentences. We thus face the task of determining what the elementary units of language are that need to be represented. A simplistic solution is to take punctuation and spaces as demarcating the boundaries of the elementary tokens we need to compute with, and it is common to regard the resulting word tokens as opaque symbols without substructure.

One measure for the suitability of such tokenisation is how readily an NLP system would encounter a token not observed in the corpus of text it was trained on. That is, how much coverage does a corpus provide relative to the hypothetical set of unique tokens forming the vocabulary of the language under consideration? Aside from phenomena such as slang, technical jargon and proper nouns, coverage depends heavily on what semantic and grammatical information a given language encodes into individual word forms. Languages that are closer to the analytic end of the linguistic typology spectrum tend to construct word forms from single or very few morphemes. At the other extreme are synthetic languages, which combine many morphemes into a single word form, either by chaining them together (agglutinative) or fusing the information of multiple conceptual morphemes into a single affix.

English is a suitable instance of a language that is relatively analytic, as it uses a fairly simple system of inflection and conjugation. Grammatical relations are largely encoded by dedicated function words (e.g. *the* and *of*) and syntactic structure. Few grammatical distinctions are marked directly on words—for example, English nouns are inflected only for their grammatical number, with the plural typically indicated by an *-s* suffix. In more analytic languages the assumption of opaque symbols is therefore quite reasonable. Although as the derivational example of *unkingly* shows, this assumption is an oversimplification even for morphologically ‘simple’ English.

The same assumption is not well-suited to more synthetic languages. Czech, for example, marks two number and seven case distinctions on nouns using a single fused suffix, giving rise to ten different inflectional variants of the word *král* (‘king’), including *krále*, *králi*, *králem*, *králové*, etc.[1] Or in the case of strongly agglutinative languages, it is possible to create new word forms by the repeated addition of suffixes, theoretically without bound. Turkish is a classic example of this:[2]

ev	(‘house’)
eviniz	(‘your (pl.) house’)
evinizdeyim	(‘I am at your house’)
evinizdeymişim	(‘I was apparently at your house’)

---

[1] From `http://en.wiktionary.org/wiki/král`, accessed 15/04/2014.

[2] From `http://en.wikipedia.org/wiki/Turkish_language`, accessed 15/04/2014, verified with the analyser of Çöltekin (2010).

Certain Germanic languages follow a similar regime in the formation of compound nouns, as in the German example:

Regen	(‘rain’)
Regenschirm	(‘umbrella’)
Regenschirmhersteller	(‘umbrella manufacturer’)
Regenschirmherstellergewerkschaft	(‘umbrella manufacturers trade union’)

To assume that the words in these examples are atomic symbols ignores the very apparent ways in which they share sub-structure. A computational system that ignores sub-word structure is likely to obtain poor coverage of the vocabulary of a language from a corpus, since morphological processes readily create forms that are very rare or non-existent in a given corpus.

A central theme in this dissertation is therefore to develop methods that account more explicitly for the observed grammatical and semantic links between word forms that share morphemes in their orthographic representations. The notion of training language processing systems from example corpora provides a bridge to the other defining element of the dissertation, which we turn to next.

A computational system that processes language in a useful way is tasked with more than merely representing the right symbols. Approaches to the larger question of casting the rules and idiosyncrasies of natural languages in algorithmic form can be characterised based on two extremes. At the one extreme lie knowledge-intensive approaches that explicitly specify the symbols and rules relevant to a language processing task, e.g. morphological analysis or translation, according to linguistic theories or hand-crafted heuristics. This can yield systems that work predictably and to a high standard, especially when the application domain is very limited. But this approach has various downsides, foremost of which is that it requires substantial additional labour to transfer a solution to new domains or other languages. Moreover, a reliance on dictionaries and prescriptive rules does not acknowledge the dynamic nature of human language, where new words and expressions arise continually.

At the other extreme are approaches that try to equip computer systems with a learning ability, rather than supplying them directly with linguistic abilities. The premise is that an appropriate combination of learning algorithm and data can produce a level of language processing that is useful. In many instances, we can assume that examples of language use (data) arise faster and more organically than linguistic expertise or linguists, which speaks in favour of approaches that rely on data. This notion has been a driving force in the success and prominence of statistical approaches to machine translation and speech recognition over the past two decades.

Model-based, data-driven approaches overcome some of the disadvantages of knowledge-intensive methods. In principle, they scale more readily across different languages and domains if adequate data is available. This latter condition is key. To clarify it further, the performance of NLP systems in this general category is a function of both model assumptions and data. The transferability, however, also makes it tempting to accept that a given set of model assumptions apply sufficiently well to multiple languages, which is not in general the case.

A particular case where a one-size-fits-all scheme is problematic is that of variations in morphological complexity. Many statistical models of language adopt the token-based view that we argued is ill-suited for morphologically rich languages. The fundamental problem is that, at the level of word tokens, many legitimate word forms occur very rarely, or do not occur at all, in a training corpus. This data sparsity makes vocabulary coverage poor and hampers the robustness of parameter estimates, both factors which constrain the performance of a model when processing new data.

This dissertation argues in favour of integrating high-level intuitions about morphology into the assumptions of statistical language models in order to address the aforementioned challenges posed by token-based processing of morphologically rich languages. The guiding principle is that morphologically related words should share statistical strength—for example, separate observations of the words *king* and *unlikely* should trigger the ability of a model to produce an informed response to an unknown word like *unkingly*, and instances of *králi* and *královÉ* should inform system behaviour towards *králem*.

1.1 Problem statement

The preceding section framed this dissertation in terms that are meant to be understandable to a broader audience, which necessarily introduces some imprecision. In this section we now state and motivate the thesis in more precise terms.

Our thesis is that basic linguistic intuitions about how words are formed in morphologically rich languages can be exploited in the formulation of probabilistic models that perform better at their tasks. Probabilistic modelling offers a coherent mathematical setting in which to express assumptions. It allows difficult tasks to be reformulated accurately in terms of simpler tasks, e.g. "What is the probability of observing this sentence?" can change to the more approachable "What is the probability of observing each word in the sentence?". Similarly, probabilistic models can be extended and combined in a principled way, limiting the number of dials that have to be set manually in heuristic approaches.

We deal with language modelling tasks of two kinds. The first is the narrow sense that *a language model (LM) assigns probabilities to word sequences*. This is the sense in which the term *language model* is widely understood in the literature, and which will henceforth be implied in this dissertation unless otherwise stated. A token-centric view in language models is at odds with morphologically rich languages, where sub-word elements are correlated in meaningful ways across different words. By accounting for such correlations in the design of LMs, making use of morphological information that we assume is given, we show in this dissertation how the sparsity in the training data of morphologically rich languages can be mitigated to improve their predictive performance. Predictive performance is quantified primarily in two ways. The first is to measure a LM's intrinsic ability to predict word sequences in held out test data, which we do in terms of the perplexity metric. The second is to consider what effect our morphology-targeting LM extensions have when situated in a machine translation system. In this context, LMs play a crucial role in biasing system output toward translations that are more fluent renditions of the target language. LMs that encode patterns at the sub-word level are by definition better suited for this role when translation is into morphologically rich languages. We thus measure translation quality in such settings by comparing system output to reference translations, using the automated quality metric BLEU.

The second type of modelling task we consider is that of inducing meaningful sub-word structures from raw text or word lists. Whereas the previously outlined task is primarily one of sequence-based density estimation, this second sense of modelling natural language focuses on the unsupervised learning of word structure, independent of actual word usage in sentences. A dominant approach in this task is to model words as sequences of morphemes, with the goal of learning what those morphemes are and how to segment a given word accordingly. This type of language modelling (in the broad sense) is of practical interest in various other subproblems in NLP, including part-of-speech-tagging, syntactic parsing, or indeed, for integrating sub-word level information into LMs, as we do in the first part of this dissertation. The idea that basic linguistic intuitions can inform and improve the design of a model finds expression in morphology learning by observing that some morphological processes involve discontiguous sub-word elements. We exploit this to devise a model that goes beyond linear segmentation to capture elements of non-concatenative morphology as well. As an example, in Arabic, the words *kitab* and *kutub* share the same root k·t·b. We show that modelling such discontiguous morphemes can improve the segmentational component of a model, and that it can identify part of a lexicon of such morphemes given merely an unannotated list of words.

1.2 Dissertation outline

The dissertation is structured as follows.

Chapter 2 provides further background central to understanding the rest of the material, and motivates the technical approaches followed.

Chapters 3 and 4 document two different approaches to integrating morphological information into probabilistic LMs. Chapter 3 introduces a hierarchical Bayesian model that accounts for the productive compound word formation attested in certain

Germanic languages. Chapter 4 introduces a method for integrating morphological information into probabilistic LMs based on distributed feature representations. The method is demonstrated to be effective across a range of languages featuring different levels of morphological complexity.

In Chapter 5, the focus shifts to language modelling in the broader sense, and we present an unsupervised approach to learning contiguous and discontiguous morphemes from raw text. The approach is applied primarily to Semitic languages, which make use of both kinds of morphemes.

Finally, Chapter 6 draws together the work of the preceding three chapters to summarise our main findings and present possible avenues for further research.

The remainder of this chapter highlights the specific research contributions presented in this dissertation.

1.3 Main research contributions

- **New generative model of productive compounding**

  We present a new language model where the formation of compound words in terms of their theoretically unbounded number of components is part of the generative process. This is done by extending the Hierarchical Pitman-Yor Language model with an additional back-off level where compound heads are conditioned on the sentential context, a decision based on grammatical considerations, while compound modifiers are generated by an additional language model. The effect is to reduce data sparsity due to compounding, and it results in higher precision outputting of compounds from a translation system using our language model. This work was originally presented at the 2012 EACL student workshop (Botha, 2012) and published in the proceedings of COLING (Botha et al., 2012).

- **Unsupervised learning of distributed morpheme representations**

  We present a simple technique for learning distributed feature representations for morphemes (given an external source of morphological segmentation) as part of a distributed language model. We show that the resulting morpheme vectors can be composed through addition to form word vectors that perform well in replicating human judgements of the semantic similarity of pairs of words. The benefit lies in being able to go beyond the vocabulary of the original source of the word vectors.

- **Distributed language modelling for rich morphology**
  The model within which the aforementioned morpheme representations are obtained is a novel variant of the log bilinear language model. Its key property is to provide a soft tying of parameters for words that share morphological content. We show that this is particularly beneficial for modelling rare words.

- **Integration of normalised distributed language model into a machine translation decoder**
  We provide the first demonstration, to our knowledge, of integrating a normalised distributed language model into a machine translation decoder. This and the previous two contributions listed here form part of work accepted for presentation and publication at ICML 2014 (Botha and Blunsom, 2014).

- **Joint representation of contiguous and discontiguous morphemes**
  We formulate a novel approach to the problem of representing concatenative and non-concatenative morphology in a unified fashion. This is done by applying mildly context-sensitive simple Range Concatenating Grammars (SRCGs), and we illustrate the approach on Semitic morphology. This and the next contribution listed here were originally reported in the proceedings of EMNLP 2013 (Botha and Blunsom, 2013).

- **Formulation of mildly-context sensitive adaptor grammars**
  In order to do unsupervised learning within our SRCG-based approach to modelling morphology, we formulate an extension of adaptor grammars to the SRCG-formalism. This provides an additional method for inducing SRCG

grammars and their equivalents, which could be useful in applications beyond morphology.

Chapter 2

Background

Chapter Abstract

This chapter introduces background material that the rest of the dissertation builds on. We formulate the key aspects of sequence-based statistical language models (LMs) and their evaluation. We present two alternative perspectives on language modelling by introducing back-off n-gram models and distributed language models. This sets up a short discussion of some existing methods for integrating sub-word level information into those LMs, and what their limitations are. The latter half of the section introduces aspects of Bayesian probability theory and non-parametric modelling that are relevant in subsequent chapters. Although most terminology and notation will be introduced when needed, we begin this section with some preliminaries.

2.1 Preliminaries

This dissertation relies heavily on probability theory, especially as defined for discrete events. A discrete random variable X can take on any value from an event space $\Omega = \{x_1, x_2, \ldots\}$, which encodes the potential outcomes of a stochastic operation or process. A simple example is that of tossing a coin, where $\Omega = \{\text{heads}, \text{tails}\}$. In general, Ω can be a countably infinite set.

A probability mass function $P : \Omega \rightarrow [0, 1]$ maps a particular value x_i of X to the probability of that the event $X = x_i$ occurs, namely $P(X = x_i)$. We will sometimes abbreviate this as $P(x_i)$, taking the random variable assignment as implicit.

This function characterises the distribution of the random variable, and accordingly we usually refer simply to a "probability distribution". The event space Ω is also referred to as the support of the distribution.

2.2 Statistical language modelling

The canonical task in statistical language modelling is to assign a probability to a sequence of words, e.g.

$$P(\textit{the sky is grey}) = ?$$

Such probabilities are inferred by using statistical estimates from data, in other words, the model is *trained* on a corpus of text in the language of interest.

LMs play a crucial role in applications like speech recognition, machine translation and predictive text input, where an intended utterance must be decoded from an ambiguous signal. LMs that better capture the regularities and idiosyncrasies of natural language are better at discriminating among alternative output candidates that arise in these applications. They are not directly concerned with the input signal (audio features, a foreign language, or a sequence of keystrokes), but instead reward output that is a more fluent and coherent rendition of the target language.

The fundamental challenge in assigning probabilities to word sequences is to contend with the fact that the set of possible word sequences is infinite, while any given data set used for estimation is finite. The thesis investigated here is premised on the observation that morphological processes have a strong effect on the sparsity of training data. Broadly speaking, more complex morphological processes exacerbate data sparsity. The overall approach of modelling sub-word elements is therefore intended to help overcome data sparsity. The motivation is that this is likely a more viable long term solution than estimating morphologically blind models from ever more data, effective as that may be for particular languages in the short term (Brants et al., 2007).

The dominant strategy[1] in LM estimation is to decompose the probability $P(\mathbf{w})$ of a word sequence $\mathbf{w} = w_1 \ldots w_L$, using the chain-rule, as

$$P(\mathbf{w}) = \prod_{i=1}^{L} P(w_i \mid w_1, \ldots, w_{i-1}) \,. \tag{2.1}$$

The task of estimating probabilities for entire word sequences of arbitrary length is thereby simplified into one of estimating probabilities for individual words, conditioned on all the preceding words in the sequence. An approximation that further simplifies this is based on the assumption that a particular word only depends directly on the ones immediately preceding it. A word sequence can accordingly be modelled as a Markov chain of order $n-1$:

$$P(\mathbf{w}) \approx \prod_{i=1}^{L} P(w_i \mid w_{i-n+1}, \ldots, w_{i-1}) \,. \tag{2.2}$$

The sequence of n words $(w_{i-n+1}, \ldots, w_i)$ is commonly referred to as an n-*gram*, with *history* $h = (w_{i-n+1}, \ldots, w_{i-1})$. We use the short-hand notation for a word sequence, $w_i^j \equiv (w_i, \ldots, w_j)$. Alternative terms that we use are that h is the *context* from which the *target word* w_i must be predicted. The support of the conditional distribution $P\big(w_i \mid w_{i-n+1}^{i-1}\big)$ is the vocabulary $\mathcal{V}$, the discrete set of word types to be modelled.

Sequence boundaries require some special handling in Markov chains. Common practice is to assume that the conditioning context of the first token w_1 in a sequence is provided by a special padding symbol, while the use of a designated symbol marking the end of the sequence allows for coherent modelling of variable-length sequences.

We often need to compute the probability of a set of word sequences, such as sentences in a corpus. If $\{\mathbf{w}_1, \ldots, \mathbf{w}_N\}$ are N independent word sequences, then the joint probability (using the aforementioned Markov assumption) is

$$P(\mathbf{w}_1, \ldots, \mathbf{w}_N) = \prod_{j=1}^{N} P(\mathbf{w}_j) = \prod_{j=1}^{N} \prod_{i=1}^{|\mathbf{w}_i|} P\big(w_{j,i} \mid w_{j,i-n+1}^{j,i-1}\big) \,,$$

---

[1] Other strategies also exist, e.g. the whole-sentence model of Rosenfeld et al. (2001).

In subsequent formulations we refer to a corpus simply as one sequence of tokens, w_1^M, taking these mechanics of boundary symbols and sentence independence as being implied.

The reliance on the conditional probability distributions $P\big(w_{i-n+1}^{i-1}\big)$ makes parameter estimation more tractable and is advantageous for the dynamic programming employed by machine translation (MT) decoders. We consider language model estimation in §2.2.2. For now, we assume an estimated model P_{LM} is given and consider how its quality can be evaluated.

2.2.1 Language model evaluation

A common evaluation method is to measure how well the language model P_{LM} predicts previously unseen test data w_1^N. *Perplexity* is a useful metric for this purpose, as it is well-defined for any length N and allows for a fair comparison of models in spite of differences in their assumptions or internal workings. The perplexity of a model P_{LM} on the test sequence w_1^N is

$$\text{ppl} = 2^{H(P_{LM})}, \tag{2.3}$$

where H is the cross-entropy

$$H(P_{LM}) = \sum_{i=1}^{N} -\frac{1}{N} \log_2 P\big(w_i \mid w_1^{i-1}\big) . \tag{2.4}$$

The intuitive interpretation of a perplexity value of k is that the uncertainty of the model with respect to the test data is equivalent to the uncertainty in the outcome of throwing a k-sided fair die. A model is thus on average k-ways perplexed by the observation of each test word. Lower perplexity corresponds to lower uncertainty, which is the mark of a better model.[2] Note that perplexity, as defined above, is independent of model assumptions such as the Markov assumption applied in the formulation of n-gram models.

A comparison of two models P_{LM_1} and P_{LM_2} in terms of their perplexity on a given data set is only fair if the support of the probability distributions is equivalent,

---

[2] For a basic mathematical derivation of perplexity in terms of entropy, see (Koehn, 2010, ch. 7).

in other words, if they model the same vocabulary $\mathcal{V}$. This matter requires special attention when going beyond the standard LM assumption that $\mathcal{V}$ is a finite set, as we do in Chapter 3.

2.2.2 Two dominant approaches to Markov-based LMs

The focus now turns to two dominant approaches to modelling the categorical conditional distributions $P\big(w_i \mid w_1^{i-1}\big)$ used in Markovian language models.

1) One approach is to infer these distributions by using exact-matching of n-grams, thereby directly representing the high-dimensional discrete space of n-grams. For historical reasons, this class of models are referred to simply as *n-gram language models*.

2) The other major approach is to embed words in a low-dimensional, continuous vector space and learn a function that uses the arising word vectors to map an n-gram history into a probability for the target word w_i. These *distributed language models* are also known by the terms neural, connectionist or continuous-space language models.

We instantiate the thesis that basic linguistic intuitions about word formation are beneficial in modelling morphologically rich languages within each of these two approaches. The next two sections present them in more detail.

2.2.2.1 n-gram language models

A discrete n-gram LM is in fact a collection of conditional multinomial distributions like $P\big(w_i \mid w_{i-n+1}^{i-1}\big)$, estimated directly from the empirical distribution of n-grams in training data. Maximum likelihood estimates suffer a variety of problems: Some n-grams are not observed in the training data yet a model must still be able to score them. n-grams that do appear in the data may do so with a low frequency that is not representative of their true distribution in the language (Good, 1953). To address these and other more subtle challenges arising from data sparsity, a wide variety of techniques have been devised to smooth the empirical n-gram distributions by redistributing probability mass (Chen and Goodman, 1998).

The interpolated modified Kneser-Ney (MKN) model (Kneser and Ney, 1995; Chen and Goodman, 1999) is considered the state-of-the art discrete n-gram LM (Chelba et al., 2014). It serves as a baseline model in the empirical evaluations reported later in this dissertation. We briefly introduce key smoothing concepts it relies on, some of which have analogues in the models we subsequently develop.

The first concept is that of basing probability estimates of a certain random event on more general random events. Following on from the initial example in this chapter, we can regard the observation of the word *grey* in the context *sky is* as a more specific event than observing it in the shorter context *is*. That is, the bigram *is grey* is a generalisation of the trigram *sky is grey*. Thus to overcome the data sparsity inherent in estimating n-gram LM parameters, an effective technique is to let $P_{LM}(\textit{grey} \mid \textit{sky}, \textit{is})$ be a function of $P(\textit{grey} \mid \textit{is})$.

More formally, defining $P_{LM}\big(w_i \mid w_{i-n+1}^{i-1}\big)$ as a function of lower-order conditional distributions $P_{LM}\big(w_i \mid w_{i-n+2}^{i-1}\big)$ can be done through interpolation (Jelinek and Mercer, 1980) or by strictly switching to the lower-order model (Katz, 1987). Interpolation typically works better (Chen and Goodman, 1998), so we restrict the attention to its functional form:

$$P_{LM}\big(w_i \mid w_{i-n+1}^{i-1}\big) = \alpha\big(w_{i-n+1}^{i}\big) + \gamma\big(w_{i-n+1}^{i-1}\big)\, P_{LM}\big(w_i \mid w_{i-n+2}^{i-1}\big)\,, \tag{2.5}$$

where $\alpha\big(w_{i-n+1}^{i}\big)$ denotes a smoothed version of the n-gram conditional distribution (see below), while the γ parameters are set to normalise the overall probability distribution, $P_{LM}\big(w_i \mid w_{i-n+1}^{i-1}\big)$. This procedure can be followed recursively and bottom out with a reliance on unigram model $P_{LM}(w_i)$.

The matter of defining $\alpha(\cdot)$ provides the connection to the second important smoothing concept of *discounting*. Instead of relying directly on the actual number of times an n-gram is observed in the training data, $C(w_i^j)$, estimation is done using counts that are adjusted down, to give a discounted frequency $C_{disc}(w_i^j) \leq C(w_i^j)$. MKN uses absolute discounting (Ney et al., 1994) to adjust an n-gram count by a constant D_m,

$$C_{MKN}(w_{i-n+1}^{i}) = \max\big(C(w_{i-n+1}^{i}) - D_m, 0\big)\,, \tag{2.6}$$

where $m \in \{1, 2, 3+\}$ designates one of three discount levels according to whether the n-gram occurs once, twice, or more than twice.

MKN combines discounting and (interpolated) back-off in a particular way. $\alpha(\cdot)$ is the discounted maximum likelihood estimate

$$\alpha\big(w_{i-n+1}^{i}\big) = \frac{C_{MKN}(w_{i-n+1}^{i})}{C(w_{i-n+1}^{i-1})} \tag{2.7}$$

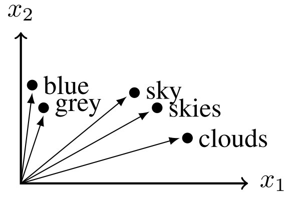
The remaining element for using Equation 2.5 is to specify the back-off distribution $P_{LM}\big(w_i \mid w_{i-n+2}^{i-1}\big)$. Kneser-Ney smoothing uses purpose-built lower-order distributions that exploit the intuition that the diversity of contexts a word appears in often represents a more accurate view of its true probability than the n-gram count itself. That is, the empirical prevalence of the n-gram *Los Angeles* overstates the true probability of the word *Angeles*, so Kneser-Ney smoothing bases the estimate instead on the number of unique n-gram histories the word appears in.

n-gram language models, and MKN smoothing in particular, have been relied on heavily in the domain of machine translation in the last 15 years, and have more recently received significant amounts of engineering attention in order to scale to vast training data sizes (Heafield, 2011; Heafield et al., 2013). A main limitation of MKN is that it is fundamentally heuristic-based, albeit heuristics derived from rigorous empirical work. This makes it challenging to incorporate new smoothing ideas. Teh (2006a) proposed a Bayesian approach to n-gram language modelling which implements some of the same principles that MKN depends on, but offers a framework that better lends itself to modifications of the basic smoothing behaviour. In Chapter 3 we use that framework to create a more sophisticated smoothing method which exploits simple linguistic intuitions about word formation.

2.2.2.2 Distributed language models

A salient feature of the n-gram models in the previous section is that they parametrise a high-dimensional discrete space directly, based on the identities of words. This amounts to a harsh binary perspective on the way words are correlated in text by requiring that a test n-gram (or its backed-off version) exactly match an event in the training data.

An alternative approach that acknowledges a more multifaceted notion of correlation among word occurrence is to represent words in a multidimensional vector space. Thus instead of each word type v in the vocabulary $\mathcal{V}$ being merely a symbol in a discrete set, each v has an associated real-valued *distributed feature vector* $\mathbf{r}_v \in \mathbb{R}^d$. This immediately enables reasoning about the relations among words in terms of how they relate in vector space, with the intuition that "similar" words are closer to each other, using a metric such as the angle or Euclidean distance between their representation vectors. As an example, the words of the vocabulary fragment $\mathcal{V} = \{\text{blue}, \text{clouds}, \text{grey}, \text{skies}, \text{sky}, \dots\}$ could have feature representations that look as follows when plotted in two-dimensional space:



In general, the different dimensions have no intrinsic meaning, although this may depend on the method used for obtaining distributed representations. For example, in the area of distributional semantics, vector representations of words have been constructed directly from corpus-based co-occurrence statistics, using the most frequent words as basis vectors (Bullinaria and Levy, 2007; Mitchell and Lapata, 2008, *inter alia*). However, we consider models where the word vectors are learnt as part of LM training, and the different dimensions simply capture different correlates.

Distributed language models (DLMs) apply distributed feature representations in assigning probabilities to word sequences. The general idea can be stated as follows in terms of n-gram modelling: An n-gram history w_{i-n+1}^{i-1} is mapped to a vector $\mathbf{x} \in \mathbb{R}^{d'}$ as some function $f(\cdot)$ of its word vectors, i.e. $\mathbf{x} = f(\mathbf{r}_{w_{i-n+1}}, \dots, \mathbf{r}_{w_{i-1}})$, where d' is free to differ from d. A further function $g(\cdot)$ then transforms $\mathbf{x}$ into a conditional probability: $P_{LM}\big(w_i \mid w_{i-n+1}^{i-1}\big) = g(w_i, \mathbf{x})$. A variety of neural network architectures and related methods have been used to model the functions f and g while jointly learning the representation vectors (Xu and Rudnicky, 2000; Bengio et al., 2003; Morin and Bengio, 2005; Mnih and Hinton, 2007; Mikolov et al., 2010).

The formulation above is a simplification of the *neural probabilistic language model* (Bengio et al., 2003), but is sufficient to highlight the key advantages of DLMs over the discrete n-gram LMs of the previous section. The main benefit is that, by making the conditional probabilities smooth functions of the n-gram context, generalisation to new n-grams is more automatic. It becomes possible for two lexically distinct phrases such as *the grey clouds* and *the blue sky* to effect similar conditioning on an immediately following word, assuming the word vectors involved are pair-wise "nearby" in the distributed representation space as demonstrated in the preceding diagram. This obviates the need for explicit back-off strategies. In more general terms, DLMs mitigate the sparsity problem associated with the high-dimensional parameter space of discrete n-gram LMs (which is in principle $|V|^n$) by operating in a lower dimensional continuous space. In particular, $d \ll |\mathcal{V}|$, and values of 50 to 1000 are typical for d, whereas typically $|\mathcal{V}| > 10^5$, or much larger for morphologically rich languages.

This dissertation addresses two issues in DLMs. The first is to consider how they can be improved for morphologically rich languages by using the internal structure of words, and the second is to demonstrate the full integration of DLMs into a machine translation system (chapter 4).

2.2.3 Language modelling at the sub-word level

Both approaches to language modelling presented above usually consider words as the fundamental modelling unit. They seek to mitigate data sparsity in different ways, but do not directly address a major source of that data sparsity, which is that of complex morphology. Morphological complexity comes in many guises, but the common outcome is that languages with more complicated rules for forming words tend to have larger vocabularies of surface word forms. This exacerbates the data sparsity to which n-gram LMs are prone to by definition, while DLMs are also stretched by having fewer training instances of individual word forms from which to learn suitable representation vectors and the accompanying probability function.

A further implication of rich morphology is that test data can be expected to contain relatively many word forms not appearing in the training vocabulary.[3] The standard practice for dealing with this out-of-vocabulary (OOV) problem is to set the model up in a way that reserves some probability mass for a generic "unknown-word" symbol, which we will denote as UNK. This has the shortcoming of treating truly novel words, such as proper nouns, the same as previously unseen morphological variants of a known word, ignoring the obvious relation among them.

Both these problems can be addressed by shifting the fundamental unit of representation to be at the sub-word level. The sought-after effects is that morphologically related word forms should share statistical strength in some way and that some OOVs can be handled in terms of their sub-word structure.

Previous approaches to integrating morphological information into language models have been proposed.[4]

A naïve strategy is to segment words into their morphemes in a preprocessing step, and then construct otherwise unmodified language models from the modified tokenisation. As an example of decomposition, the sentence

The forecasters predicted more flooding

may be segmented into morphemes to give

The forecaster s predict ed more flood ing

This technique reduces the number of unique token types, which directly addresses sparsity, but introduces its own problems: It begs the question of how aggressively to segment—this often opens the door to ad-hoc heuristics that do not transfer reliably across multiple tasks or languages. And it disrupts the contextual conditioning in back-off n-gram models by making a hard switch away from modelling sentential

---

[3] Mismatches in the domains of text are another major source of out-of-vocabulary words in test data.

[4] The focus here is on the types of language models introduced in §2.2.2, but we note that maximum-entropy language models also offer a useful way of integrating morphological information, e.g. (Minkov et al., 2007).

coherence, say via $P(\textit{predicted} \mid \textit{the}, \textit{forecasters})$, toward modelling fragments, say $P(\textit{ed} \mid s, \textit{predict})$. This strategy has been used in speech-recognition (Geutner, 1995; Ircing et al., 2001; Hirsimäki et al., 2006), and machine translation (Virpioja et al., 2007).

A more systematic approach is that of *factored language models* (FLMs) (Bilmes and Kirchhoff, 2003; Kirchhoff et al., 2008). FLMs cast each word as a bundle of K features $w_i = (f_i^{(1)}, \ldots, f_i^{(K)})$ over which a variety of conditional probability distributions are defined. To clarify the notation, a standard n-gram model trivially uses word form as its single feature ($K = 1$) and the n-gram conditional distribution is written as $P\left(f_i^{(1)} \mid f_{i-n+1}^{(1)}, \ldots, f_{i-1}^{(1)}\right)$. The intention is to allow for the integration of information beyond word forms. For example, a word's part-of-speech and stem could be used as additional features, $w_i = (f^{\text{surface}_i}, f^{\text{PoS}_i}, f^{\text{stem}_i})$, as in $\overrightarrow{\text{cloudy}} = (\textit{cloudy}, \textit{noun}, \textit{cloud})$ ($K = 3$).

This factored representation bears superficial resemblance to the distributed feature representations introduced in §2.2.2.2. The main difference is that the FLM features are discrete random variables over which arbitrary directed graphical models are defined. Furthermore, feature values are likely set with reference to some external source and not trained as part of the FLM.

A FLM that goes beyond standard n-gram conditioning might incorporate a submodel $P\left(f_i^{\text{surface}} \mid f_{i-1}^{\text{stem}}\right)$, which conditions the target word on the stem of the preceding word. Multiple conditional distributions of this kind can be combined, similar to the way standard n-gram models rely on lower-order conditionals. But with the richer information encoded, an important question becomes how to prioritise the back-off sequence. Bilmes and Kirchhoff (2003) proposed solving this by generalising back-off as a procedure that happens on a graph (instead of the linear procedure of dropping context words as in back-off n-gram models). They rely on genetic algorithms to acquire the structure of the back-off graph over the K feature dimensions and n sequence positions in an n-gram approach, while the individual conditional distributions use standard smoothing or discount algorithms.

The factored language model approach has also been applied to neural probabilistic language models (Alexandrescu and Kirchhoff, 2006), which, as pointed out before, remove the necessity to deal with back-off paths. Associations among different features are learnt as part of the network weights rather than having to be specified explicitly as present or absent. These *factored neural language models* (FNMLs) in effect associate a distributed representation with each feature component of a word, so the overall word representation $\mathbf{r}_w = [\mathbf{f}^{(1)}; \ldots; \mathbf{f}^{(K)}]$ is constructed by concatenating the feature vectors $\mathbf{f}^{(k)} \in \mathbb{R}^{d(k)}$, where $1 \leq k \leq K$ and the dimensionality $d(k)$ may vary across components.

FLMs are a suitable vehicle for integrating external information into language models, including morphology.[5] They can lower perplexity (Bilmes and Kirchhoff, 2003; Alexandrescu and Kirchhoff, 2006; Wu et al., 2012) and improve speech recognition accuracy (Vergyri et al., 2004).

The main shortcoming of FLMs is that the number of factors K is a fixed constant for all words. Therefore they cannot naturally encode variable-length information, such as the surface-level morphemes of a word. In largely fusional languages, such as Czech, there is likely no loss in assuming a fixed-length segmentation, say prefix + stem + suffix.[6] But a better solution is needed for agglutinative languages or phenomena where the number of morphemes per word can vary substantially across the distinct surface words of a given language.

In Chapters 3–4, we present two separate language modelling approaches that might be regarded as variations on the idea of FLMs but that have inherent support for variable-length decompositions of words.

2.2.4 Summary

The preceding section set up some terminology and definitions used in statistical language modelling. We introduced the two widely used approaches of back-off

---

[5] As an example of FLMs using non-morphological information, Adel et al. (2013) applied them successfully to the problem of code-switching by incorporating a language-ID factor for each word.

[6] Another option is to apply FLMs to morpheme-segmented data, so that each morpheme is assigned a set of factors (Mousa et al., 2011).

based n-gram LMs and vector-based DLMs. Finally, we discussed an existing general approach to integrating morphological information into both types of language models, and pointed out that it does not naturally support strong agglutination.

2.3 Non-parametric Bayesian modelling

This section introduces the key elements of the mathematical framework employed in Chapters 3 and 5.

Bayesian modelling centres around two concepts that address the questions of inferring the parameters θ of a model given observed data D, and making predictions about a new data point $\tilde{x}$ under the model. First, instead of working with point-estimates of θ, the Bayesian approach quantifies the uncertainty about the value of θ by means of probability distributions over its potential values. The *prior distribution* $P(\theta \mid \alpha)$ expresses our beliefs about what the value of θ might be in absence of observed training data. The hyperparameters α control the behaviour of the prior distribution.

The objective, however, is to make inferences in light of observed data. The *posterior distribution* $P(\theta \mid D, \alpha)$ quantifies the uncertainty in the values of θ following the observation of training data D. The prior and posterior distributions are related by Bayes' law as

$$P(\theta \mid D, \alpha) \propto P(D \mid \theta)\, P(\theta \mid \alpha), \tag{2.8}$$

where the term $P(D \mid \theta)$ is referred to as the *likelihood distribution* of the data D conditioned on the model parameters θ. This factorisation is often useful in problem settings where the terms on the right-hand side of (2.8) are computed more easily than directly computing the posterior probabilities on the left-hand side.

Our focus falls instead on the modelling convenience afforded by the specification of a prior distribution. Judicious choice of prior distribution provides a principled way to encode external knowledge or intuitions relevant to a problem into a probabilistic model. The effect of the prior on the posterior distribution is strongest when little or no data has been observed. The latter two properties of Bayesian modelling makes it especially suitable for our aims of overcoming the data sparsity problem in morphologically rich languages by encoding sub-word information into models, which we can do through constructing appropriate Bayesian priors.

The second central concept of the Bayesian approach is posterior inference. The prediction of a new data point $\tilde{x}$ is approached by relying on all possible values of the parameters θ rather than a point-estimate. That is, the *posterior predictive distribution* $P(\tilde{x} \mid D, \alpha)$ is determined by marginalising the posterior distribution over θ:

$$P(\tilde{x} \mid D, \alpha) = \int P(\tilde{x} \mid \theta) P(\theta \mid D, \alpha) \, \mathrm{d}\theta. \tag{2.9}$$

For many types of models, this integral cannot be solved analytically nor computed tractably. The two major solutions to this impasse are to approximate the posterior distribution with a simpler function which enables exact numerical optimisation (Beal, 2003), or to use Markov chain Monte Carlo (MCMC) techniques to approximate the integral with a finite number of samples drawn from the true distribution (Neal, 1993; Gilks et al., 1996). Our work follows the latter option.

2.3.1 Pitman-Yor process prior

A central component of the Bayesian modelling work in this dissertation is the use of the Pitman-Yor process (PYP; Pitman, 1995; Pitman and Yor, 1997; Ishwaran and James, 2001) to construct prior distributions that encode sub-word structure. This follows widespread and successful application of the PYP in NLP tasks including unsupervised learning of grammars (Cohn et al., 2010; Levenberg et al., 2012; Cohen et al., 2010), word segmentation (Mochihashi et al., 2009; Neubig et al., 2010), morphology (Goldwater et al., 2006), parts of speech (Blunsom and Cohn, 2011) and n-gram language modelling (Teh, 2006a; Wood and Teh, 2009).

The suitability of the PYP in these diverse applications, and indeed in this dissertation, arises from two properties. The first is that it falls within the class of non-parametric statistical models (Wasserman, 2007), which means that the number of model parameters (or more generally, the model complexity) is driven by the observed data instead of being fixed *a priori*. Non-parametric modelling matches the boundlessness of many linguistic objects very well—words, morphemes and syntactic structures all seem to constitute countably infinite sets. The second key property of the PYP is that it naturally produces power-law distributions that model the distribution of word frequencies (Zipf, 1932) and that are useful for other linguistic elements (Goldwater et al., 2011).

The PYP is a stochastic process that generates distributions over exchangeable random partitions of a countably infinite set of elements. A PYP is specified by a *base distribution* P_0, and two hyperparameters: the *discount* a and *strength* b, with $0 \leq a < 1$ and $b > -a$. We write it as $\mathcal{PY}(a, b, P_0)$. Setting $a = 0$ reduces the PYP to the more familiar Dirichlet process (Ferguson, 1973) with concentration parameter $\alpha = b$. The role of the base distribution is considered in the next section, but we note that it determines the support of a draw G from the PYP, where G is itself an infinite dimensional probability distribution.

The PYP has various characterisations. An infinite dimensional distribution G sampled from a PYP can be constructed by a stick-breaking procedure (Sethuraman, 1994; Ishwaran and James, 2001). However, in our applications it is more useful to work with draws from G. The Pitman-Yor Chinese Restaurant Process (PYCRP; Ishwaran and James, 2003) provides a means of obtaining such a draw $\mathbf{y} \sim G$ and circumvents the problem of representing the infinite dimensional object.

The PYCRP can be defined through the metaphor of an unbounded sequence of customers entering a restaurant one-by-one and being seated at tables with unbounded seating capacity and of which there is a limitless supply. The table allocation of customer x_i is recorded as table index $z_i \in \mathbb{Z}^+$. Let $\mathbf{z}_n$ denote the sequence of table indices after n customers have been seated, and let n_k denote the number of customers seated at the k^{th} of the m non-empty tables in the restaurant. Hence $n = \sum_{k=1}^{m} n_k$. The first customer x_1 sits at the first table by definition, i.e. $z_1 = 1$. Given $n \geq 1$ previously seated customers, the next customer x_{n+1} is seated at table index $z_{n+1} \in [1, m+1]$, sampled from the probability distribution

$$P(z_{n+1} = k \mid \mathbf{z}_n, a, b) = \begin{cases} \dfrac{n_k - a}{n + b} & \text{if } 1 \leq k \leq m \\[2ex] \dfrac{am + b}{n + b} & \text{if } k = m + 1. \end{cases} \tag{2.10}$$

The dependence on n_k in the first case means customers are more likely to be assigned to tables that already have relatively more customers. However, assignment to a previously unoccupied table (case $k = m + 1$) always carries non-zero probability, and is itself dependent on the number of occupied tables m. The interaction of this preferential attachment dynamic and the positive attitude towards novelty induces a power-law distribution over the number of customers n_k at each table, provided $0 < a < 1$ (Goldwater et al., 2011).

The joint probability of a particular seating assignment $\mathbf{z}_n$ under the PYCRP is obtained as the product of the individual seating decisions:

$$P(\mathbf{z} \mid a, b) = P(z_1 \mid a, b) \cdot \prod_{i=2}^{n} P(z_i \mid \mathbf{z}_{i-1}, a, b) \tag{2.11}$$

$$= 1 \cdot \frac{(a(1) + b) \ldots (a(m-1) + b) \times \prod_{k=1}^{m} \big((1-a) \ldots (n_k - 1 - a)\big)}{(1 + b) \ldots (n - 1 + b)} \tag{2.12}$$

$$= \frac{\Gamma(1+b)}{\Gamma(1+b)} \cdot \frac{\prod_{k=1}^{m-1} (ak + b)}{(1 + b) \ldots (n - 1 + b)} \times \prod_{k=1}^{m} \left((1-a) \ldots (n_k - 1 - a) \frac{\Gamma(1-a)}{\Gamma(1-a)} \right) \tag{2.13}$$

$$= \frac{\Gamma(1+b)}{\Gamma(n+b)} \left(\prod_{k=1}^{m-1} (ak + b) \right) \left(\prod_{k=1}^{m} \frac{\Gamma(n_k - a)}{\Gamma(1-a)} \right). \tag{2.14}$$

The numerator in (2.12) accounts separately for the initial occupation of the tables and for the assignment of subsequent customers to each table. The final form is efficient for computation, making use of the Gamma function which is defined as $\Gamma(x) = \int_0^\infty u^{x-1} e^{-u} \, \mathrm{d}u$ for $x > 0$, with the identity $\Gamma(x) = (x-1)\Gamma(x-1)$ for real-valued $x > 0$.

2.3.2 Pitman-Yor process base distributions

A sequence of table assignments $\mathbf{z}_n$ under the PYCRP is fundamentally just a partition of a sequence of integers, $\{1, \ldots, n\}$, into m clusters such that they exhibit certain frequency distributions. For concrete modelling applications, the usefulness of this depends on what meaning is assigned to the customers and their tables. The role of the base distribution P_0 of the PYP is to provide that meaning. In the restaurant metaphor, the first customer x_i to be seated at table with index k triggers the sampling of a dish ℓ_k from the base distribution, $\ell_k \sim P_0$. All subsequent customers to join a non-empty table k share in the same dish ℓ_k previously chosen for that table.

If we define the identity y_i of a customer x_i at table z_i to be the table label ℓ_{z_i}, the effect is to produce a sequence $\{y_i\}$ distributed as follows:

$$\begin{aligned} y_i \mid G &\sim G \\ G \mid a, b, P_0 &\sim \mathcal{PY}(a, b, P_0) \end{aligned}$$

The conditional probability of the identity y_{n+1} for an additional customer x_{n+1}, given labels $\boldsymbol{\ell}$ for the n customers seated at m tables according to the seating arrangement $\mathbf{z}_n$, is

$$\begin{aligned} P(y_{n+1} = y \mid \mathbf{z}_n, \boldsymbol{\ell}, a, b) &= \sum_{k=1}^{m} \delta_y(\ell_k) \frac{n_k - a}{n + b} + \frac{am + b}{n + b} P_0(y) \\ &= \frac{N_y - am_y + (am + b) P_0(y)}{n + b}, \end{aligned} \tag{2.15}$$

where δ is the Kronecker delta indicator function, N_y is the number of customers seated at tables labelled $l_k = y$, and m_y is the number of such tables.

As an example,[7] the PYP can be used to define a simple unigram language model over a closed vocabulary $\mathcal{V}$. Let the base distribution $P_0 = \mathrm{Uniform}(|\mathcal{V}|)$ and regard a customer x as a generic word token. A label ℓ drawn from the base distribution and assigned to the token via the table it joins in the associated restaurant then supplies the token its identity as a particular word type from the vocabulary $\mathcal{V}$. Note that the

---

[7] This example derives from the connections between language model smoothing and the PYP pointed out by Goldwater et al. (2006); Teh (2006a).

same label ℓ can be drawn repeatedly from P_0. The term $N_y - am_y$ in the numerator of Equation (2.15) can be understood to discount the frequency N_y with which a word y occurs in a corpus by an amount am_y, while the second term provides interpolation with the uniform base distribution, in analogy to the language model smoothing described by Equation (2.5), p15. This smoothing ability forms the basis of the language model (Teh, 2006a) that we extend in Chapter 3.

The base distribution can, however, also be used to generate more complex labels than word identities. In particular, they can be used to generate trees of various kinds (Cohn et al., 2010; Johnson et al., 2007a; Levenberg et al., 2012), a line of work we extend to another grammar formalism in Chapter 5.

2.4 Summary

This chapter described the two approaches to sequence-based statistical language modelling that subsequent chapters extend by integrating information about subword structures. In the second part of the chapter, we introduced the aspects of Bayesian modelling with the non-parametric Pitman-Yor process priors that we employ in the next chapter and the penultimate chapter of this dissertation.

Chapter 3

A Hierarchical Bayesian Language Model for Compounding

Chapter Abstract

In this chapter we introduce an n-gram-based LM that exploits the sub-structure of single-token compound words to attain better smoothing. The LM extends the Hierarchical Pitman-Yor language model (Teh, 2006a) with the ability to provide probabilities for out-of-vocabulary compound words. This chapter is an extension of material originally published as (Botha, 2012) and (Botha et al., 2012).

Compounding is a process that forms words by combining two or more other words. For example, in English, the independent words *space* and *ship* may combine to form the noun *spaceship*, while *inspiring* and *awe* form the hyphenated adjective *awe-inspiring*. Many languages make use of compounding as an important vehicle for novel word formation. Our focus on compounding is motivated by the prevalence of *closed compounds* in certain languages. Such compound words are written as single orthographic units, and can pose problems to LMs, and NLP systems generally, that regard punctuation-delimited tokens as their elementary units of analysis. Closed compounds are especially problematic in languages that exhibit highly productive compounding, such as German, Dutch and Afrikaans. In these instances, compounding can occur recursively, theoretically without bounds. Word-based LMs are thus prone to suffer from sparse data effects that can be attributed to compounds specifically. The aim of the model we propose is to overcome data sparsity due to compounding by accounting for compound structure.

A compound is said to consist of a *head* component and one or more *modifier components*, with optional *linking elements* between consecutive components (Goldsmith and Reutter, 1998).

Examples of German compounds

- A basic noun-noun compound:

  Auto + *Unfall* = *Autounfall* ('car crash')

- *Linking elements* can appear between components

  Küche + *Tisch* = *Küchentisch* ('kitchen table')

- Components can undergo stemming

  Schule + *Hof* = *Schulhof* ('schoolyard')

- Compounding is recursive

  (*Regen* + *Schirm*) + *Hersteller*

  = *Regenschirmhersteller* ('umbrella manufacturer')

- Compounding extends beyond noun components

  Zwei-Euro-Münze ('two Euro coin')

  Fahrzeug ('vehicle') – from *fahren* ('to drive') + *Zeug* ('stuff/thing')

3.1 Structural dependencies

The linguistic intuition that we propose to exploit in our language model is that the head of a compound word determines the morphosyntactic properties of the whole word (Williams, 1981). We focus our investigation on German, which is predominantly right-headed. The right-most component of a compound thus determines the agreement requirements that needs to be fulfilled between the compound itself and other words in the sentence. For example, the *Bahn* ('track/rail') in the compound *Eisenbahn* ('railway') identifies the word as singular feminine, which determines the requirements for its agreement with verbs, articles and adjectives.

A language model could therefore give a reasonable assessment of the syntactic fluency of a sequence of German words by abstracting away the non-head components of compounds. For example, the sentence *I'm going by train* can be rendered in German as either of the following:

- Ich fahre mit der Eisenbahn.
- Ich fahre mit der Bahn.

Collapsing all compounds to their heads and ignoring modifiers would decrease sparsity and allow more robust n-gram probabilities to be estimated from data. But such a strategy would not be probabilistically sound as a generative model of a corpus. Moreover, a model that ignores modifiers would assign the same probability value to *Eisenbahn* and the empirically much rarer *Bobbahn* ('bobsled'), which would be unsatisfactory in a task where the language model plays a discriminative role. The model needs to account for the non-head components in some way. We expect the identity and number of modifier components to be strongly correlated with the identity of the head. In particular, the conditional distributions of modifier given head will be sharply peaked. A simple approximation is thus to assume that, conditioned on the head, modifiers are generated by a reverse n-gram model:

$$P(\textit{eisenbahn} \mid \textit{mit der}) \equiv P(\textit{bahn} \mid \textit{mit der}) \times P(\textit{eisen} \mid \textit{bahn}) \times P(\textit{\$} \mid \textit{eisen})$$

The end symbol $ indicates the word boundary and doubles as a control on the number of modifiers. In general, we will use this process as a *back-off* strategy, i.e., when the trigram *mit der Eisenbahn* is unobserved. Note that this is markedly different from linguistically naïve back-off models that would score the unobserved trigram *mit der Eisenbahn* by falling back on bigram or unigram estimates. In our model, we instead permit the model to back off to this decomposition before dropping valuable context information.

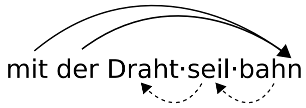


Figure 3.1: Intuition for the proposed generative process of a compound word: The context generates the head component, which generates a modifier component, which in turn generates another modifier. (Literal translation: “with the cable car”)

3.2 An n-gram model with productive compounding

This section defines a hierarchical Bayesian model that embeds the aforementioned intuitions about compound formation into an n-gram language model. The approach is to extend the Pitman-Yor language model (HPYLM; Teh, 2006a) by modifying its base distributions to include a sub-model for generating a compound’s modifiers conditional on its head.

A seemingly obvious alternative method of integration would be to to use two distinct word-level and compound-level n-gram models, independently built using existing heuristic smoothing algorithms. However, the advantage of the unified hierarchical Bayesian model proposed here is that it can learn an interpolation between those levels, obviating the need to introduce and tune an extraneous interpolation scheme between sub-models, while opening the door for future extensions, e.g. analysing compounds occurring in the n-gram history.

3.2.1 Hierarchical Pitman-Yor Language Model (HPYLM)

The HPYLM applies the technique of iteratively shortening the context of an n-gram to affect smoothing over an n-gram conditional distribution, as used by conventional n-gram LMs. Let $\mathbf{u}$ denote the context of an n-gram w_1^n, i.e. $\mathbf{u} = [w_1, \ldots, w_{n-1}]$, and define $\pi(\mathbf{u})$ as a function that truncates $\mathbf{u}$ by dropping the left-most context-word.

The HPYLM defines the distribution $G_{\mathbf{u}}$ of a word w given its context $\mathbf{u}$ as being drawn from a Pitman-Yor process with hyperparameters $a_{|\mathbf{u}|}$ and $b_{|\mathbf{u}|}$. The base distribution is taken to be the distribution $G_{\pi(\mathbf{u})}$ governing the probability that the word w follows the truncated context $\pi(\mathbf{u})$, as in the case of back-off models. Since this latter distribution $G_{\pi(\mathbf{u})}$ likewise needs to be modelled, we assume it is itself drawn from another PYP, which has hyperparameters $a_{|\mathbf{u}|-1}$ and $b_{|\mathbf{u}|-1}$. This procedure is applied recursively until reaching the unigram distribution $G_\emptyset$, for which a final PYP prior is used, with its base distribution G_0 being the uniform distribution over the vocabulary, $P(w) = 1/|\mathcal{W}|$.

Conditioned on the context $\mathbf{u}$, a word w is generated as follows:

$$
\begin{aligned}
w &\sim G_{\mathbf{u}} && (3.1)\\
G_{\mathbf{u}} &\sim \mathcal{PY}(a_{|\mathbf{u}|}, b_{|\mathbf{u}|}, G_{\pi(\mathbf{u})}), && (3.2)\\
G_{\pi(\mathbf{u})} &\sim \mathcal{PY}(a_{|\mathbf{u}|-1}, b_{|\mathbf{u}|-1}, G_{\pi\circ\pi(\mathbf{u})}) && (3.3)\\
&\vdots\\
G_\emptyset &\sim \mathcal{PY}(a_0, b_0, G_0) && (3.4)\\
G_0 &= \text{Uniform}(|\mathcal{W}|) && (3.5)
\end{aligned}
$$

3.2.2 HPYLM + compounds (HPYLM+c)

We define a compound word $\tilde{w}$ as a sequence of components $[c_1, \ldots, c_L]$, plus an end symbol \$ marking either the left or the right boundary of the word, depending on the direction of the model. To maintain generality over this choice of direction, let Λ be an index set over the positions, such that c_{Λ_1} always designates the head component.

Following the motivation in §3.1, we introduce an additional level of back-off where the head component c_{Λ_1} is conditioned on the context $\mathbf{u}$ of the word, while the remaining components $c_{\Lambda_2}, \ldots, c_{\Lambda_L}$, as well as an end-of-sequence symbol \$, are generated by a modifier model F, independently of the context $\mathbf{u}$. We use a bigram HPYLM for F.

The two main modifications we make to the word-based HPYLM to obtain the HPYLM+c are as follows:

1. Replace the support of the base distribution at the root of the hierarchy with the reduced vocabulary $\mathcal{M}$, the set of unique elementary components c. $\mathcal{M}$

also includes word types consisting of a single component to begin with.

2. Add an additional level of conditional distributions $H_{\mathbf{u}}$ (with $|\mathbf{u}| = n - 1$) where items from $\mathcal{M}$ combine to form the observed compound words.

The process for generating a word from the HPYLM+c is as follows (see also Figure 3.2b):

$$\textit{Draw head:}\quad c_{\Lambda_1} \sim G_{\mathbf{u}} \tag{3.6}$$

$$G_{\mathbf{u}} \sim \mathcal{PY}(a_{|\mathbf{u}|}, b_{|\mathbf{u}|}, G_{\pi(\mathbf{u})}), \tag{3.7}$$

$$G_{\pi(\mathbf{u})} \sim \mathcal{PY}(a_{|\mathbf{u}|-1}, b_{|\mathbf{u}|-1}, G_{\pi\circ\pi(\mathbf{u})}) \tag{3.8}$$

$$\vdots$$

$$G_{\emptyset} \sim \mathcal{PY}(a_0, b_0, G_0) \tag{3.9}$$

$$G_0 = \text{Uniform}\,(|\mathcal{M}|) \tag{3.10}$$

$$\textit{Draw modifiers:}\quad c_{\Lambda_i} \sim F_{c_{\Lambda_{i-1}}}, \quad 2 \leq i \leq L \quad \text{(i.e. until \$ is drawn)} \tag{3.11}$$

$$F_c \sim \mathcal{PY}(a'_1, b'_1, F_{\emptyset}) \tag{3.12}$$

$$F_{\emptyset} \sim \mathcal{PY}(a'_0, b'_0, F_0) \tag{3.13}$$

$$F_0 = \text{Uniform}\,(|\mathcal{M}|) \tag{3.14}$$

$$\textit{Create word:}\quad \tilde{w} = \text{concatenate}(c_1, \ldots, c_L) \tag{3.15}$$

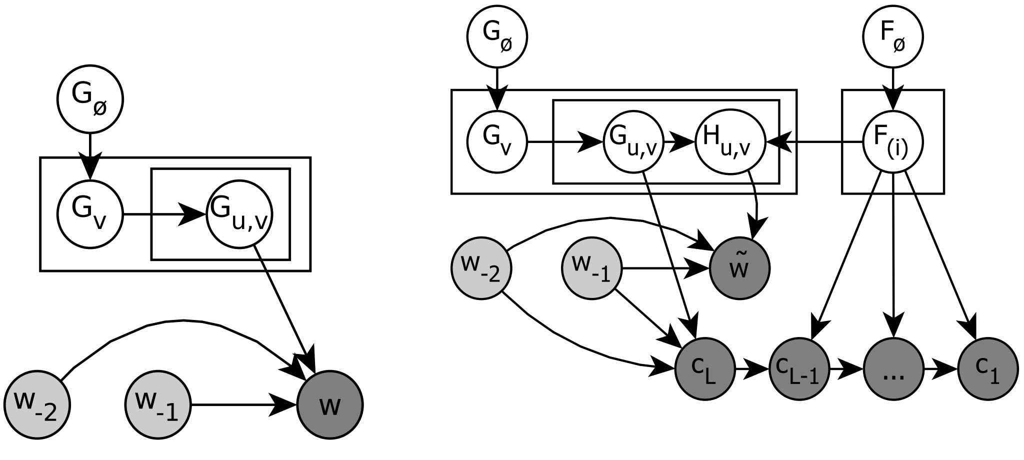
$$\tilde{w} \sim H_{\mathbf{u}} \tag{3.16}$$

$$H_{\mathbf{u}} \sim \mathcal{PY}(a''_{|\mathbf{u}|}, b''_{|\mathbf{u}|}, G_{\mathbf{u}} \times F) \tag{3.17}$$

So the base distribution for the prior of the word n-gram distribution $H_{\mathbf{u}}$ is the product of a distribution $G_{\mathbf{u}}$ over compound heads, given the same context $\mathbf{u}$, and the modifier model F which conditions on the head component.

For German, the linguistically motivated choice for conditioning in F is $\Lambda^{\text{ling}} = [L, L-1, \ldots, 1]$ such that c_{Λ_1} is the true head component; \$ is drawn from $F(\cdot|c_1)$ and marks the left word boundary.

In order to see if the correct linguistic intuition has any bearing on the model's extrinsic performance, we will also consider the reverse, supposing that the left-most component were actually more important in this task, and letting the remaining



(a) Trigram HPYLM generating word w.

(b) Trigram HPYLM+c generating word $\tilde{w}$ according to the conditioning scheme Λ^{ling}, such that c_L is the head and $c_1 \ldots c_{L-1}$ are the modifiers.

Figure 3.2: Plate diagrams illustrating the generative structure for two trigram models HPYLM (left) and HPYLM+c (right). Variables w_{-2} and w_{-1} denote the two words in the trigram context. Hyperparameters and their priors are omitted for clarity.

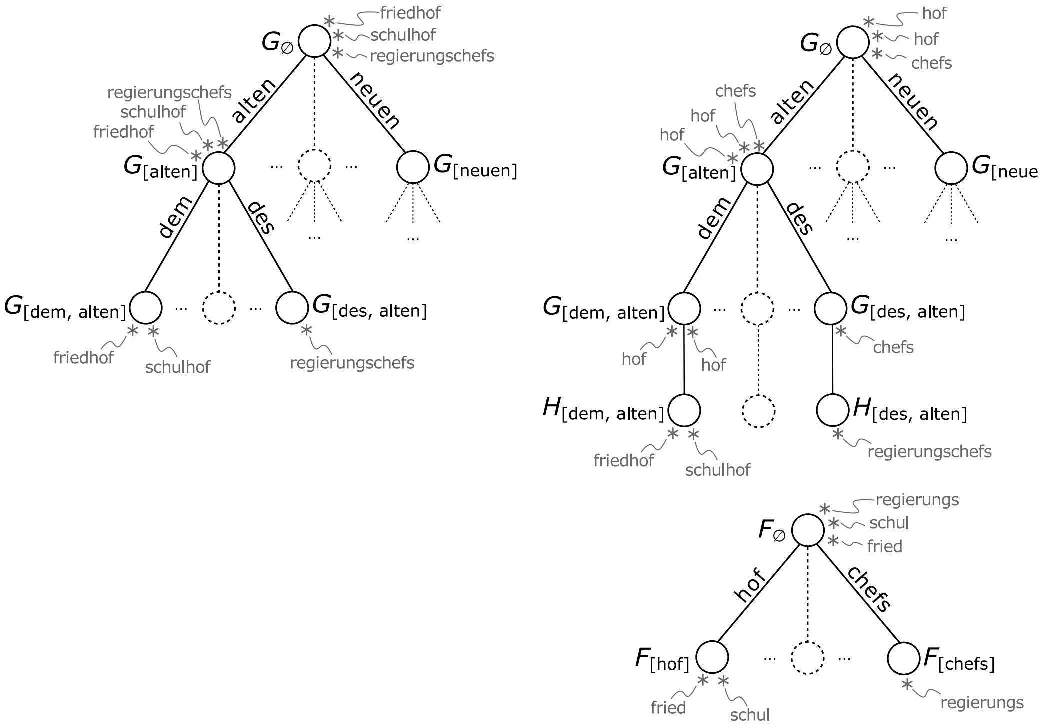
components be generated left-to-right. This is expressed by $\Lambda^{\text{inv}} = [1, \ldots, L]$, where \$ this time marks the right word boundary and is drawn from $F(\cdot|c_L)$.

3.2.3 Example

This section provides an example of how the models are instantiated. We illustrate the trigram case and focus on the structures related to the following fragment of a hypothetical corpus:

Observed n-grams	**Segmentation**
dem alten schulhof (‘the old schoolyard’)	schul·hof
dem alten friedhof (‘the old cemetery’)	fried·hof
des alten regierungschefs (‘the old head of government’)	regierungs·chefs
…	

The hierarchies induced by these n-grams in both models can be visualised as trees, as shown in Figure 3.3. The key thing to notice is how the HPYLM+c combines the statistics of distinct compounds (*schulhof* and *friedhof*) sharing a head, as indicated by the presence of two *hof*-customers in the restaurants of both $G_{\text{[dem, alten]}}$ and $G_{\emptyset}$, unlike in the HPYLM.



(a) HPYLM (b) HPYLM+c

Figure 3.3: This visualisation illustrates the tree structures formed by the hierarchical priors in the HPYLM and HPYLM+c models in representing the corpus fragment given in §3.2.3. Grey elements represent some of the customers seated within the various restaurants and each asterisk denotes a single customer. Only the minimal number of customers relevant to the example are shown explicitly.

Under the HPYLM, the probability of the word *schulhof* given the context *dem alten*, conditioned on a particular seating arrangement across the hierarchy, is computed recursively using Equation 2.15 (p. 26), as follows:

$$P_{\text{HPYLM}}(\textit{schulhof} \mid \textit{dem alten}) \tag{3.18}$$

$$= G_{[\text{dem, alten}]}(\textit{schulhof}) \tag{3.19}$$

$$= \frac{N^{(\text{d.a.})}_{\text{schulhof}} - a_2 m^{(\text{d.a.})}_{\text{schulhof}}}{n^{(\text{d.a.})} + b_2} + \frac{\left(a_2 m^{(\text{d.a.})} + b_2\right) G_{[\text{alten}]}(\textit{schulhof})}{n^{(\text{d.a.})} + b_2} \tag{3.20}$$

with

$$G_{[\text{alten}]}(\textit{schulhof}) = \frac{N^{(\text{a.})}_{\text{schulhof}} - a_1 m^{(\text{a.})}_{\text{schulhof}}}{n^{(\text{a.})} + b_1} + \frac{\left(a_1 m^{(\text{a.})} + b_1\right) G_\emptyset(\textit{schulhof})}{n^{(\text{a.})} + b_1} \tag{3.21}$$

$$G_\emptyset(\textit{schulhof}) = \frac{N^{(\emptyset)}_{\text{schulhof}} - a_0 m^{(\emptyset)}_{\text{schulhof}}}{n^{(\emptyset)} + b_0} + \frac{\left(a_0 m^{(\emptyset)} + b_0\right) G_0(\textit{schulhof})}{n^{(\emptyset)} + b_0}, \tag{3.22}$$

where $G_0(\textit{schulhof}) = 1/|\mathcal{W}|$. The notation here uses superscripts to identify the associated restaurant in the hierarchy. For instance, $N^{(\text{d.a.})}_{\text{schulhof}}$ is the number of customers seated at tables labelled *schulhof* in the restaurant for the context d.a. $\equiv$ [dem, alten], while $m^{(\text{a.})}_{\text{schulhof}}$ is the number of tables labelled *schulhof* in the restaurant for the context a. $\equiv$ [alten].

Under the HPYLM+c, the same n-gram probability is computed as follows, with the main difference being the product in the base distribution probability:

$$P_{\text{HPYLM+c}}(\textit{schulhof} \mid \textit{dem alten}) \tag{3.23}$$

$$= H_{[\text{dem, alten}]}(\textit{schulhof}) \tag{3.24}$$

$$= \frac{N^{(\text{d.a.})''}_{\text{schulhof}} - a''_2 m^{(\text{d.a.})''}_{\text{schulhof}}}{n^{(\text{d.a.})''} + b''_2} \tag{3.25}$$

$$+ \frac{\left(a''_2 m^{(\text{d.a.})''} + b''_2\right)\left(G_{[\text{dem, alten}]}(\textit{hof}) \times F_{[\text{hof}]}(\textit{schul}) \times F_{[\text{schul}]}(\$)\right)}{n^{(\text{d.a.})''} + b''_2}. \tag{3.26}$$

$G_{[\text{dem, alten}]}(\textit{hof})$ is computed similarly as before, but now has the argument *hof* and will bottom out with a term $G_0(\textit{hof}) = 1/|\mathcal{M}|$. The cascade of computations in the modifier model has a similar form, for instance:

$$F_{[\text{hof}]}(\textit{schul}) = \frac{N^{(\text{h.})'}_{\text{schul}} - a'_1 m^{(\text{h.})'}_{\text{schul}}}{n^{(\text{h.})'} + b'_1} + \frac{\left(a'_1 m^{(\text{h.})'} + b'_1\right) F_{\emptyset}(\textit{schul})}{n^{(\text{h.})'} + b_1} \tag{3.27}$$

$$F_{\emptyset}(\textit{schul}) = \frac{N^{(\emptyset)'}_{\text{schul}} - a'_0 m^{(\emptyset)'}_{\text{schul}}}{n^{(\emptyset)'} + b'_0} + \frac{\left(a'_0 m^{(\emptyset)'} + b'_0\right) F_0(\textit{schul})}{n^{(\emptyset)'} + b'_0}, \tag{3.28}$$

where h. $\equiv$ [hof] and $F_0(\textit{schul}) = 1/|\mathcal{M}|$.

3.3 Inference

For ease of exposition we describe inference with reference to the trigram HPYLM+c model, but the general case should be clear. The model is specified by the latent variables $\mathcal{L} = (G_{[\emptyset]}, G_{[v]}, G_{[u,v]}, H_{[u,v]}, F_{\emptyset}, F_c)$, where $u, v \in \mathcal{W}$, $c \in \mathcal{M}$, and hyperparameters $\Omega = (a_i, b_i, a'_j, b'_j, a''_2, b''_2)$, where $i = 0, 1, 2$; $j = 0, 1$; single primes designate the hyperparameters in F and double primes those of $H_{[u,v]}$. By marginalising out the latent variables in $\mathcal{L}$, we can use a collapsed Gibbs sampler to do inference via the hierarchical variant of the Pitman-Yor Chinese Restaurant Process (PYCRP) (Teh, 2006b).

When the prior of $G_{\mathbf{u}}$ has a base distribution $G_{\pi(\mathbf{u})}$ that is itself PYP-distributed, as in the HPYLM, the restaurant metaphor changes slightly from the description in chapter 2. In general, each context $\mathbf{u}$ has an associated restaurant. Whenever a customer is seated at a previously unoccupied table in some restaurant R which has a PYP-distributed base distribution, a dish must be sampled from that base distribution. That is, a further customer is spawned and joins the parent restaurant pa(R) associated with the context $\pi(\mathbf{u})$. This induces a consistency constraint over the hierarchy: the number of tables m_ℓ labelled with dish ℓ in restaurant R must equal the number of customers N_ℓ at tables serving dish ℓ in the parent restaurant pa(R).

We take care to satisfy this constraint in the HPYLM+c model, where a restaurant associated with the distribution $H_{\mathbf{u}}$ has as base distribution a product of PYP-distributed distributions, F and $G_{\mathbf{u}}$. Here, when a dish ℓ is drawn for a previously unoccupied table in trigram restaurant $H_{[u,v]}$, a customer c_{Λ_1} joins the corresponding trigram restaurant $G_{[u,v]}$, *and* customers $c_{\Lambda_2}, \ldots, c_{\Lambda_L}, \$$ are sequentially sent to the restaurants associated with $F_{c_{\Lambda_1}}, \ldots, F_{c_{\Lambda_L}}$, respectively.

3.3.1 Posterior inference

The Chinese restaurant representations effectively allows us to replace the collection of Pitman-Yor priors with a collection of seating arrangements, S. Marginalising over the seating arrangements S and parameters Ω yields the posterior predictive probability of a word w given the training data $\mathcal{D}$:

$$P(w \mid \mathcal{D}) = \int_{S,\Omega} P(w \mid S, \Omega)\, P(S, \Omega \mid \mathcal{D})\, dS\, d\Omega, \tag{3.29}$$

This integral can be approximated by averaging over a number of posterior samples (S, Ω) generated using Markov chain Monte Carlo methods. In particular, we can use a Gibbs sampler that explores the space of possible seating assignments by iteratively resampling a single model variable conditioned on all other model variables. Given a current seating arrangement S, a new configuration S' is sampled by individually reseating each customer x_i in the trigram restaurants $H_{[u,v]}$. Reseating a customer x_i means removing them from their current table z_i and assigning another table k' drawn from the conditional distribution $P(z_i \mid \mathbf{z}^{-i})$. Here, $\mathbf{z}^{-i}$ denotes the table allocations $(z_1, \dots, z_{i-1}, z_{i+1}, \dots, z_{|\mathbf{z}|})$ of the other customers in the given restaurant.

If the original table z_i only seated a single customer, the removal of that customer triggers further customer removals from the base distribution restaurants in accordance with the hierarchy consistency constraints.

In the absence of any strong intuitions about appropriate values for the hyperparameters, we place vague priors over them and use slice sampling (Neal, 2003) to update their values during generation of the posterior samples: $a \sim \text{Beta}(1, 1)$ and $b \sim \text{Gamma}(10, 0.1)$.[1]

Lastly, we follow the pragmatic course of using a single sample (S, Ω) to approximate the integral over the posterior.[2]

3.3.2 Sampler description

The previous section outlines the Gibbs sampler in broad terms. In this section, we detail the algorithms involved.

---

[1] We used the slice sampler implementation released by Mark Johnson, `http://web.science.mq.edu.au/~mjohnson/Software.htm`

[2] Preliminary experiments (C. Dyer, p.c.) indicated that the posterior over the latent model structure is quite sharply peaked, so that a single sample constitutes a low-variance estimator of the posterior predictive distribution.

The scheme described above poses the challenge of keeping track of a table assignment for each individual customer in every restaurant. Given that language models need to be trained on substantial amounts of data, this bookkeeping can be problematic, both in terms of memory and implementation.

Blunsom et al. (2009) introduced an efficient solution to this problem, which leverages the exchangeability property of the hierarchical Dirichlet process and which carries over straightforwardly to the PYP. Instead of tracking what table each customer is seated at, the authors suggest maintaining histograms of table occupancy. When a customer has to be added or removed from a restaurant, it is done by sampling directly from the histogram. A histogram Hist_w maps a table occupancy value t to the number of tables labelled with w and having t customers, $\text{Hist}_w[t]$.

As a brief example, suppose we have an assignment of 6 customers to tables as z_1=1, z_2=1, z_3=2, z_4=1, z_5=3, z_6=4 and the tables are labelled as ℓ_1=ℓ_3=*sun*, ℓ_2=ℓ_4=*moon*. The equivalent histogram representation is $\text{Hist}_{\textit{sun}}[3]$=1, $\text{Hist}_{\textit{sun}}[1]$=1 and $\text{Hist}_{\textit{moon}}[1]$=2.

The two basic operations required for updating the seating arrangement in the HPYLM+c are ADDWORDTOMODEL and REMOVEWORDFROMMODEL, as defined in Algorithms 1–2. To facilitate reading the algorithms in conjunction with the mathematical definitions given before, the psuedocode retains the quantities N_w, m_w, and m, but we note that these can all be reconstructed from the histogram representations as follows:

$$N_w = \sum_{t \in \text{Hist}_w} t \times \text{Hist}_w[t] \tag{3.30}$$

$$m_w = \sum_{t \in \text{Hist}_w} \text{Hist}_w[t] \tag{3.31}$$

$$\text{and } m = \sum_{w} m_w, \text{ as stated earlier.} \tag{3.32}$$

Algorithm 1 Adding an n-gram $(\mathbf{u}, w)$ to the HPYLM+c.

```
function AddWordToModel($w$, $\mathbf{u}$)
    $\mathbf{c} \leftarrow$ SegmentCompound($w$)    ▷ for example, as detailled in §3.4.2.
    $P_0^w \leftarrow G_{\mathbf{u}}(c_{\Lambda_1}) \times \prod_{i=2}^{L} F_{c_{\Lambda_{i-1}}}(c_{\Lambda_i}) \times F_{c_{\Lambda_L}}(\$)$    ▷ base probability, cf. Equation 3.17
    if AddCustomerToRestaurant($w$, $P_0^w$, $H_{\mathbf{u}}$, $a''_{|\mathbf{u}|}$, $b''_{|\mathbf{u}|}$) then
        RecursiveAdd($c_{\Lambda_1}$, $\mathbf{u}$, $G_*$, $\mathbf{a}$, $\mathbf{b}$)
        for $i \leftarrow 2, L$ do
            RecursiveAdd($c_{\Lambda_i}$, $c_{\Lambda_{i-1}}$, $F_*$ ,$\mathbf{a}'$, $\mathbf{b}'$)
        RecursiveAdd($\$$, $c_{\Lambda_L}$, $F_*$, $\mathbf{a}'$, $\mathbf{b}'$)
end function

function AddCustomerToRestaurant($w$, $P_0^w$, $X$, $a$, $b$)
    ▷ Hist$_w$ is the table occupancy histogram for customer type $w$ for the restaurant $X$.
    isNewTable ← false
    $p_{share} = \max(N_w - a \times m_w, 0)$    ▷ based on ...
    $p_{new} = (a \times m + b) \times P_0^w$    ▷ ... Equation 2.10&2.15
    $r \leftarrow random(0, p_{share} + p_{new})$
    if $r \leq p_{new}$ then
        Hist$_w[1] \leftarrow$ Hist$_w[1] + 1$
        isNewTable ← true
    else
        $r \leftarrow random(0, p_{share})$
        for all $t \in$ Hist$_w$ do    ▷ sample a table occupancy from histogram
            $r \leftarrow r - ($Hist$_w[t] \times (t - a))$
            if $r \leq 0$ then    ▷ add to a table currently containing $t$ customers
                Hist$_w[t+1] \leftarrow$ Hist$_w[t+1] + 1$
                Hist$_w[t] \leftarrow$ Hist$_w[t] - 1$
                break
    $N_w \leftarrow N_w + 1$
    return isNewTable
end function

function RecursiveAdd($w$, $\mathbf{u}$, $X_*$, $\mathbf{a}$, $\mathbf{b}$)
    if $\mathbf{u} = \emptyset$ then return
    $P_w^0 \leftarrow X_{\pi(\mathbf{u})}(w)$
    if AddCustomerToRestaurant($w$, $P_0^w$, $X_{\mathbf{u}}$, $\mathbf{a}_{(|\mathbf{u}|)}$, $\mathbf{b}_{(|\mathbf{u}|)}$) then
        RecursiveAdd($w$, $\pi(\mathbf{u})$, $X_*$, $\mathbf{a}$, $\mathbf{b}$)
end function
```

Algorithm 2 Removing an n-gram $(\mathbf{u}, w)$ from the HPYLM+c.

```
function RemoveWordFromModel($w$, $\mathbf{u}$)
    $\mathbf{c} \leftarrow$ segment($w$)
    if RemoveCustomerFromRestaurant($w$, $H_{\mathbf{u}}$) then
        RecursiveRemove($c_{\Lambda_1}$, $\mathbf{u}$, $G_*$)
        for $i \leftarrow 2, L$ do
            RecursiveRemove($c_{\Lambda_i}$, $c_{\Lambda_{i-1}}$, $F_*$)
        RecursiveRemove($, $c_{\Lambda_L}$, $F_*$)
end function

function RemoveCustomerFromRestaurant($w$, $X$)
    ▷ $Hist_w$ is the table occupancy histogram for customer type $w$ for the restaurant $X$.
    isTableEmptied ← false
    $r \leftarrow random(0, N_w)$
    for all $t \in$ Hist$_w$ do    ▷ sample a table occupancy from the histogram
        $r \leftarrow r -$ Hist$_w[t] \times t$
        if $r \leq 0$ then    ▷ remove from a table currently containing $t$ customers
            Hist$_w[t] \leftarrow$ Hist$_w[t] - 1$
            if $t > 1$ then
                Hist$_w[t-1] \leftarrow$ Hist$_w[t-1] + 1$
            else
                isTableEmptied ← true
            break
    $N_w \leftarrow N_w - 1$
    return isTableEmptied
end function

function RecursiveRemove($w$, $\mathbf{u}$, $X_*$)
    if $\mathbf{u} = \emptyset$ then return
    if RemoveCustomerFromRestaurant($w$, $X_{\mathbf{u}}$) then
        RecursiveRemove($w$, $\pi(\mathbf{u})$, $X_*$)
end function
```

	En	De	De LM
Sentences	1.7m	1.7m	2.4m
Word Tokens	49m	38m	59m
Word Types	112k	351k	

(a) Statistics of training corpora.

Parts/Compound	# Compound Types
2	197233
3	25128
4	1194
≥5	59
Total number:	223614

(b) Compound types by length.

Table 3.1: Summary of training data and compound segmentation. The combined reading is that 223614 of the 351k word types are designated as compounds.

3.4 Experiments

We evaluate the proposed compound model HPYLM+c on German in terms of its intrinsic performance (§3.4.4), and by its influence when used as part of a machine translation system (§3.4.5). In both settings, we compare against baselines obtained with the standard HPYLM as well as the Modified Kneser-Ney (MKN) LM.

The next two sections describe further methodology details.

3.4.1 Data preparation

We use corpus data released as part of the 2011 ACL Workshop on Machine Translation's shared task[3]. Standard data preprocessing steps included normalising punctuation, tokenising and lowercasing all words.

For language model training, we used the combination of the corpora Europarl, news-commentary and news-2011, filtered to exclude duplicate sentences. This amounts to 59 million word tokens.

The language model vocabulary $\mathcal{W}$ consists of all word types appearing in the target-side of the bitext. As the machine translation system used in the subsequent extrinsic evaluation cannot output word types outside of this vocabulary, the choice was made not to model anything beyond that.[4] LM training tokens outside of that

---

[3] `http://www.statmt.org/wmt11/`

[4] See §3.4.5.1 for details on bitext preprocessing.

vocabulary were replaced by the UNK symbol. Models were trained using all resulting 4-grams; i.e. no pruning was done on low-frequency n-grams.

Test data for language model evaluation comprised the combination of the newstest corpora of 2008–2010, released with WMT11. Statistics are provided in Table 3.1a.

3.4.2 Compound segmentation method

For this evaluation, we used a deterministic, *a priori* segmentation of compound words into their parts. This means we assume a single, fixed analysis of a compound regardless of the context it occurs in, which is necessitated by the fact that our probabilistic model does not specify a step for choosing an analysis. To construct a segmentation dictionary, we used the one-best segmentation output of a supervised maximum-entropy-based compound splitter (Dyer, 2009) on all the words in the training vocabulary. In addition, word-internal hyphens were also taken as segmentation points.[5] We regard the set of compounds $\mathcal{C} \subset \mathcal{W}$ as those word types that are split into two or more parts by this procedure. Similarly, $\mathcal{M}_{\mathcal{C}} \subset \mathcal{M}$ is the set of unique word parts that result from segmenting the compounds $\mathcal{C}$. The majority of compounds thus identified consist of two or three parts (Table 3.1b).

In our method, linking elements do not constitute items in the vocabulary of word components $\mathcal{M}$. Regarding linking elements as components in their own right would sacrifice important contextual information and disrupt the conditional bigram distributions, $F(\cdot|c_{\Lambda_{i-1}})$, generating compound modifiers. That is, faced with the compound küche·<u>n</u>·tisch (‘kitchen table’), we want the generation of the modifier to be according to $P(\text{küche}|\text{tisch})$, and not $P(\text{küche}|\text{n})$. To account for linking elements we follow the pragmatic option of merging any linking element onto its adjacent component—for Λ^{ling}, merging happens onto the preceding component (i.e. using $P(\text{küchen}|\text{tisch})$), while for Λ^{inv}, the linking element is merged onto the succeeding

---

[5] Our choice of a supervised compound splitter is based on a desire to focus the evaluation on the language model given high quality segmentations. The splitter of Dyer (2009) obtained a segmentation F-score of ~95% in their evaluation. Unsupervised methods could also be used with our model.

Linking element handling	$\lvert\mathcal{M}_{\mathcal{C}}\rvert$	$\lvert\mathcal{M}\rvert$	**Segmented example**
(no segmentation, i.e. $\mathcal{M} = \mathcal{W}$)	223614	350997	Küchentisch
delete	45183	144358	Küche·tisch
merge left (Λ^{ling})	50868	152323	Küchen·tisch
merge right (Λ^{inv})	63315	164095	Küche·ntisch

Table 3.2: Effect of linking elements on the number of unique word components modelled. The second column is the number of components measured only over the set of compounds $\mathcal{C}$, while the third column is over all words $\mathcal{W}$, showing what the size of $\mathcal{M}$, the support of the base distribution $G_\emptyset$, would be under the alternative schemes for handling linking elements.

component (i.e. using $P(\text{ntisch}|\text{küche})$). This ensures that the set of head components $\{c_{\Lambda_1}\}$ is not fragmented.[6] Table 3.2 summarises the effect of these different merging options on the size of the vocabulary $\mathcal{M}$ of word components modelled.

3.4.3 MCMC sampler configuration

We initialised the samplers used for the HPYLM and HPYLM+c with random seating assignments instead of basing the initial configuration on the actual conditional distributions for adding customers to restaurants. This worked satisfactorily and was convenient in our implementation, so we did not experiment with alternative initialisation schemes.

Each sampler was then run for 300 iterations of burn-in, and we gauged convergence toward the underlying stationary distribution at the hand of the posterior likelihood (Figure 3.4). This approach takes a cue from earlier work on the HPYLM on non-trivial amounts of training data, where a similarly low number of iterations was used (Huang and Renals, 2010). One consideration for why this works reasonably well—in light of the favourable comparisons of the HPYLMs to traditional back-off LMs—is that the model does not need to infer complex latent structure, as may be the case in other MCMC applications where vastly more iterations are standard practice. The identities of word tokens are fully observed, so that the sampler

---

[6] It is worth noting that for German the presence and identity of linking elements between consecutive components c_i and c_{i+1} are largely governed by the first component c_i (Goldsmith and Reutter, 1998). A more sophisticated model could incorporate this bias.

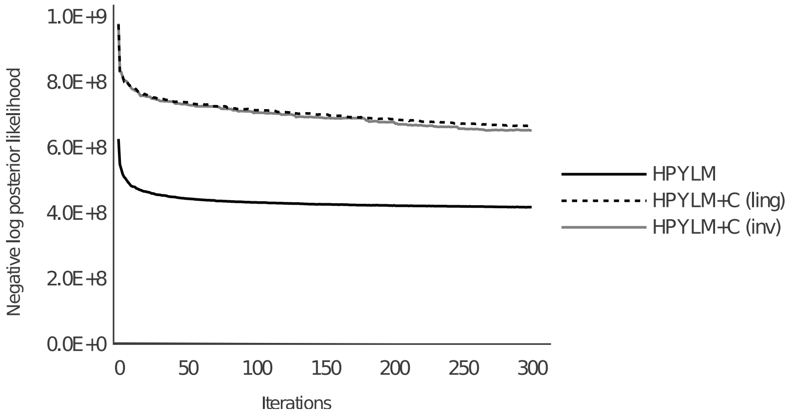


Figure 3.4: Sampler burn-in for the 4-gram models.

primarily has to explore different seating arrangements across the hierarchy of Chinese restaurant processes. Moreover, the training data tends to be sparse, so that the range of possible configurations for the restaurants corresponding to rare n-grams are limited by a paucity of customers.

3.4.4 Intrinsic evaluation

The modelling approach proposed in this chapter is premised on the hypothesis that better smoothing of n-gram distributions can be obtained by analysing compounds into their components. This intrinsic evaluation aims to test that hypothesis and to explore the relative performance of the model variants that make different assumptions around the conditional dependencies of compound parts.

The HPYLM+c models were compared against two baseline methods, one using MKN smoothing and the other being the HPYLM, both without any special handling of compounds.[7]

3.4.4.1 Results

Test-set perplexities are summarised in Table 3.3. We find that the HPYLM obtains a perplexity of 294.0 on our test-set, lower than the MKN perplexity of 299.9. This

---

[7] The SRILM toolkit was used for training and testing the MKN models (Stolcke, 2002).

	PPL	vs. MKN	vs. HPYLM
MKN	299.9		
HPYLM	294.0	-2.0%	
HPYLM+c ling	294.1	-1.9%	$+0.0\%$
HPYLM+c inv	305.5	$+1.9\%$	$+3.9\%$

Table 3.3: Test-set perplexity results for two variants of the proposed compounding language model and two baseline language models. The last two columns give the percentage perplexity reduction obtained by the model of the given row relative to each baseline.

outcome is consistent with earlier results that the HPYLM outperforms the MKN model for training data sets of non-trivial sizes (Huang and Renals, 2009).

The HPYLM+c that assumes right-headed compounds (Λ^{ling}) virtually matches the performance of the HPYLM baseline, a result that we consider in more detail shortly. The assumption of left-headed compounds (Λ^{inv}), which is *inconsistent* with the known properties of German, leads to a marked degradation in perplexity. The HPYLM+c with this assumption achieves perplexity of 305.5, which is higher than the perplexity of both baseline models. This result confirms the validity of our strategy to condition actual compound heads on sentential context.

The aforementioned comparison features a bias toward the word-based baseline models, since the perplexity of different models on a given data set is directly comparable only if the models are defined over the same vocabulary. This is not the case here—according to the generative process of the HPYLM+c (§3.2.2), the generation of an additional modifier component $c \in \mathcal{M}$ for any partially formed token $\tilde{w}$ always carries non-zero probability. HPYLM+c thereby defines a distribution over a countably infinite set of word types, compared to the finite vocabularies modelled by the word-based HPYLM and MKN. The preceding results therefore establish a type of upper bound on the perplexities of the HPYLM+c models on the test-set.

To obtain a fairer comparison, we performed the costly renormalisation of the right-headed HPYLM+c over the finite vocabulary of words modelled by the baseline LMs. We find that the test-set perplexity of the renormalised model is 292.2, which is below the 294.0 of the HPYLM baseline. The fact that the difference in perplexity between the infinite model and its renormalisation is not too great suggests that the HPYLM+c reserves a reasonable amount of probability mass for words outside the finite vocabulary; it would not be ideal if too large a portion of mass was allocated beyond that vocabulary, since “most” items in the countably infinite domain of the model would not constitute well-formed compounds that could practically occur in the language.

3.4.4.2 Effect of Markov order

The baseline word-based language models deal with unseen test n-grams by progressively truncating the context to rely on lower-order distributions for which the training data is less sparse. The simplifying assumption encoded by this strategy is that novelty in n-grams arises strictly from the high likelihood of encountering a new element from the space of $|\mathcal{W}|^n$ possible n-grams.

In the HPYLM+c, the back-off procedure for handling unseen n-grams is first to decompose the predicted compound into its components, and thereafter to truncate context. The underlying assumption is therefore that compounding is the primary driver of novelty in n-grams, and that the space of possible n-grams should be viewed as $|\mathcal{W}|^{n-1} \times |\mathcal{M}|^L$, where the number of components in a compound, L, is in principle unbounded but in practice likely to be bounded at the order of 10.

The aforementioned assumption in the HPYLM+c is violated when compounding is *not* responsible for an unseen n-gram. The model would then prematurely decompose the predicted word when in fact immediately truncating the context without decomposing the predicted word would result in matching some observed k-gram ($1 \leq k \leq n$).

The impact of this effect is measured in a further experiment by training language models with lower maximum orders, $n = 2$ and $n = 3$. At lower n, there should be fewer test tokens for which premature decomposition could occur, in principle. The results in Table 3.4 show that the relative margin of improvement by HPYLM+c over HPYLM increases as n is decreased from 4 to 2. This outcome implies that priori-

	n=2	n=3	n=4
MKN	394.5	307.2	299.9
HPYLM	396.6	303.3	294.0
HPYLM+c	390.0	299.3	294.1
HPYLM relative to MKN	$+0.5\%$	-2.6%	-1.9%
HPYLM+c relative to HPYLM	-1.6%	-1.3%	$+0.0\%$

Table 3.4: Test-set perplexity for models trained with different n-gram orders, along with relative differences at the bottom. HPYLM+c refers to the variant using Λ^{ling}.

tising generalisation through compound decomposition higher than generalisation through context truncation tends to constrain the model's performance.

Convergence of lower-order models For $n = 2$ and $n = 3$, the sampler had not fully converged after 300 iterations. We suspect this is due to the higher entropy in the distributions governing the seating assignments: If $n = 2$, there should be more customers (hence more possible seating configurations) in the average restaurant for a particular context-length $j < n$ than in the same restaurant if n is larger. This did not affect perplexity, which was stable when evaluating with different individual samples from the posterior around 300 iterations.

3.4.4.3 Breakdown by different test subsets

A more fine-grained view of model performance can be obtained by partitioning test tokens into meaningful subsets and computing the perplexity over the test tokens in each subset. In this section, the subsets are characterised by the type of token (compounds vs. non-compound) and the "hit-length" h, where $h = 1$ means the token appears in an unseen context and the language model has to rely on the unigram distribution to score it, through to $h = n$, for which the full n-gram occurred in training data. Note that h is defined from the view-point of a standard n-gram model, using surface-level matching, and does not include decomposition.

The results of this breakdown are reported in terms of relative cross-entropy reductions in Figure 3.5 and the sizes of the different subsets are summarised in

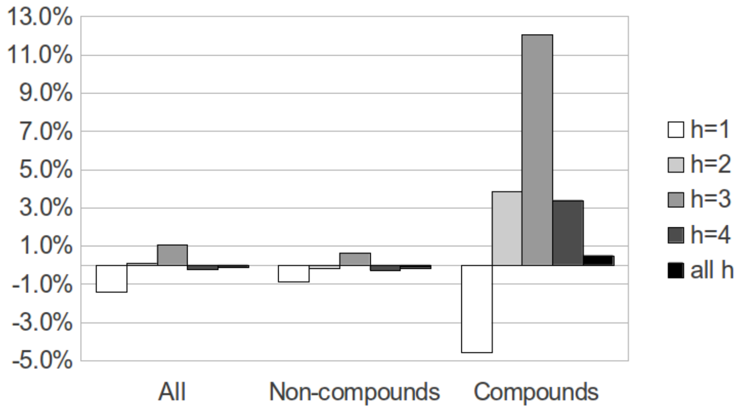


Figure 3.5: Percentage relative cross-entropy reduction of HPYLM+c with respect to the baseline HPYLM, broken down by test token hit-length h and compound status (defined in §3.4.4.3). Both are 4-gram models. Bars going down imply improvement over the baseline.

Figure 3.6.[8]

The breakdown indicates that HPYLM+c improves over the baseline HPYLM only for test tokens appearing in fully unseen contexts ($h = 1$); this improvement is larger for compounds (-4.6$\%$) than for non-compounds (-0.9$\%$). This improvement for non-compounds with $h = 1$ is consistent with the fact that many compound heads appear as words in themselves; their prediction thus benefits from the way the HPYLM+c shares statistical strength across instantiations of a given head.

The categories where the HPYLM+c performs worse than the baseline are for trigram hits generally, and for compounds at hit-lengths $h \neq 1$.

3.4.4.4 Qualitative analysis

For a more qualitative insight into the model performance, we did a further direct comparison of the HPYLM+c model and the MKN baseline by ranking test set compounds by the difference in probability value that each model assigns to the n-gram. The test compounds where the compound model does best (Table 3.5 top) are all words for which an analysis into a context-dependent head and modifiers should clearly be beneficial. For example, in scoring the phrases *wochen vor den*

---

[8] This analysis uses cross-entropy rather than perplexity because the dynamic range of perplexity varies drastically across the test subsets, which impedes interpretation when presenting results on a single visual scale.

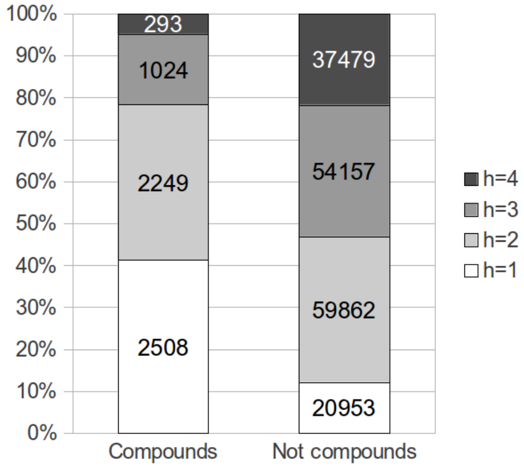


Figure 3.6: Composition of test-set in terms of number of tokens with hit-length h and compound status. The total 178525 test tokens breaks down into 6074 compounds and 172451 non-compounds, the latter including the end-of-sentence padding and OOVs, which are handled with mapping to an UNK symbol.

präsidentschaftswahlen (‘weeks before the presidential elections’) and *tage vor den parlamentswahlen* (‘days before the parliamentary elections’), the head *wahlen* is having a mutually reinforcing effect.

In contrast, we find that the cases where the MKN baseline model does best (Table 3.5 bottom) feature various words that are not strictly speaking compounds, and feature over-segmentation by the supervised compound segmentor, e.g. *ging+rich* and *wissen+schaften* (‘sciences’), or through the greedy splitting on hyphens, e.g. *ki-+moon*. These are words where our compound model’s smoothing is hurting performance, since it allocates some probability mass toward observing other modifiers with these heads, which in the case of these proper nouns will not happen. This is evidence of success on the part of our model’s underlying mechanism, but demonstrates that more care should be taken with the particular segmentation method used.

3.4.4.5 Effect of training data size

We performed a scaling experiment to investigate the model’s sensitivity to training data size. The vocabulary at each training data size is determined directly by the data (excluding singletons in order to learn reasonable conditional distributions involving

HPYLM+c better	Δ
gegen die umstrittene **wieder+wahl**	0.058
aufbau der afghanischen **sicherheits+kräfte**	0.036
dessen zentralen **gesichts+punkten**	0.035
in annapolis , **mary+land**	0.035
wochen vor den **präsidentschafts+wahlen**	0.032
dieses vertrauen nicht **miss+brauchen**	0.030
für psychiatrie und **psycho+therapie**	0.028
tage vor den **parlaments+wahlen**	0.028
reduktion der **treibhausgas+emissionen**	0.025
in einem unblutigen **militär+putsch**	0.021
Baseline (MKN) better	Δ
, newt **ging+rich**	0.511
nächtlichem **flug+lärm**	0.449
generalsekretär ban **ki-+moon**	0.423
in st. **peters+burg**	0.420
im 17. **jahr+hundert**	0.419
saalpublikums in st. **peters+burg**	0.359
militanten klerikers moqtada **al-+sadr**	0.352
un-hochkommissarin für **menschen+rechte**	0.286
schwebt in **lebens+gefahr**	0.231
der akademie der **wissen+schaften**	0.212

Table 3.5: Compounds from the monolingual test set for which HPYLM+c outperforms MKN by the largest margin (top) and *vice versa* (bottom). We define the margin Δ as the difference in probability that the models assign to the given test n-gram.

the unknown word symbol). Perplexity is therefore not directly comparable from one size to another, hence we report relative perplexity differences in Figure 3.7.

The results show that HPYLM+c provides a small but consistent improvement over the HPYLM baseline as training data size is varied over two orders of magnitude. The two Bayesian models outperform the MKN baseline by an increasing relative margin as data size grows, and the additional benefit of HPYLM+c over HPYLM is additive. These trends are consistent with our intuition that productive compounding affects language model performance not just at the smallest of data sizes; extrapolation suggests consistent benefit could be derived from accounting for compounding could at larger data sizes as well.

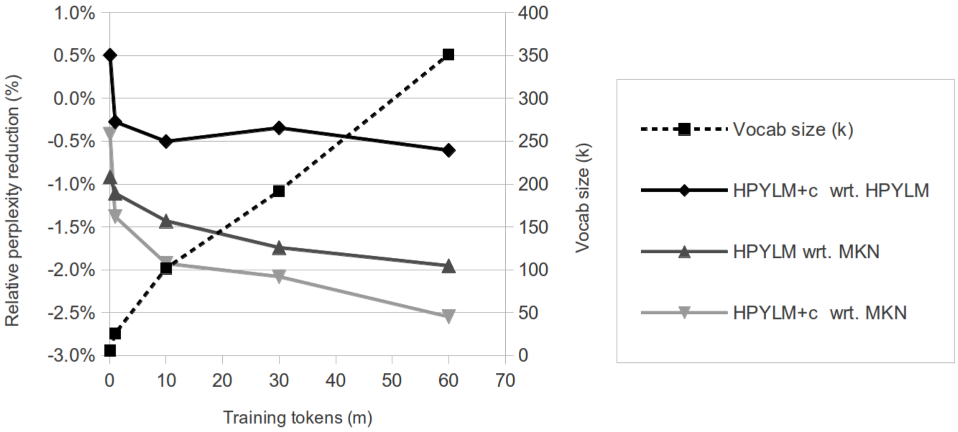


Figure 3.7: Relative perplexity differences of HPYLM+c against the two baseline models as the number of training tokens is varied from 100k to 60m tokens. 4-gram models are used throughout.

3.4.5 Extrinsic machine translation evaluation

Our intrinsic evaluation established that the HPYLM+c obtains small improvements over the MKN and HPYLM baseline models. This section reports on a machine translation experiment, where we compare the quality of the output produced by the translation system when using different language models.

We measure translation quality with the automated metric BLEU (Papineni et al., 2001).

3.4.5.1 System details

This evaluation uses the hierarchical phrase-based translation system implemented by the *cdec* decoder (Dyer et al., 2010).

The English-German bitext consisted of Europarl and news commentary, filtered after preprocessing to exclude sentences longer than 50 tokens, resulting in 1.7 million parallel sentences; word alignments were inferred from this using the Berkeley Aligner (Liang et al., 2006) and used as a basis from which to extract a synchronous context-free grammar (SCFG; Chiang, 2007).

The weights of the log-linear translation model were optimised towards the BLEU metric using *cdec*'s implementation of Minimum Error-rate Training (MERT; Och, 2003). The development set news-test2008 (2051 sentences) was used for this

	PPL	**BLEU**	std.dev
MKN	299.9	13.9	±0.3
HPYLM	294.0	13.9	±0.3
HPYLM+c (Λ^{ling})	294.1	13.9	±0.1
HPYLM+c (Λ^{inv})	305.5	13.7	±0.2

Table 3.6: Machine translation results obtained with different language models. The last two columns give BLEU mean and standard deviation over three runs. For reference, we also show the perplexity of each (4-gram) language model on the monolingual test set.

weight tuning. The evaluation result is BLEU measured on the test set newstest2011 (3003 sentences, 171460 tokens). To counter the known instability of MERT+BLEU (Clark et al., 2011), three repetitions were done and the results averaged.

3.4.5.2 Language model feature function

The independent variable in this translation experiment is the language model. The two baseline systems respectively use the 4-gram MKN and HPYLM baseline language models from the preceding monolingual evaluation as a feature function in the log-linear translation model. To test the effect of the compounding LM on translation, this feature function is replaced by the HPYLM+c variant in question. Each translation system therefore uses a single language model feature.

3.4.5.3 Translation results

In terms of BLEU score, we do not find a meaningful difference between the various systems (Table 3.6). The system using HPYLM+c matches the two baselines, a result that indicates our more expressive modelling is not sacrificing any performance in this task. This is an important outcome, as it means we avoid a common pitfall whereby a new model is proposed to target some specific phenomenon only to sacrifice performance globally. Consistent with the intrinsic evaluation, the use of the left-headed model leads to a small decrease in BLEU.

	P	R	F
MKN	$25.5{\pm}1.5$	$17.2{\pm}1.0$	$20.5{\pm}0.8$
HPYLM	$24.5{\pm}2.4$	$17.5{\pm}0.7$	$20.4{\pm}0.4$
HPYLM+c (Λ^{ling})	$27.7{\pm}2.3$	$17.4{\pm}0.7$	$21.3{\pm}0.4$
HPYLM+c (Λ^{inv})	$24.0{\pm}2.6$	$17.2{\pm}0.4$	$20.0{\pm}0.6$

Table 3.7: Precision, Recall and F-score for compounds in the translation output, relative to the reference translations containing 2652 compounds in total. Standard deviations across the three MT repetitions are given next to the measurements.

3.4.5.4 Compound-specific evaluation

Next, we turn to a more fine-grained look at the translation output. The BLEU metric is likely to miss small improvements in translation quality. Moreover, in the reference translations of the test set, only 2652 of the 72661 tokens are compounds; a moderate improvement in generating them is unlikely to have a big impact on the BLEU.

To assess the quality of the translation system in terms of compound words, we use precision, recall and balanced F-score of hypothesised compounds with respect to compounds in the reference sentences. Specifically, let R_i (resp. H_i) be the set of compound words occurring in the i^{th} reference sentence (resp. hypothesis). Precision P and recall R is calculated through micro-averaging as

$$P = \frac{\sum_i |H_i \cap R_i|}{\sum_i |H_i|} \qquad \text{and} \qquad R = \frac{\sum_i |H_i \cap R_i|}{\sum_i |R_i|},$$

while the F-score is the harmonic mean of precision and recall, $F_1 = \frac{2 \times P \times R}{P+R}$. These metrics are computed per individual system run and then averaged across repetitions to obtain the percentage-based results shown in Table 3.7.

The right-headed HPYLM+c obtains an average improvement in compound precision of 9%–13% relative to the two baseline systems (2–3 absolute percentage points). It does so without a decrease in recall, hence the gain in precision is not achieved simply by the system being more conservative about outputting compounds in the first place. The left-headed HPYLM+c performs marginally below the baseline methods in this evaluation as well.

3.4.5.5 Qualitative analysis of translated compounds

The translation output was further analysed to find specific examples of precision improvements attained by the HPYLM+c, compared to the HPYLM baseline. Three examples are given in Table 3.8 on p. 58.

In the Example 1, the baseline system only partially produced the translation of the input phrase *prostate cancer*, while the HPYLM+c system succeeded in producing the full compound.

Example 2 illustrates an improved lexical choice associated with the HPYLM+c system: while *agenda* is a valid translation of the orthographically equivalent English input word, the more idiomatic translation *tagesordnung* is produced by the HPYLM+c system, thereby matching the reference.

Finally, in Example 3 the baseline system incorrectly posits the compound *selbstmordattentate* ('suicide attacks'), while the HPYLM+c system correctly produces the less specific compound.

In the first two examples, the sentential context plays to the strength of the main assumption of the HPYLM+c (i.e. "context generates head"), offering a potential explanation for the improved output. However, in the final example, this is not the case; the compound's context in the hypotheses is not particularly informative, and the improvement has to be put down to a more complex discriminating role fulfilled by the language model during decoding.

3.5 Related work

Another common approach for addressing the sparsity effects of productive compounding (Koehn and Knight, 2003; Koehn et al., 2008; Stymne, 2009; Berton et al., 1996), and rich morphology (Habash and Sadat, 2006; Geutner, 1995), has been to use pre/post-processing with an otherwise unmodified translation system or speech recognition system. This approach views the existing machinery as adequate and shifts the focus to finding a more appropriate segmentation of words into tokens, i.e. compounds into parts or words into morphemes, thus achieving a vocabulary reduction. The downside of such a method is that training a standard n-gram language model on pre-segmented data introduces unwanted effects: in the case of German compounds, the split-off modifiers would take precedence in the head's n-gram context, and during back-off the actual word-context information is discarded first. As illustrated earlier in the "forecasters predicted" example in §2.2.3, the problem is similar when modelling sequences of morphemes as n-grams, and earlier work in speech recognition has shown that taking steps against this effect can improve recognition accuracy (Ircing et al., 2001).

Pre-processing also often requires heuristics to guard against over/under-segmentation, which do not generalise well to different settings or languages. The modelling approach introduced in this chapter is also subject to the whims of our compound segmentation method, but the model is more robust since it retains the original surface form of the word—recall that the decomposition step amounts to interpolated back-off.

Baroni and Matiasek (2002) proposed basic models of German compounds for use in predictive text input, exploiting the same link between right-headedness and context as we have, although their focus was restricted to compounds with two components.

Considering language modelling with PYPs more generally, two pieces of work in particular warrant discussion. Wood and Teh (2009) generalised hierarchical PYPs beyond tree hierarchies to directed acyclic graph structures. They modelled domain adaptation by tying together multiple domain-specific HPYLMs and a latent "background" HPYLM. The intuition for modelling a given n-gram is that a domain-specific model is able to back-off either by shortening the context or by switching to the background model while retaining the same context. Importantly, this choice is available at all context lengths and is executed by using a base distribution that is a mixture of PYPs. In contrast, our HPYLM+c ties together a head-generating HPYLM (G_*) and a modifier-generating HPYLM (F_*) only at the level of full contexts, and does so via a base distribution (of H) that is product instead of a mixture. It could therefore be interesting to apply the graphical LM of Wood and Teh (2009) to the compounding problem, as it would allow back-off through context-dropping and back-off through modifier-dropping at different context-lengths.

Finally, the PYP has recently also been used to devise language models that can leverage the output of traditional finite-state morphological analysers. Chahuneau et al. (2013b) define an open-vocabulary PYP-based word-model that generates words from morphemes. This word-model is then integrated into an HPYLM by replacing the uniform base distribution G_0, and performs well in their evaluation. Though related in the sense of also modelling sub-word elements, the HPYLM+c contrasts to their approach by seeking to model cross-word dependencies.

3.6 Summary

In this chapter we formulated a Bayesian n-gram language model capable of scoring compound words in terms of their components, which mitigates the data sparsity caused by productive compounding. We exploited the linguistic notion that the compound head is the most relevant part for syntactic coherence within a sentence. Our results on German indicate that this is a useful property to take into account, as the perplexity degraded when setting up the model to ignore the right-headedness of German. In an extrinsic evaluation in a machine translation system, the proposed language model contributed to outputting compound words with higher precision.

Example 1	
Src:	in the period 1994-2009 she along with her colleagues monitored more than 1,500 patients with **prostate cancer** in britain and in parallel over three thousand healthy men .
Ref:	im zeitraum von 1994-2009 verfolgte sie mit ihren kollegen in britannien mehr als 1500 patienten mit **prostatakrebs** , gleichzeitig aber auch 3000 gesunde männer .
HPYLM (i–ii):	im zeitraum 1994-2009 sie zusammen mit ihren kollegen überwacht mehr als 1.500 patienten mit **prostata-** in großbritannien und parallel über 3 000 gesunde männer .
HPYLM (iii):	in den jahren 1994-2009 sie zusammen mit ihren kollegen überwacht mehr als 1.500 patienten mit **prostata- krebs** in großbritannien und gleichzeitig über 3 000 gesunde männer .
HPYLM+c (i–iii):	in den jahren 1994-2009 sie zusammen mit ihren kollegen überwacht mehr als 1.500 patienten mit **prostatakrebs** in großbritannien und parallel über 3 000 gesunde männer .

Example 2	
Src:	co-chairs alan simpson and erskine bowles have taken on one of the most difficult jobs in washington and have , in their own estimation , forced the issue of the deficit onto the public **agenda** in ways that can 't be ignored .
Ref:	die stellvertretenden vorsitzenden alan simpson und erskine bowles haben einen der schwierigsten jobs in washington übernommen und haben , gemäß ihrer eigenen einschätzung , so sehr darauf gedrängt , die defizitfrage auf die öffentliche **tagesordnung** zu setzen , dass diese nicht mehr ignoriert werden können .
HPYLM (i–iii):	ko-vorsitzenden alan simpson und erskine bowles auf sich genommen haben eines der schwierigsten jobs in washington und haben , in ihrer eigenen einschätzung , die frage des defizits auf der öffentlichen **agenda** in einer weise , die nicht ignoriert werden kann .
HPYLM+c (i–iii):	ko-vorsitzenden alan simpson und erskine bowles haben eines der schwierigsten jobs in washington und haben , in ihrer eigenen einschätzung , die frage des defizits in einer art und weise auf die öffentliche **tagesordnung** , die nicht ignoriert werden kann .

Example 3	
Src:	the majority of members of the committee concluded that the message conveyed was that **suicide** should not be seen as a solution of life 's problems .
Ref:	die mehrheit der aufsichtsratsmitglieder hatte die schlussfolgerung gezogen , die vermittelte botschaft sei hierbei , dass **selbstmord** nicht als lösung von lebenskrisen angesehen werden dürfe .
HPYLM (i–iii):	die mehrheit der mitglieder des ausschusses zu dem schluss , dass die botschaft übermittelt wurde , dass **selbstmordattentate** sollte nicht als eine lösung der probleme des lebens betrachtet werden .
HPYLM+c (i–iii):	die mehrheit der mitglieder des ausschusses zu dem schluss , dass die botschaft übermittelt wurde , **selbstmord** sollte nicht als eine lösung der probleme des lebens betrachtet werden .

Table 3.8: Translation examples where compounds in the hypotheses are more correct with HPYLM+c than HPYLM as language model. Roman numerals mark the MT repetitions, which are shown individually only when variation occurred around the word of interest.

Chapter 4

Rich morphology in distributed language models

Chapter Abstract

In this chapter we develop a distributed language model that composes morpheme vectors into word vectors so that morphologically related words are linked in the model. We show that this leads to substantial improvements in perplexity and that the acquired morpheme vectors successfully enable the construction of representations for unknown words. A further contribution is to demonstrate the full integration of a distributed language model into a machine translation decoder, which leads to improvements in the translation quality metric. Material in this chapter was originally published as (Botha and Blunsom, 2014).

In the previous chapter we leveraged existing morphological information in the form of compound segmentations to improve the smoothing of n-gram distributions. While this lead to some performance improvements, our results highlighted the challenge in devising suitable back-off schemes that correctly prioritise the different types of generalisation (e.g. compound decomposition versus context truncation).

This provides part of the motivation for shifting the focus to distributed language models (DLMs), which circumvent the need for carefully constructing back-off schemes by modelling arbitrary correlations between the feature representations of context words and their subsequent target word.

Word-based distributed representations have been found to capture some morphological regularity automatically (Mikolov et al., 2013b), but we contend that there is a case for building *a priori* morphological awareness into the language models' inductive bias. Conversely, compositional vector-space modelling has recently been applied to morphology to good effect (Lazaridou et al., 2013; Luong et al., 2013), but lacked the probabilistic basis necessary for use with a machine translation decoder.

The method we propose strikes a balance between probabilistic language modelling and morphology-based representation learning. Word vectors are composed as a linear function of arbitrary sub-elements of the word, e.g. surface form, stem, affixes, or other latent information. The effect is to tie together the representations of morphologically related words, directly combating data sparsity. This is executed in the context of a log-bilinear (LBL) LM (Mnih and Hinton, 2007), which is sped up sufficiently by the use of word classing so that we can integrate the model into an open source machine translation decoder and evaluate its impact on translation into 6 languages, including the morphologically complex Czech, German and Russian.

4.1 Additive word representations

As introduced in chapter 2, a DLM associates with each word type v in the vocabulary $\mathcal{V}$ a d-dimensional feature vector $\mathbf{r}_v \in \mathbb{R}^d$. Regularities among words are captured in an opaque way through the interaction of these feature values and a set of transformation weights. This leverages linguistic intuitions only in an extremely rudimentary way, in contrast to hand-engineered linguistic features that target very specific phenomena, as often used in supervised-learning settings.

We seek a compromise that retains the unsupervised nature of DLM feature vectors, but also incorporates pre-existing morphological information into the model in a flexible and efficient manner. In particular, morphologically related words should share statistical strength in spite of differences in word form.

As a way of integrating additional information about a word type, we define a mapping $\mu : \mathcal{V} \mapsto \mathcal{F}^+$ of a surface word into a variable-length sequence of *factors*, i.e. $\mu(v) = (f_1, \ldots, f_{K(v)})$, where $v \in \mathcal{V}, f_i \in \mathcal{F}$ and $1 \leq i \leq K(v)$. A factor f, which could be a word's stem or suffix, or some other relevant sub-word element, has an associated *factor feature vector* $\mathbf{r}_f \in \mathbb{R}^d$. Under this view, the vector representation $\tilde{\mathbf{r}}_v$ of a word v is reformulated as a function $\omega_\mu(v)$ of its factor vectors, adding them together:

$$\tilde{\mathbf{r}}_v = \omega_\mu(v) = \sum_{f \in \mu(v)} \mathbf{r}_f \tag{4.1}$$

The vectors of morphologically related words become explicitly linked through shared factor vectors, for example

$$\begin{aligned} \overrightarrow{\mathrm{imperfection}} &= \overrightarrow{im} + \overrightarrow{perfect} + \overrightarrow{ion} \\ \overrightarrow{\mathrm{perfectly}} &= \overrightarrow{perfect} + \overrightarrow{ly}, \end{aligned}$$

where we use the notation $\overrightarrow{\mathrm{word}}$ and $\overrightarrow{factor}$.

A valuable by-product of this approach is that it gives a well-defined method for constructing representations for out-of-vocabulary (OOV) words using the available morpheme vectors.

In practice, we set up the factorisation μ to also include the surface form of a word as a factor. This is done to account for non-compositional constructions such as

$$\overrightarrow{\mathrm{greenhouse}} \neq \overrightarrow{green} + \overrightarrow{house}$$

and also makes the approach more robust to noisy morphological segmentations. The word representation for the preceding example is therefore

$$\overrightarrow{\mathrm{greenhouse}} = \overrightarrow{greenhouse} + \overrightarrow{green} + \overrightarrow{house}.$$

An obvious shortcoming in the choice of addition as composition function ω is its commutativity. Inclusion of the surface factor also overcomes the associated invariance in the order of morpheme factors, with the desired outcome that $\overrightarrow{\mathrm{houseboat}} \neq \overrightarrow{\mathrm{boathouse}}$, for example.

The number of factors per word is free to vary across the vocabulary, making the approach applicable across the spectrum of more fusional languages (e.g. Czech, Russian) to more agglutinative languages (e.g. Turkish). This is in contrast to factored language models (Alexandrescu and Kirchhoff, 2006), which assume a fixed number of factors per word. Their method of concatenating factor vectors to obtain a single representation vector for a word can be seen as enforcing a partition on the feature space. Our method of addition avoids such a partitioning and better reflects the absence of a strong intuition about what an appropriate partitioning might be. A limitation of our method compared to theirs is that the deterministic mapping μ currently enforces a single factorisation per word type, which sacrifices information obtainable from context-disambiguated morphological analyses.

The additive composition function ω can be regarded as an instantiation of the weighted addition strategy *wadd* that performed well in the distributional semantics approach to derivational morphology reported by Lazaridou et al. (2013). Their weighted addition involves using a global pair of weights α and β to compose a $\overrightarrow{\text{word}}$ as $\alpha \times \overrightarrow{stem} + \beta \times \overrightarrow{suffix}$. (Their focus is limited to the segmentation of a word into a stem and suffix, and evaluated only on English.) Importantly, they fit the two weights while keeping the vector representations fixed. As described shortly, we situate our composition method within a distributed language model. Our feature representations can therefore adjust jointly with the effect of the composition. To a first approximation, individual feature values can thereby absorb a weighting component specific to a morpheme type or a feature dimension, rather than using a single pair of global weights.

Another aspect of our composition method is how it models derivational ambiguity. Unlike the recursive neural-network method of Luong et al. (2013), we do not impose a single tree structure over a word, which ignores the semantic ambiguity inherent in words like un[[lock]able] vs. [un[lock]]able.[1] The consideration is that a model should either account for such ambiguity directly, or operate in a way that

---

[1] Example due to de Almeida and Libben (2005).

treats alternative derivations uniformly so that it makes minimal assumptions about the language in question. Our approach is to do the latter.

In contrast to these two recent approaches to vector-based morphological modelling, our additive representations are readily implementable in a probabilistic language model suitable for use in a decoder.[2]

4.2 Log-bilinear language models

In this section we formalise the distributed language model into which we subsequently integrate our compositional approach to morphology.

Log-bilinear (LBL) models (Mnih and Hinton, 2007) are an instance of DLMs that make the same Markov assumption as n-gram language models, as introduced in chapter 2. The conditional distribution $P(w_i \mid w_{i-n+1}^{i-1})$ of a word given its context is modelled by a smooth scoring function $\nu(\cdot)$ over vector representations of words.

The LBL predicts the vector $\mathbf{p}$ for the next word as a function of the context vectors $\mathbf{q}_j \in \mathbb{R}^d$ of the preceding n–1 words,

$$\mathbf{p} = \boldsymbol{\eta}\left(\sum_{j=1}^{n-1} \mathbf{q}_j C_j\right), \tag{4.2}$$

where $C_j \in \mathbb{R}^{d \times d}$ are position-specific weight matrices, and $\boldsymbol{\eta}$ is the identity function $\boldsymbol{\eta}(\mathbf{x}) = \mathbf{x}$, written this way for a subsequent variation.

The continuous scoring function $\nu(w)$ measures how well a word w fits the predicted representation $\mathbf{p}$ and is defined as

$$\nu(w) = \mathbf{p} \cdot \mathbf{r}_w + b_w, \tag{4.3}$$

where $\mathbf{r}_w \in \mathbb{R}^d$ is the representation vector of the target word w and b_w is a bias term encoding the prior probability of the word type. The conditional probability of the word is calculated through exponentiation and normalisation of the scoring function:

$$P\left(w_i | w_{i-n+1}^{i-1}\right) = \frac{\exp\left(\nu(w_i)\right)}{\sum_{v \in \mathcal{V}} \exp\left(\nu(v)\right)}. \tag{4.4}$$

---

[2] Our focus is on surface morphology, but the method could equally be used to integrate other information, e.g. PoS, lemma.

This model is subsequently denoted as **LBL** with parameters $\Theta_{\mathsf{LBL}} = (C_j, Q, R, \mathbf{b})$, where $Q, R \in \mathbb{R}^{|\mathcal{V}|\times d}$ contain the word representation vectors as rows, and $\mathbf{b} \in \mathbb{R}^{|\mathcal{V}|}$. Q and R imply that separate representations are used for conditioning and output. See Figure 4.1a for an illustrative diagram.

4.3 Variants of the log-bilinear model

4.3.1 Additive log-bilinear model

We introduce a variant of the LBL that makes use of additive representations (§4.1) by associating the *composed word vectors* $\tilde{\mathbf{r}}$ and $\tilde{\mathbf{q}}_j$ with the target and context words, respectively. We redefine the representation matrices to be $Q^{(f)}, R^{(f)} \in \mathbb{R}^{|\mathcal{F}|\times d}$ thus containing a vector for each factor type in the global vocabulary of factors $\mathcal{F}$. This model is designated **LBL++** and has parameters $\Theta_{\mathsf{LBL++}} = \big(C_j, Q^{(f)}, R^{(f)}, \mathbf{b}\big)$. See the example in Figure 4.1c.

Words sharing factors are tied together, which is expected to improve performance on rare word forms. Note that the composed word vectors are not model parameters in their own right, but are strictly a function of the factor vectors according to the composition function ω and mapping μ. The scoring function for this model is an expansion of Equations 4.2–4.3 to include the additive representations:

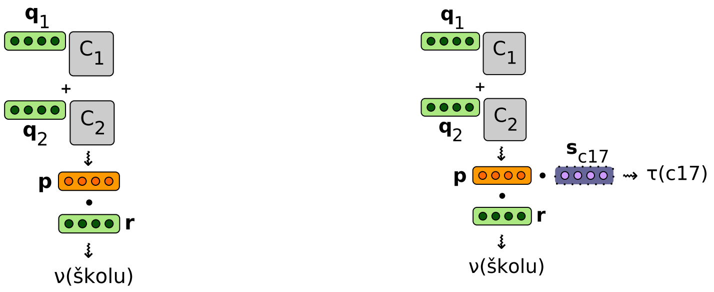
$$\nu_{\mathsf{LBL++}}(w) = \boldsymbol{\eta}\left(\sum_{j=1}^{n-1} \tilde{\mathbf{q}}_j C_j\right) \cdot \tilde{\mathbf{r}}_w + b_w \tag{4.5}$$

$$= \boldsymbol{\eta}\left(\sum_{j=1}^{n-1} \Big(\sum_{f\in\mu(w_j)} \mathbf{q}_f\Big) C_j\right) \cdot \Big(\sum_{f\in\mu(w)} \mathbf{r}_f\Big) + b_w. \tag{4.6}$$

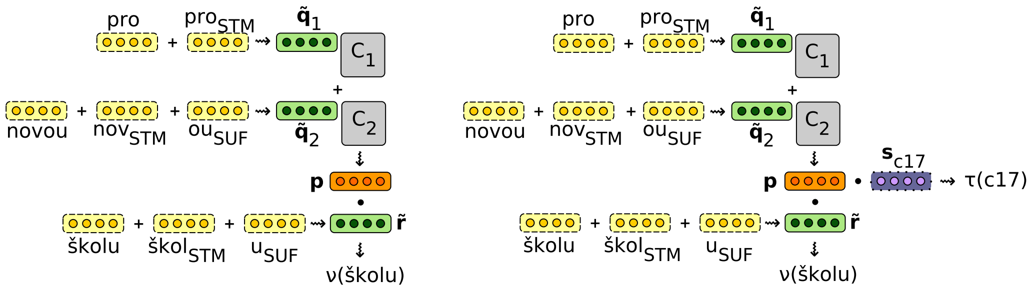
Similar to Equation 4.4, the probability model is

$$P\left(w_i | w_{i-n+1}^{i-1}\right) = \frac{\exp\left(\nu_{\mathsf{LBL++}}(w_i)\right)}{\sum_{v\in\mathcal{V}} \exp\left(\nu_{\mathsf{LBL++}}(v)\right)}. \tag{4.7}$$

Representing the mapping μ with a sparse transformation matrix $M \in \mathbb{Z}_+^{\mathcal{V}\times|\mathcal{F}|}$ establishes the relationship between composed word and factor representation matrices as $R = MR^{(f)}$ and $Q = MQ^{(f)}$. A row in M selects the factor vectors for



(a) LBL (b) CLBL



(c) LBL++ (d) CLBL++

Figure 4.1: Illustration of how three different 3-gram model variants treat the Czech phrase *pro novou školu* (‘for [the] new school’), where the preposition causes accusative case markings on the adjective and noun. In (b) and (d), we assume the vocabulary clustering (§4.3.2) assigned class 17 to the target word.

a given word type from the $R^{(f)}$ (or $Q^{(f)}$) matrix. These row-vectors are a generalisation of one-hot vectors: they have multiple non-zero entries, corresponding to the multiple factors that make up a word, and integer values greater than one would encode multiple occurrences of the same factor within a given word. In practice, we exploit this correspondence between composed word representations and factor representations for test-time efficiency—the composed word vectors are compiled offline so that the computational cost of LBL++ probability lookups is equivalent to that of LBL.

We consider two obvious variations of the LBL++ to evaluate the extent to which interactions between context and target factors affect the model: **LBL+o** only factorises **o**utput words and retains simple word vectors for the context (i.e. $Q \equiv Q^{(f)}$), while **LBL+c** does the reverse, only factorising **c**ontext words.[3] Both reduce to the LBL when setting μ to be the identity function, such that $\mathcal{V} \equiv \mathcal{F}$.

The factorisation permits an approach to handling unknown context words that is less harsh than the standard method of replacing them with a global unknown symbol—instead, a vector can be constructed from the known factors of the word (e.g. the observed stem of an unobserved inflected form). A similar scheme can be used for scoring unknown target words, but requires changing the event space of the probabilistic model. We use this vocabulary stretching capability in our word similarity experiments, but leave the extensions for test-time language model predictions as future work.

A consequence of the additional free parameters introduced by the additive representations is that a maximum-likelihood training criterion would favour the degenerate solution where the non-surface vectors in $\mathcal{F} \setminus \mathcal{V}$ are pushed to zero, rendering the additive extension ineffective. Including a regularisation term in the training objective is crucial and should yield non-trivial values for the parameters tied through our additive representations. (Figure 4.2 in the experimental section demonstrates the sensitivity of performance to the regularisation strength.)

4.3.2 Class-based model decomposition

DLMs are known to achieve much lower perplexities than conventional n-gram models, but their application to decoder-based tasks like translation and speech recognition have largely been limited to rescoring of lattices or n-best lists of hypotheses. However, they could exercise a greater influence in those tasks if integrated into the decoders.

The key obstacle to using DLMs in a decoder is the expensive normalisation over the vocabulary (Equation 4.4). Our approach to reducing the computational cost of normalisation is to use a class-based decomposition of the probabilistic model (Goodman, 2001; Mikolov et al., 2011). Using Brown-clustering (Brown et al.,

---

[3] The +c, +o and ++ naming suffixes denote these same distinctions when used with the CLBL model introduced later.

1992),[4] we partition the vocabulary into $|\mathcal{C}|$ classes, denoting as $\mathcal{C}_c$ the set of vocabulary items in class c, such that $\mathcal{V} = \mathcal{C}_1 \cup \cdots \cup \mathcal{C}_{|\mathcal{C}|}$. This is a hard clustering, which we treat as fixed and observed for a given vocabulary and dataset. The class $c(w)$ of a word $w \in \mathcal{V}$ can therefore be looked up. We emphasise that we are working in a discriminative setting.

In this model, the probability of a word conditioned on the history h of $n-1$ preceding words[5] is decomposed as

$$\begin{aligned} P(w|h) &= \sum_{c' \in \mathcal{C}} P(c'|h)P(w|h,c') && (4.8) \\ &= P(c(w)|h)P(w|h,c(w)) + \sum_{c' \in \mathcal{C}, c' \neq c(w)} (P(c'|h) \cdot 0) && (4.9) \\ &= P(c(w)|h)P(w|h,c(w)). && (4.10) \end{aligned}$$

This class-based model, **CLBL**, extends the LBL by associating a representation vector $\mathbf{s}_c$ and bias parameter t_c with each class c, such that $\Theta_{\mathsf{CLBL}} = (C_j, Q, R, S, \mathbf{b}, \mathbf{t})$. The same prediction vector $\mathbf{p}$ is used to compute both class score $\tau(c) = \mathbf{p} \cdot \mathbf{s}_c + t_c$ and word score $\nu(w)$, which are normalised separately:

$$P(c|h) = \frac{\exp(\tau(c))}{\sum_{c'=1}^{|\mathcal{C}|} \exp(\tau(c'))} \tag{4.11}$$

$$P(w|h,c) = \frac{\exp(\nu(w))}{\sum_{v' \in \mathcal{C}_c} \exp(\nu(v'))}. \tag{4.12}$$

Figure 4.1b provides an illustration.

We favour this flat vocabulary partitioning for its computational adequacy, simplicity and robustness. Computational adequacy is obtained by using $|\mathcal{C}| \approx |\mathcal{V}|^{0.5}$, thereby reducing the $\mathcal{O}(|\mathcal{V}|)$ normalisation operation of the LBL to two $\mathcal{O}(|\mathcal{V}|^{0.5})$ operations in the CLBL, as Brown clustering mostly yields balanced class sizes.

---

[4] In preliminary experiments, Brown clusters gave much better perplexities than frequency-binning (Mikolov et al., 2011).

[5] This is therefore not a class-based language model in the sense of predicting the next class conditioned on the preceding *classes* (Brown et al., 1992).

Other methods for achieving more drastic complexity reductions exist in the form of frequency-based truncation, shortlists (Schwenk, 2004), or casting the vocabulary as a full hierarchy (Mnih and Hinton, 2008) or partial hierarchy (Le et al., 2011). We expect these approaches could have adverse effects in the rich morphology setting, where much of the vocabulary is in the long tail of the word distribution.

4.3.3 Non-linearity

The use of non-linear activation functions in DLMs are known to improve performance over strictly linear ones (Mnih et al., 2009; Pachitariu and Sahani, 2013), but comes at a cost. Typical sigmoidal and $\tanh$ non-linearities shrink the gradients, which causes slower convergence during training than the linear models (~20 vs. ~4 iterations in our experience with CLBL, §4.3.2). Non-linearities can create difficulties during training (Pascanu et al., 2012) and incur a computational cost at test-time that one would wish to avoid when using language models in a decoder. Rectified linear units (Nair and Hinton, 2010) have been used as a workaround (Vaswani et al., 2013), but a pertinent question is whether a simple technique like our additive representations might serve as a substitute for non-linearities altogether.[6]

For comparison, we define a non-linear variant **nLBL** that uses a sigmoidal activation function on the prediction vector, i.e. $\boldsymbol{\eta} \equiv \boldsymbol{\sigma}$ (cf. Equation 4.2), which operates element-wise as $\sigma(x) = (1 + e^{-x})^{-1}$. Our definition here is simpler than the various options explored by Mnih et al. (2009) in that it only applies to the final prediction vector.

4.4 Experiments

The overarching aim of our evaluation is to investigate the effect of using the proposed additive representations across languages with a range of morphological complexity.

---

[6] This consideration is limited to single layer networks—in multi-layer networks, non-linearities take on a defining role.

Our intrinsic language model evaluation has two parts. We first perform a model selection experiment on small data to consider the relative merits of using additive representations for context words, target words, or both, and to validate the use of the class-based decomposition.

Then we consider class-based additive models trained on tens of millions of tokens and large vocabularies. These larger language models are applied in two extrinsic tasks: i) a word-similarity rating experiment on multiple languages, aiming to gauge the quality of the induced word and morpheme representation vectors; ii) a machine translation experiment, where we are specifically interested in testing the impact of an LBL LM feature when translating into morphologically rich languages.

4.4.1 Training and initialisation

Model parameters Θ are estimated by optimising an L2-regularised log likelihood objective. Training the CLBL and its additive variants directly against this objective is fast because normalisation of model scores, which is required in computing gradients, is over a small number of events, namely classes, and word types in a given class.

For the classless LBLs we use noise-contrastive estimation (NCE) (Gutmann and Hyvärinen, 2012; Mnih and Teh, 2012) to avoid normalisation during training. NCE achieves this by casting the problem as one of discriminating between real training instances and synthetically generated “noise” instances. For each unigram in the data, we generated 100 noise instances from the empirical unigram distribution. While NCE speeds up training, it leaves the expensive test-time normalisation of LBLs unchanged, precluding their usage during decoding.

Bias terms b (resp. t) are initialised to the log unigram probabilities of words (resp. classes) in the training corpus, with Laplace smoothing, while all other parameters are initialised randomly according to sharp, zero-mean Gaussians. Representations are thus learnt from scratch and not based on publicly available embeddings, meaning our approach can easily be applied to many languages.

	DATA-1M		DATA-MAIN		
	Tokens	$\lvert\mathcal{V}\rvert$	Tokens	$\lvert\mathcal{V}\rvert$	Sentence Pairs
CS	1m	46k	16.8m	206k	0.7m
DE	1m	36k	50.9m	339k	1.9m
EN	1m	17k	19.5m	60k	0.7m
ES	1m	27k	56.2m	152k	2.0m
FR	1m	25k	57.4m	137k	2.0m
RU	1m	62k	25.1m	497k	1.5m

Table 4.1: Corpus statistics. The number of sentence pairs for a row X refers to the English→X parallel data (but row EN has Czech as source language).

Optimisation is performed by stochastic gradient descent with updates after each mini-batch of L training examples. We apply AdaGrad (Duchi et al., 2011), which effectively adjusts the learning rate individually for each model parameter as a function of a global step-size hyperparameter ξ, which we tune on held-out development data. The number of training iterations I is also treated as a hyperparameter, and training is halted once the perplexity on the development data starts to increase.[7]

4.4.2 Data and methods

We make use of data from the 2013 ACL Workshop on Machine Translation.[8] We first describe data used for translation experiments, since the monolingual datasets used for language model training were derived from that. The language pairs are English→{German, French, Spanish, Russian} and English↔Czech. Our parallel data comprised the Europarl-v7 and news-commentary corpora, except for English–Russian where we used news-commentary and the Yandex parallel corpus.[9] Preprocessing involved lowercasing, tokenising and filtering to exclude sentences of more than 80 tokens or substantially different lengths.

4-gram language models were trained on the target data in two batches: DATA-1M consists of the first million tokens only, while DATA-MAIN uses the full target-side of the corpora listed above. Statistics are given in Table 4.1. newstest2011 was

---

[7] L=10k–40k, ξ=0.05–0.08, dependent on $\lvert\mathcal{V}\rvert$ and data size; I=2–4 passes through the data for the linear models.

[8] http://www.statmt.org/wmt13/translation-task.html

[9] https://translate.yandex.ru/corpus?lang=en

used as development data[10] for tuning language model hyperparameters, while intrinsic LM evaluation was done on newstest2012, measuring perplexity. In addition to contrasting the LBL variants, we also compare against the MKN baseline.

4.4.2.1 Language model vocabulary

Additive representations that link morphologically related words specifically aim to improve modelling of the long tail of the lexicon, so we do not want to prune away all rare words, as is common practice in language modelling and word embedding learning. We define a *singleton pruning rate* κ, and randomly replace a fraction κ of words occurring only once in the training data with a global UNK symbol. We use low pruning rates[11] and thus model large vocabularies.[12]

4.4.2.2 Word factorisation μ

We obtain morphological segmentations from the unsupervised segmentor *Morfessor Cat-MAP* (Creutz and Lagus, 2007), which takes as input a list of word forms and their frequencies in a corpus. The output consists of the segmentation points along with labels that identify each segment as a prefix, stem or suffix. For example, the word *unsegmented* is analysed as un_{PRE} $segment_{STM}$ ed_{SUF}.

The mapping μ of a word is taken as its surface form and the morphemes identified by Morfessor. Keeping the morpheme labels allows the model to learn separate vectors for, say, in_{STM} the preposition and in_{PRE} occurring as “in-appropriate”. By not post-processing segmentations in a more sophisticated way (cf. Luong et al. (2013)) we keep the overall method language independent. Table 4.2 provides an example from the word map derived for our Czech data set.

The choice of an unsupervised method of segmentation in this chapter stands in contrast to the supervised compound segmentor used in chapter 3. The evaluation

---

[10] The newstest2011 data set does not feature Russian, so for that language some training data was held out for tuning.

[11] DATA-1M: $\kappa = 0.2$; DATA-MAIN: $\kappa = 0.05$

[12] We also mapped digits to 0, (i.e. 15.1%⇒00.0%), and cleaned the Russian data by replacing tokens having <80% Cyrillic characters with UNK.

Word v	***Factors*** $\left(f_1, \ldots, f_{\mid\mu(v)\mid}\right)$
aktivovanými	*aktivovanými aktiv*$_{STM}$ *ova*$_{SUF}$ *ný*$_{SUF}$ *mi*$_{SUF}$
čerpány	*čerpány čerpá*$_{STM}$ *ny*$_{SUF}$
dorozumívat	*dorozumívat do*$_{PRE}$ *rozum*$_{STM}$ *í*$_{SUF}$ *vat*$_{STM}$
modlitba	*modlitba modlitba*$_{STM}$
nejvážnější	*nejvážnější nej*$_{PRE}$ *váž*$_{STM}$ *nější*$_{SUF}$
nepodkopal	*nepodkopal ne*$_{PRE}$ *podkopal*$_{SUF}$
škola	*škola škol*$_{STM}$ *a*$_{SUF}$
školou	*školou škol*$_{STM}$ *ou*$_{SUF}$
školu	*školu škol*$_{STM}$ *u*$_{SUF}$
školy	*školy škol*$_{STM}$ *y*$_{SUF}$

Table 4.2: A fragment of the word map $\mu(v)$ obtained for Czech.

	Data-1M	Data-Main
Cs	2.27	1.92
De	2.81	2.98
En	1.72	1.50
Es	2.03	1.71
Fr	2.26	1.78
Ru	2.32	1.84

Table 4.3: Average number of morphemes per word type obtained with Morfessor for each language (see §4.4.2.2). These numbers reflect the segmentation only and do not include the surface factors.

setup there was simple given the focus on one language, so that there was no downside in relying on a supervised segmentor. In this chapter, we aim to have a broader evaluation on multiple languages. The use of an unsupervised tool such a Morfessor simplifies the empirical work by not requiring special models or calibration for each new language, even though this approach may affect segmentation quality adversely. The LBL++ models are specifically devised to be robust against noisy segmentation. In the experiments that follow, we set the single *perplexity threshold* parameter of Morfessor to 50 and 400 for Data-1M and Data-Main, respectively, based on prior experience. Segmentation quality was gauged informally by manual inspection of the output. Table 4.3 provides statistics that summarise the degree of segmentation obtained in each language.

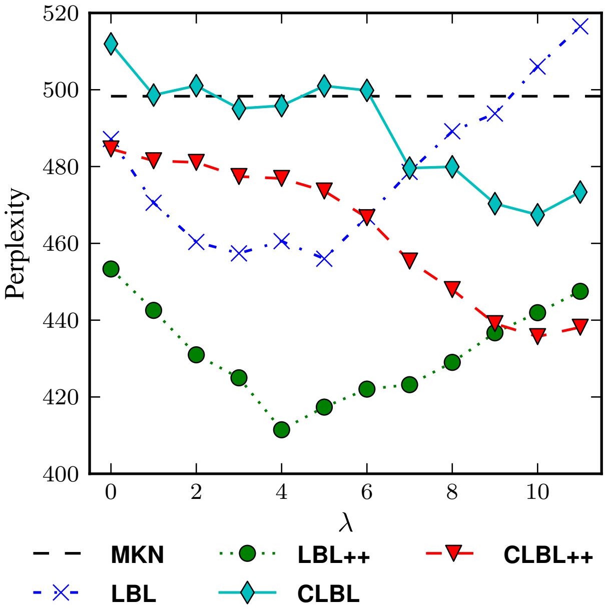


Figure 4.2: Relative performance of models is sensitive to regularisation strength λ when using 1m training tokens. Perplexities are measured on the devset newstest2011.

4.4.3 Model selection

Results on DATA-1M. The LBL training objectives include a weight penalty term $0.5\lambda\|\boldsymbol{\theta}\|^2$, which we tune against the newstest2011 dataset. For the 1m-word corpus size, we find tuning λ to be vital for good model performance (Figure 4.2).

The use of morphology-based, additive representations for both context and output words (models++) yielded perplexity reductions on all 6 languages when using 1m training tokens (Table 4.4). Furthermore, these double-additive models consistently outperform the ones that factorise only context (+c) or only output (+o) words, indicating that context and output contribute complementary information and supporting our hypothesis that is it beneficial to model morphological dependencies *across* words. The results are given by language in Table 4.4 and summarised visually in Figure 4.3.

The impact of our CLBL++ method varies by language, correlating with vocabulary size: Russian benefited most, followed by Czech and German. Even on English, often regarded as having simple morphology, the relative improvement is 4%.

The relative merits of the +c and +o schemes depend on which model is used as

	MKN	CLBL	+c	+o	++	++	nCLBL	+c	+o	++	++
Cs	550	508	-4.0%	-3.4%	-7.4%	470	487	-3.0%	0.7%	-4.6%	**465**
De	366	316	-3.5%	-1.0%	-4.3%	303	309	-2.8%	-0.6%	-4.2%	**296**
En	330	303	-1.6%	-1.3%	-3.8%	**291**	301	-2.6%	0.1%	-3.0%	292
Es	241	212	-3.6%	-2.3%	-4.9%	202	209	-2.1%	-0.2%	-4.4%	**200**
Fr	274	239	-2.5%	-1.5%	-5.5%	226	230	-1.6%	-0.7%	-2.4%	**225**
Ru	396	343	-8.0%	-3.8%	-11.4%	**304**	353	-5.4%	-0.6%	-8.3%	324
avg	360	320	-3.9%	-2.2%	-6.2%	**299**	315	-2.9%	-0.2%	-4.5%	300

Table 4.4: Test-set perplexities for DATA-1M, measuring the separate effects of context-only (+c), output-only (+b) and double (++) additive representations when applied to the linear and non-linear class-based LBL. Additive model results are also given in terms of percentage reduction in perplexity relative to the corresponding non-additive model.

starting point. With LBL, the output-additive scheme (LBL+o) gives larger improvements than the context-additive scheme (LBL+c) (Figure 4.3). The reverse is true for CLBL, indicating the class decomposition dampens the effectiveness of using morphological information in output words.

The non-linear class-based model nCLBL obtains improvements up to 4% over its linear counterpart, except for Russian where it performs worse. Significantly, we observe that our purely linear CLBL++ outperforms the non-linear nCLBL model, establishing an alternative to the non-linear model. Using additive representations with the non-linear model further decreases perplexity slightly, though the relative improvements obtained when using them for output words only are very small.

The use of classes increases perplexity slightly compared to the LBLs (see Table 4.5), but this is in exchange for much faster computation of language model probabilities, allowing the CLBLs to be used in a machine translation decoder (§4.4.7). On average, the CLBLs obtain a 35-fold speed-up over the LBLs. The running time of a model was calculated by measuring the total wall-time it took to compute the test-set perplexity, and then averaging across languages. For example, the LBL computed the test-set perplexity in 209 seconds, while the CLBL did so in 6 seconds. The test sets contain 72650 tokens on average.

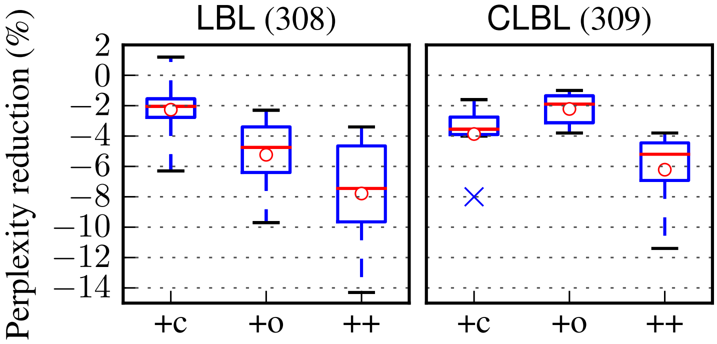


Figure 4.3: Model selection results. Box-plots show the spread, across 6 languages, of relative perplexity reductions obtained by each type of additive model against its non-additive baseline, for which median absolute perplexity is given in parentheses; for MKN, this is 348. Each box-plot summarises the behaviour of a model across languages. Circles give sample means, while crosses show outliers beyond 3× the inter-quartile range.

	MKN	360	
LBL	317	CLBL	320
LBL+c	309	CLBL+c	307
LBL+o	299	CLBL+o	313
LBL++	**290**	CLBL++	299

Table 4.5: The role of classes. Test-set perplexity averaged across languages, using DATA-1M for training.

4.4.4 Intrinsic evaluation

Results on DATA-MAIN. Based on the outcomes of the small-scale evaluation, we focus our main language model evaluation on the additive class-based model CLBL++ in comparison to CLBL and MKN baselines, using the larger training dataset, with vocabularies of up to 500k word types.

The overall trend that morphology-based additive representations yield lower perplexity carries over to this larger data setting, again with the biggest impact being on Czech and Russian (Table 4.6, top). Perplexity improvements are in the 2%–6% range, slightly lower than the corresponding differences on the small data.

Our hypothesis is that much of the improvement is due to the additive representations being especially beneficial for modelling rare words. We test this by repeating the experiment under the condition where all word types occurring only once are excluded from the vocabulary (κ=1). If the additive representations were not benefi-

			CLBL		CLBL++	
		MKN	abs	rel1	abs	rel2
κ=0.05	Cs	862	683	-20.8%	643	-5.9%
	De	463	422	-8.9%	404	-4.2%
	En	291	281	-3.4%	273	-2.8%
	Es	219	207	-5.7%	203	-1.9%
	Fr	243	232	-4.9%	227	-1.9%
	Ru	390	313	-19.7%	300	-4.2%
	avg	411	356	-10.6%	342	-3.5%
κ=1.0	Cs	634	477	-24.8%	462	-3.1%
	De	379	331	-12.6%	329	-0.9%
	En	254	234	-7.6%	233	-0.7%
	Es	195	180	-7.7%	180	0.02%
	Fr	218	201	-7.7%	198	-1.3%
	Ru	347	271	-21.8%	262	-3.4%
	avg	338	282	-13.7%	277	-1.6%

Table 4.6: Test-set perplexities for DATA-MAIN using two vocabulary pruning settings. Percentage reductions are relative to the preceding model, e.g. the first Czech CLBL improves over MKN by 20.8% (rel1); the CLBL++ improves over that CLBL by a *further* 5.9% (rel2).

cial to rare words, the outcome should remain the same. Instead, we find the relative improvements become a lot smaller (Table 4.6, bottom) than when only excluding some singletons (κ=0.05), which supports that hypothesis.

Model perplexity on a whole dataset is a convenient summary of its intrinsic performance, but such a global view does not give much insight into *how* one model outperforms another. To focus more directly on the source of improvements, we partition the test data into subsets of interest and measure perplexity over these subsets.

4.4.4.1 Analysis by frequency

We first partition on token frequency, as computed on the training data. Figure 4.4 provides further evidence that the additive models have most impact on rare words generally, and not only on singletons. Czech, German and Russian see relative perplexity reductions of 8%–21% for words occurring fewer than 100 times in the training data. Reductions become negligible for the high-frequency tokens. These tend to be punctuation and closed-class words, where any putative relevance of morphol-

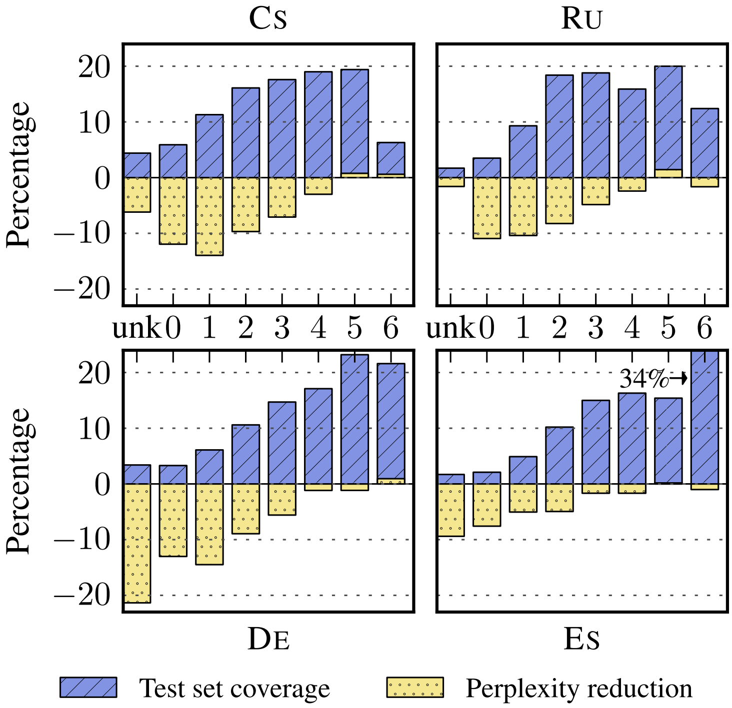


Figure 4.4: Perplexity reductions by token frequency, CLBL++ relative to CLBL. Dotted bars extending further down are better. A bin labelled with a number x contains those test tokens that occur $y \in [10^x, 10^{x+1})$ times in the training data. Striped bars give percentage test-set coverage for each bin.

ogy is overwhelmed by the fact that the predictive uncertainty is very low to begin with (absolute PPL<10 for the highest frequency subset). For the morphologically simpler Spanish case, perplexity reductions are generally smaller across frequency scales.

4.4.4.2 Analysis by part of speech

We also break down perplexity reductions by part of speech tags, focusing on German. We used the decision tree-based tagger of Schmid and Laws (2008), which reportedly has a tagging accuracy of 91%. Aside from unseen tokens, the biggest improvements are on nouns and adjectives (Figure 4.5), suggesting our segmentation-based representations help abstract over German's productive compounding.

German noun phrases require agreement in gender, case and number, which are marked overtly with fusional morphemes, and we see large gains on such test n-grams: 15% improvement on adjective-noun sequences, and 21% when consid-

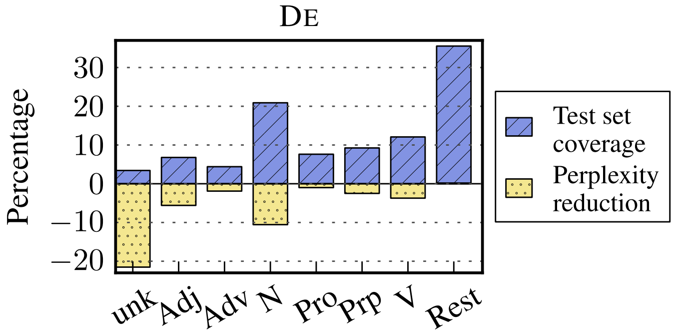


Figure 4.5: Perplexity reductions by part of speech, CLBL++ relative to CLBL on German. Dotted bars extending further down are better. Tokens tagged as foreign words or other opaque symbols resort under "Rest". Striped bars are as in Figure 4.4.

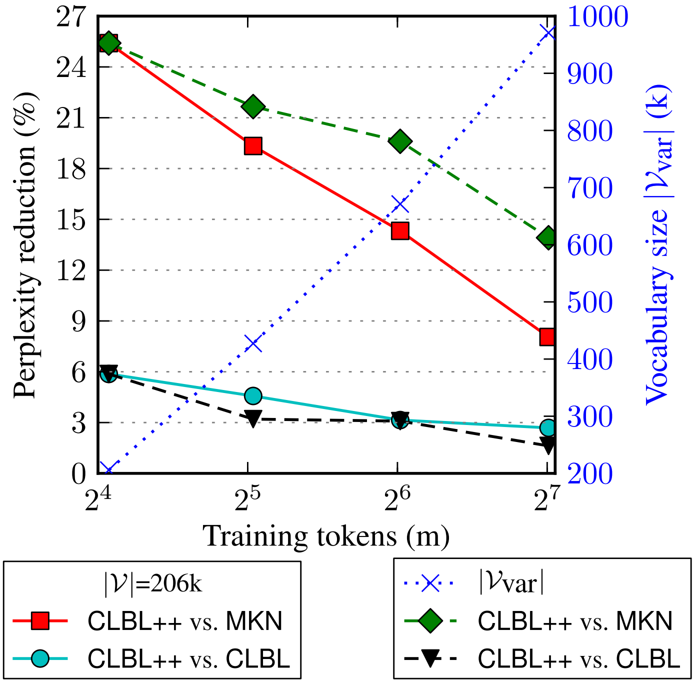


Figure 4.6: Relative perplexity reductions obtained when varying the Czech training data size (16m–128m). In the first batch, the vocabulary was first held fixed to the set $\mathcal{V}$ as obtained for DATA-MAIN (squares & circles). In the second batch, the vocabulary $\mathcal{V}_{\text{var}}$ varied according to the data set of the given size (diamonds & triangles).

ering the more specific case of adjective-adjective-noun sequences. An example of the latter kind is "der ehemalig•e sozial•ist•isch•e bildung•s•minister" (*the former socialist minister of education*), where the morphological agreement surfaces in the repeated e-suffix.

4.4.4.3 Scaling

We conducted a final scaling experiment on Czech by training models on increasing amounts of data from the monolingual news corpora. Improvements over the MKN baseline decrease, but remain substantial at 14% for the largest setting when allowing the vocabulary to grow with the data. Maintaining a constant advantage over MKN requires also increasing the dimensionality d of representations (Mikolov et al., 2013a), but this was outside the scope of our experiment. Although gains from the additive representations over the CLBL diminish down to 2%–3% at the scale of 128m training tokens (Table 4.6), these results demonstrate the tractability of our approach on large vocabularies of nearly 1m types.

4.4.5 Application to word similarity rating

In the previous section, we established the positive role that morphological awareness played in building DLMs that better predict unseen text. Here we focus on the quality of the word representations learnt in the process. We evaluate on a standard word similarity rating task, where one measures the correlation between cosine-similarity scores for pairs of word vectors and a set of human similarity ratings. An important aspect of our evaluation is to measure performance on multiple languages using a single unsupervised, model-based approach.

Morpheme vectors from the CLBL++ enable handling OOV test words in a more nuanced way than using the global unknown word vector. In general, we compose a vector $\tilde{\mathbf{u}}_v = [\tilde{\mathbf{q}}_v; \tilde{\mathbf{r}}_v]$ for a word v according to a *post hoc* word map μ' by summing and concatenating the factor vectors $\mathbf{r}_f$ and $\mathbf{q}_f$, where $f \in \mu'(v) \cap \mathcal{F}$. This ignores unknown morphemes occurring in OOV words, and uses $[\mathbf{q}_{\text{UNK}}; \mathbf{r}_{\text{UNK}}]$ for $\tilde{\mathbf{u}}_{\text{UNK}}$ only if all morphemes are unknown.[13]

To see whether the morphological representations improve the quality of vectors for known words, we also report the correlations obtained when using the CLBL++ word vectors directly, resorting to $\tilde{\mathbf{u}}_{\text{UNK}}$ for all OOV words $v \notin \mathcal{V}$ (denoted

---

[13] Concatenation of context and target vectors performed better overall than using either type individually, suggesting the two representation spaces capture complementary information.

"$-$*compose*" in the results). This is also the strategy that the baseline CLBL model is forced to follow for OOVs.

We evaluate first using the English rare-word dataset (RW) created by Luong et al. (2013). Its 2034 word pairs contain more morphological complexity than other well-established word similarity datasets, e.g. crudeness—impoliteness. We compare against their context-sensitive morphological recursive neural network (csmRNN), using Spearman's rank correlation coefficient, ρ. Table 4.7 shows our model obtaining a ρ-value between the two csmRNN results, depending on which baseline word embeddings they used to initialise the csmRNN.

This is a strong result given that our vectors come from a simple linear probabilistic model that is also suitable for integration directly into a decoder for translation (§4.4.7) or speech recognition, which is not the case for csmRNNs. Moreover, the csmRNNs were initialised with high-quality, publicly available word embeddings trained over weeks on much larger corpora of 630m–990m words (Collobert and Weston, 2008; Huang et al., 2012), in contrast to ours that are trained from scratch on much less data. This renders our method directly applicable to languages which may not yet have those resources.

A minor difference in configuration is that our vectors are 100-dimensional as opposed to the csmRNNs' 50 dimensional ones. Using embeddings[14] from the hierarchical LBL (Mnih and Hinton, 2008), we verified that the doubling in dimensionality increased ρ by less than 0.02 absolute, so this difference should be negligible in our evaluation.

Relative to the CLBL baseline, our method performs well on datasets across four languages. For the English RW, which was designed with morphology in mind, the gain is 64%. But also on the standard English WS353 dataset (Finkelstein et al., 2002), we get a 26% better correlation with the human ratings. On German, the CLBL++ obtains correlations up to three times stronger than the baseline, and 39% better for French (Table 4.8).

---

[14] Source: http://metaoptimize.com/projects/wordreprs

(Luong et al., 2013)		*Our models*	
HSMN	2	CLBL	18
HSMN+csmRNN	22	CLBL++	**30**
C&W	27	$-$*compose*	20
C&W+csmRNN	**34**		

Table 4.7: Word-pair similarity task (English), showing Spearman's $\rho\times 100$ for the correlation between model scores and human ratings on the English RW dataset. The csmRNNs benefit from initialisation with high quality pre-existing word embeddings, while our models used random initialisation. HSMN refers to the embeddings from (Huang et al., 2012); C&W to those from (Collobert and Weston, 2008).

Datasets:[15]	WS		Gur	RG		ZG
	EN	ES	DE	EN	FR	DE
HSMN	63	–	–	63	–	–
+csmRNN	65	–	–	65	–	–
CLBL	32	26	36	**47**	33	6
CLBL++	39	**28**	**56**	41	**45**	**25**
$-$*compose*	**40**	27	44	41	41	23
# pairs	353		350	65		222

Table 4.8: Word-pair similarity task in multiple languages, showing Spearman's $\rho\times 100$ and the number of word pairs in each data set. As benchmarks, we include the best results from Luong et al. (2013), who relied on more data and pre-existing embeddings not available in all languages. In the penultimate row our model's ability to compose vectors for OOV words is suppressed.

4.4.6 Qualitative analysis of morpheme vectors

A visualisation of the English morpheme vectors (Figure 4.7) suggests the model captured non-trivial morphological regularities: noun suffixes relating to persons (writ*er*, human*ists*) lie close together, while being separated according to number; negation prefixes share a region (un-, in-, mis-, dis-); and relational prefixes are grouped (surpa-, super-, multi-, intra-), with a potential explanation for their separation from inter- being that the latter is more strongly bound up in lexicalisations (*inter*national, *inter*section). Some clusters abstract over spelling variation (-ise vs. -ize), while others share similarity in terms of the stems they would attach to, e.g.

---

[15] ES WS353 (Hassan and Mihalcea, 2009); Gur350 (Gurevych, 2005); RG65 (Rubenstein and Goodenough, 1965) with FR (Joubarne and Inkpen, 2011); ZG222 (Zesch and Gurevych, 2006).

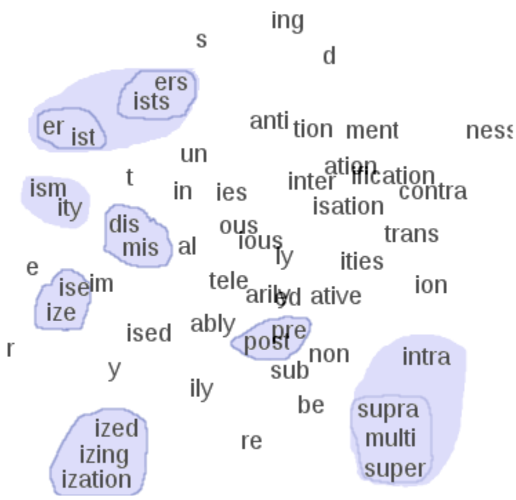


Figure 4.7: Visualisation of the morpheme vectors learnt by CLBL++ on English data, using t-SNE (van der Maaten and Hinton, 2008) for dimensionality reduction. Shading added for emphasis.

global(-ize,-izing,-ization).

4.4.7 Application to machine translation

The final aspect of our evaluation focuses on the integration of class-decomposed log-bilinear models into a machine translation system. To the best of our knowledge, this is the first study to investigate large vocabulary normalised DLMs inside a decoder when translating into a range of morphologically rich languages. We consider 5 language pairs, translating from English into Czech, German, Russian, Spanish and French.

Aside from the choice of language pairs, this evaluation diverges from Vaswani et al. (2013) by using normalised probabilities, a process made tractable by the class-based decomposition and caching of context-specific normaliser terms. Vaswani et al. (2013) relied on unnormalised model scores for efficiency, but do not report on the performance impact of this assumption. It is not clear how reliable an unnormalised language model score is as a feature that must help the translation model discriminate between alternative hypotheses, as frequency-effects would give rise to scores of very different scales.

We use *cdec* (Dyer et al., 2010, 2013) to build symmetric word-alignments and

	MKN	CLBL	CLBL++
EN→CS	12.6 (0.2)	13.2 (0.1)	**13.6** (0.04)
DE	15.7 (0.1)	**15.9** (0.2)	15.8 (0.4)
ES	24.7 (0.4)	25.5 (0.5)	**25.7** (0.3)
FR	24.1 (0.2)	24.6 (0.2)	**24.8** (0.5)
RU	15.9 (0.2)	16.9 (0.3)	**17.1** (0.1)
CS→EN	19.8 (0.4)	**20.4** (0.4)	**20.4** (0.5)

Table 4.9: BLEU scores (case-insensitive) obtained on newstest2013, with standard deviation over 3 MERT runs given in parentheses. The two right-most columns use the listed DLM as a feature in addition to the MKN feature, i.e. these MT systems have at most 2 LMs. Language models are from Table 4.6 (top).

extract rules for hierarchical phrase-based translation (Chiang, 2007). Our baseline system uses a standard set of features in a log-linear translation model. This includes a baseline 4-gram MKN language model, trained with *SRILM* (Stolcke, 2002) and queried efficiently using *KenLM* (Heafield, 2011). The DLMs are integrated directly into the decoder as an *additional* feature function, thus exercising a stronger influence on the search than in n-best list rescoring. Translation model feature weights are tuned with MERT (Och, 2003) on newstest2012.

Table 4.9 summarises our translation results. Inclusion of the CLBL++ language model feature outperforms the MKN-only baseline systems by 1.2 BLEU points for translation into Russian, and by 1 point into Czech and Spanish. The EN→DE system benefits least from the additional DLM feature, despite the perplexity reductions achieved in the intrinsic evaluation. In light of German's productive compounding, it is conceivable that the bilingual coverage of that system is more of a limitation than the performance of the language models.

On the other languages, the CLBL adds 0.5–1 BLEU points over the baseline, whereas additional improvement from the additive representations lies within MERT variance except for EN→CS.

4.5 Related work

Factored language models (FLMs) have been used to integrate morphological information into both discrete n-gram LMs (Bilmes and Kirchhoff, 2003) and DLMs (Alexandrescu and Kirchhoff, 2006) by viewing a word as a set of factors. Alexandrescu and Kirchhoff (2006) demonstrated how factorising the representations of context-words can help deal with out-of-vocabulary words, but they did not evaluate the effect of factorising output words and did not conduct an extrinsic evaluation. The FLM notion has since been used to integrate part-of-speech information into recurrent neural-network language models for ASR (Wu et al., 2012) and code-switching (Adel et al., 2013).

A variety of strategies have been explored for bringing DLMs to bear on machine translation. Rescoring lattices with a DLM proved to be beneficial for ASR (Schwenk, 2004) and was subsequently applied to translation (Schwenk et al., 2006; Schwenk and Koehn, 2008), reaching training sizes of up to 500m words (Schwenk et al., 2012). For efficiency, this line of work relied heavily on small "shortlists" of common words, by-passing the DLM and using a back-off n-gram model for the remainder of the vocabulary. Using unnormalised DLMs during first-pass decoding has generated improvements in BLEU score for translation into English (Vaswani et al., 2013).

More indirect approaches have approximated DLMs by sampling text from them and then training conventional n-gram models on that (Allauzen et al., 2013), or by casting the DLM directly as a back-off n-gram model through pruning (Wang et al., 2013), while Duh et al. (2013) used them for selecting relevant in-domain training data.

Vector representations have also been studied extensively within the area of distributional semantics (Turney and Pantel, 2010). There, vectors are typically constructed directly from the occurrence counts of words in a corpus, with each dimension of the vector space designating a specific context, for example. A variety of techniques have been considered for how to compose word vectors into representations of phrases or sentences (Mitchell and Lapata, 2010; Baroni and Zamparelli, 2010, *inter alia*), which have been extended upon to address the question of composing morpheme vectors into words by Lazaridou et al. (2013).

Other recent work has moved beyond monolingual vector-space modelling, incorporating phrase similarity ratings based on bilingual word embeddings as a translation model feature (Zou et al., 2013), or formulating translation purely in terms of distributed models (Kalchbrenner and Blunsom, 2013).

Accounting for linguistically derived information such as morphology (Luong et al., 2013; Lazaridou et al., 2013) or syntax (Hermann and Blunsom, 2013) has proved beneficial to learning vector representations of words. Our contribution is to create morphological awareness in a *probabilistic* language model.

Parameter tying within an LBL has also been used in domain adaptation (Xiao and Guo, 2013), where the representation space is subdivided into source, target and background domains.

4.6 Summary

We introduced a method for integrating morphology into probabilistic distributed language models. Our method has the flexibility to be used for morphologically rich languages (MRLs) across a range of linguistic typologies. Our empirical evaluation focused on multiple MRLs and different tasks. The primary outcomes are that (i) our morphology-guided DLMs improve intrinsic language model performance when compared to baseline DLMs and n-gram MKN models; (ii) word and morpheme representations learnt in the process compare favourably in terms of a word similarity task to a recent more complex model that used more data, while obtaining large gains on some languages; (iii) machine translation quality as measured by BLEU was improved consistently across six language pairs when using DLMs during decoding, although the morphology-based representations led to further improvements beyond the level of optimiser variance only for English→Czech. By demonstrating that the class decomposition enables full integration of a normalised DLM into a decoder, we open up many other possibilities in this active modelling space.

Chapter 5

Unsupervised learning of complex morphology

Chapter Abstract

The previous two chapters considered methods for integrating morphological information into two different types of LMs on the assumption that an external source of information is available. In this chapter, we go beyond sequence-based language modelling to consider how morphological information can be obtained in the first place using unsupervised methods. We introduce a novel approach to modelling concatenative and non-concatenative morphology jointly. This chapter is an extension of material originally published as (Botha and Blunsom, 2013).

Past work on unsupervised morphology learning has overwhelmingly considered *concatenative morphology*, where surface words can be analysed into a sequence of continuous morphemes or string segments. Consecutive instances of the *Morpho-Challenge* shared-task boosted contributions to unsupervised learning in this regard (Kurimo et al., 2010). But another important aspect of unsupervised learning of surface-level morphology concerns *non-concatenative* processes, where the morphemes of a word do not necessarily coincide with contiguous strings but are instead dispersed throughout the word. This type of morphology has received much less attention within the realm of unsupervised learning—the survey by Hammarström and

Borin (2011) identifies only 10 of the 200+ studies within its purview as targeting non-concatenative morphology.

Concatenative and non-concatenative morphological processes do not function in isolation. The classical example is Arabic, which uses non-concatenative processes to derive noun and verb stems, onto which various clitics and affixes attach in a concatenative fashion. As another example, Tagalog verb conjugation makes extensive use of infixing and circumfixing in its conjugation of verbs.[1] In both these examples, a segmentational approach recovers some of the building blocks of words, but naturally misses out on certain regularities, e.g. the notion that the Arabic stems *katabat* ('she wrote') and >*akotubu* ('I write') are bound together by the shared, discontiguous root *ktb*; or that two Tagalog words ***s***_um_***ulat*** and ***s***_in_***ulat***_an_ (different conjugations of 'wrote') share the same root, *sulat*. An unsupervised model that does not have the expressivity to capture these kind of relations therefore cannot benefit from the strong signals they encode. Modelling non-concatenative morphology is thus a way of extracting more value from the unannotated text by reducing sparsity.

The primary contribution of this chapter is the formulation of a probabilistic model that jointly captures concatenative and non-concatenative morphology, as exemplified above. We extend adaptor grammars (Johnson et al., 2007a) to a mildly context-sensitive grammar formalism that easily captures both kinds of surface-level morphology. We argue that the general approach is suitable to a variety of settings, but focus the application on Semitic morphology.

The structure of the chapter is as follows: The next section provides more background on Semitic morphology. The subsequent section reviews existing computational treatments of Semitic morphology and explains the relative merits of an unsupervised approach. The modelling work follows in three parts: First, we cover background material for unsupervised learning with adaptor grammars. Secondly, we present the more expressive grammar formalism and demonstrate its application

---

[1] Tagalog is spoken in the Philippines.

to Semitic morphology. Finally, we report empirical evaluation results on Arabic and Hebrew.

5.1 Basic overview of Semitic morphology

The Semitic language family is a suitable test-bed for the modelling approach we propose in light of the fundamentally non-concatenative root-templatic morphology.[2] Member languages that are widely spoken today include Arabic, Hebrew, Amharic and Tigrinya.

In contrast to many languages where the word stem is the most elementary semantic building block, Semitic stems are derived from semantic root morphemes and grammatical templates which combine in a non-concatenative way.[3]

The roots consist of between two and five *radicals* which are exclusively consonants. Roots are often referred to by the number of radicals they have. *Triliteral* roots featuring three radicals are the dominant kind. Because of the dominance of triliteral roots, existing modelling approaches are often limited to this class. An aim of the model we introduce is that it should easily encode roots of different lengths.

Templates encode grammatical information, such as number or person. To a first approximation, verb and noun stems are derived by interspersing root radicals into templates. In Arabic, for example, *kitAb* ('book'), *kutub* ('books'), *katab* ('write') and *takAtab* ('correspond') all derive from the root *ktb*. The root cannot in general be identified as a continuous substring of the stem.

Such stems undergo further transformation to give rise to surface word forms. Various inflectional affixes and clitics attach to stems by concatenation to give rise to multi-segment surface words. A clitic embodies the syntactic properties of an

---

[2] Many different linguistic terms apply to this discussion. For simplicity's sake we do not dwell on their subtle distinctions and use whichever term is most convenient and sufficiently precise in a given context. "Root-templatic" morphology is also referred to as root-and-pattern morphology. The particular sense in which it is non-concatenative is also variously denoted as transfixation, intercalation and interdigitisation.

[3] Linguistic morphology distinguishes between the consonant-vowel skeleton (e.g. CVCVC) and the vocalism (e.g. i…a) (McCarthy, 1981). We use "templates" to refer to the combination.

independent word but nonetheless surfaces as a bound morpheme. Pronouns, prepositions, conjunctions and the definite article are all cliticised in Arabic, which means segmentation is challenging also when one ignores the non-concatenative element of the morphology. Indeed, one hypothesis we investigate is that an account of the underlying root has a disambiguating effect on segmentation (§5.8).

The following list shows typical examples of the aforementioned morphological processes,[4] for surface words arising from stems based on the root *ktb*:

- verb person: *katab* ('write') ⇒ *kataba* ('he wrote'), *katabat* ('she wrote'), *katabona* ('we wrote')
- pronominal enclitics: *katabato**hu*** ('she wrote **it**')
- tense proclitic: *>akotubu* ('I write') ⇒ ***sa**>akotubu* ('I **will** write')
- determiner: ***Al**kitAbi* ('**the** book')
- prepositional proclitics: ***bi**AlokitAbi* ('**for** the book')
- conjunction: ***wa**biAlokitAbi* ('**and** for the book')

5.1.1 Orthographic variety

Another central property of Semitic languages is that their orthographic systems are so-called *abjads*, which means the explicit writing of vowels is partly or completely optional. The examples given so far use the less common *vocalised* form, as it illustrates the morphological processes better. In standard orthography, a system of diacritic marks is used to indicate that vocalisation, especially in literary and religious texts. But the more dominant case is *unvocalised* text, which suppresses some vocalisation information. This can render individual words highly ambiguous so that their correct interpretation relies on sentential context. For example, the unvocalised word token *ktb* has at least 15 readings.[5] As the focus in this chapter is on

---

[4] See (Habash, 2010) for a more complete introduction to Arabic morphology.

[5] This example was obtained using the Xerox Arabic Morphological Analyzer, available at `https://open.xerox.com/Services/arabic-morphology`.

type-based morphological analysis it does not include solving this disambiguation problem.[6] However, the relevance of the orthographic variation is that it complicates computational approaches to dealing with Semitic text. A key test for an unsupervised model is whether it can effectively handle such variation without requiring manual modification. Aside from a usability standpoint, the advantage of such a model is that it could learn directly from text that is heterogeneous in its vocalisation.

5.1.2 Complex non-concatenative processes outside scope

The preceding characterisation of root-templatic derivation is a simplification. It excludes cases where the correspondence between the characters in the surface word and those in the constituent morphemes is not one-to-one. Such cases occur where deletion and insertion of characters happens during word formation. The first instance of this is weak roots, where one or more of the radicals do not appear in the derived stem. The second instance concerns gemination, where there is a lengthening of a consonant, e.g. the stem *kuttAb* (‘writers’) as derived from root *ktb*.

These phenomena are present in the data used for our empirical evaluation though not accounted for by the model we introduce. The reason for limiting the scope this way is that the probabilistic modelling would otherwise be significantly more complex. Instead, we focus on giving a unified probabilistic account of concatenative and non-concatenative processes purely at the level of surface strings.

5.2 Existing approaches

This section surveys the dominant existing computational treatments of Semitic morphology. The aim is to situate our unsupervised probabilistic approach among alternative approaches while considering their relative merits.

---

[6] For studies that make direct use of sentential context to address this ambiguity problem, see (Habash and Rambow, 2005; Poon et al., 2009; Lee et al., 2011).

5.2.1 Rule-based methods

There is an extensive literature on encoding linguistic rules explicitly in formal computational frameworks to perform automatic morphological analysis. The dominant strand of work uses finite-state transducers (FSTs). FSTs are naturally suited to concatenative morphology, given that concatenation is one of the fundamental operations in regular languages. To deal with non-concatenative phenomena, FSTs require careful augmentation (Beesley and Karttunen, 2003). Early efforts on both Arabic (Kataja and Koskenniemi, 1988) and Hebrew (Lavie et al., 1988) made extensions to the foundational two-level morphological framework (Koskenniemi, 1984) to cater for root-templatic morphology. More complete approaches have directly operationalised the influential theory of non-concatenative morphology proposed by McCarthy (1981), using multi-tape transducers that are able to process the root, template and vocalism of the Semitic stem separately (Kay, 1987; Kiraz, 2000). The *MAGAED* system extends this approach further and its creators argue that the non-concatenative handling of stems is essential to deal efficiently with the different Arabic dialects (Habash et al., 2005; Habash and Rambow, 2006). Other augmentations make use of registers to give FSTs some limited memory to handle short-term dependencies (Cohen-Sygal and Wintner, 2006), whereas Gasser (2009) used weighted FSTs with a unification operation over attribute-value pairs.

The FST-based approaches are broadly lexical in their view of inflection and derivation—the morphosyntactic properties of words are carried by lexical morphemes. A separate strand of work in Semitic computational morphology is based instead on *inferential-realisational* theories of morphology (Stump, 2001), where morphosyntactic properties are specified purely by rules. Finkel and Stump (2002) use this to formulate an account of Hebrew verb morphology using hierarchies of inherited rules. Smrž (2007) developed a complete account of Arabic morphology that falls within this theoretical class. His ElixirFM system makes use of a functional morphology framework (Forsberg and Ranta, 2004) to formulate a domain-specific language embedded in a functional programming language.

The aforementioned approaches in both strands of work share the property of encoding morphological analysis and generation. A more ad-hoc approach that is limited to analysis is that of the Buckwalter Arabic Morphological Analyser (BAMA Buckwalter, 2002). BAMA relies on lists of stems and affixes together with consistency constraints on the co-occurrence of different morphemes within a word. It ignores the non-concatenative derivation of stems, but its lexica, which also indicate root morphemes, have been influential in the development or evaluation of other methods (Darwish, 2002; Smrž, 2007; Rodrigues and Ćavar, 2007, *inter alia*).

These rule-based approaches typically perform exceptionally well at their task, given the level of targeted design involved. But the limitations are substantial: A significant amount of manual effort and expert linguistic knowledge is required to develop the methods described in this section. This includes having a precise linguistic framework to operationalise, or a sufficient amount of lexical information as in the case of BAMA. Consequently, transferring a given method or system to other dialects or languages within the Semitic family can be a big undertaking. Finally, even within their target language, the rule-based approaches that have a lexical component invariably offer limited coverage and require manual addition for application to new domains.

5.2.2 Supervised learning

Supervised learning of morphology allows some of the limitations of a rule-based approach to be overcome, and shifts the resource requirement away from linguistic expertise toward data. Common sources of supervision include pairs of inflected words, words labelled with their roots, and lexica of affixes. A learning component then has to infer parametrisations or parameter values (or both) from these labelled resources. The existing literature on Semitic morphology that falls into this category is limited and fragmented.

One relevant strand of work considers the simpler problem of predicting the Arabic broken plural (e.g. *kutub*) given a base form (e.g. *katab*). Clark (2001b, 2002) applies pair hidden Markov models (HMMs), which have two emission streams, to train stochastic FSTs from such word-pair data. The approach is well-motivated for non-concatenative phenomena such as Semitic stem derivation, but it does not perform well in evaluation.

A more common theme has been to leverage the aforementioned sources of supervision to construct methods for identifying the root morpheme of a given word. Darwish (2002) combines pre-existing Arabic resources with various heuristics and rudimentary statistical models to devise an analyser that outputs a ranked list of candidate roots for a word.

Daya et al. (2008) approached root identification from the perspective of character-based classification. They train off-the-shelf classifiers for each radical and consider various ways of combining their predictions. Their investigation demonstrates the feasibility of the supervised approach by obtaining F-scores in the eighties, while transferring it from Hebrew to Arabic required relatively little effort. The methods are however specific to triliteral roots and do not contain a segmentational component.

Boudlal et al. (2009) employed a first-order HMM with roots as hidden states to identify the root of a given word with an accuracy of 94%; they relied on greedy methods and dictionaries of templates, roots and affixes to limit the hidden state space.

5.2.3 Unsupervised learning

As stated in the introduction of this chapter, the existing work on unsupervised learning of non-concatenative morphology is much more limited than concatenative morphology, and this extends to Semitic morphology. We take "unsupervised" to be situations where instances of the expected output of a method are not supplied during its training.

Probabilistic generative models constitute an attractive approach in this category as they allow high-level intuitions to be encoded in a coherent fashion. The only work we are aware of that directly targets root-templatic morphology in this sense is the non-parametric Bayesian model of Fullwood and O'Donnell (2013). Their model generates a word from a root, template, and “residue”, which captures the non-root characters of the word. For example, *kitAbi* would arise through the combination of root(ktb), template(r-r-r-) and residue(iAi). Each of these categories is modelled by an independent PYP-distributed lexicon, where the base distributions support arbitrary length sequences. Their evaluation is limited to template identification accuracy, where it performs very well. Two main shortcomings of the model are the lack of affix modelling and the atomicity of the lexicon elements:

1. The model only targets stem-formation. Given a word and its inferred template and residue, one could in principle extract a segmentation into prefix, stem and suffix, but segmentation of multiple adjacent affixes is nonetheless precluded.
2. Lexicon elements are atomic in the sense that their characters are sampled independently from a uniform distribution over the alphabet. This means there is no aggregation of probability mass around the recurrence of, say, consonants in the root lexicon—the PYP only caches the full strings.

The approach we introduce in §5.7 was developed contemporaneously with Fullwood and O’Donnell (2013), but specifically overcomes these two shortcomings, while aiming to be more easily extensible.

Other probabilistic approaches targeting Semitic segmentation have exploited regularities beyond word types, such as using sentential context (Poon et al., 2009; Lee et al., 2011) or multi-lingual phrases (Snyder and Barzilay, 2008).

There are unsupervised methods that indirectly address root-templatic morphology without necessarily attempting to model the word formation process. Words sharing the same root can be clustered based on heuristic scoring of character tuples (de Roeck and Al-Fares, 2000), which is useful for information retrieval. Baroni et al. (2002) similarly acquire morphologically related word pairs (e.g. *Verkauf*, *Verkäufe*) from raw text using string-edit distances and distributional cues.[7] Their

---

[7] This is a variation of the knowledge-free method by Schone and Jurafsky (2000), which was affix-based and thus limited to concatenative morphology.

approach is in principle applicable to Semitic morphology since the string-edit operations suit root-templatic morphology, but they frame the method as a potentially useful precursor to morphological analysis and did not apply it to Semitic languages.

5.3 Unsupervised learning with CFGs

Our overall approach to the problem of learning morphology from unannotated text is to construct a model based on rewrite rules. This type of approach has been applied successfully to morphological segmentation (Johnson et al., 2007a,b; Johnson, 2008) and offers well-defined mechanisms for encoding prior intuitions about the morphology.

The following two sections develop the relevant background material for learning with context-free grammars, which we subsequently build on for learning non-concatenative phenomena.

5.3.1 Probabilistic context-free grammars

A context-free grammar $\mathcal{G}_{\text{CFG}} = (S, N, T, P)$ is specified by a set of non-terminal symbols N, terminal symbols T and rewrite rules P of the form $A \rightarrow \gamma$, where $A \in N$ and $\gamma \in (N \cup T)^+$. $S \in N$ is the start symbol. A string $w \in T^+$ in the implied context-free language is derived from the start symbol S through a sequence of rule applications $r_1, \ldots, r_k \in P$ that expand non-terminal symbols until no further expansions are possible.

This symbolic grammar can be cast as a generative probabilistic model, known as a probabilistic context-free grammar (PCFG), by defining a probability distribution over each subset $P_A \subset P$ that contains the rules rewriting a symbol $A \in N$. That is, each rule $r \in P_A$ has an associated real value θ_r such that $0 \leq \theta_r \leq 1$ and $\sum_{r \in P_A} \theta_r = 1$. The stochastic procedure for generating a string from the model follows the same derivation pattern outlined above, but samples the rule r to use for expanding a given non-terminal A from a categorical distribution parametrised by θ. The key point is that this assumes conditional independence among expansions given a fixed $\mathcal{G}_{\text{CFG}}$, so that the joint probability of a string w and a derivation $R = r_1, \ldots, r_k$ is $P(w, R|\theta) = \prod_{r \in R} \theta_r$.

Two learning problems thus become apparent—what are the rules in P, and what are their probabilities θ? The former problem is challenging to pursue in a completely unsupervised fashion (cf. Stolcke and Omohundro, 1994; Clark, 2001a). If one assumes a set of rules is given, the second issue of learning their probability distributions can be approached through maximum likelihood estimation (Lari and Young, 1990; Goodman, 1998) or in a parametric Bayesian fashion by introducing a Dirichlet prior over θ (Johnson et al., 2007b).

5.3.2 Context-free adaptor grammars

Adaptor grammars (Johnson et al., 2007a) provide an elegant solution to both learning problems mentioned above. The core idea of adaptor grammars is that whole sub-trees are memorised and can be reused when generating strings from a probabilistic grammar. This relaxes the independence assumptions governing non-terminal expansion in the generative process of a PCFG to better match the pattern that non-trivial constructions in language re-occur across different instances. For example, the same stem would occur in many different surface word forms. Likewise, a given discontiguous Hebrew root would occur across different stems, which is the kind of regularity we intend to capture with our extension of adaptor grammars beyond context-free rules (§5.6.2).

The tendency for a sub-tree fragment to be reused is governed by the choice of adaptor function. We follow earlier applications (e.g. Johnson et al., 2007a; Huang et al., 2011) and use the Pitman-Yor process (PYP) as adaptor function (Pitman, 1995; Pitman and Yor, 1997). As with the language models discussed in chapter 3, the PYP's extensibility via its base distribution and its power-law behaviour make it suitable for modelling distributions over trees.

5.3.2.1 Formalism

A Pitman-Yor context-free adaptor grammar (PY-CFAG) is specified as a tuple $\mathcal{G} = (\mathcal{G}_{\text{PCFG}}, M, \boldsymbol{a}, \boldsymbol{b}, \boldsymbol{\alpha})$, where $\mathcal{G}_{\text{PCFG}}$ is a PCFG as defined before and $M \subseteq N$ is a set of *adapted non-terminals*. The vectors $\boldsymbol{a}$ and $\boldsymbol{b}$, indexed by the elements of M, are the discount and concentration parameters for each adapted non-terminal, with $a \in [0, 1]$ and $b \geq 0$. $\boldsymbol{\alpha}$ are parameters to Dirichlet priors on the rule probabilities θ.

PY-CFAG defines a generative process over a set of trees $\boldsymbol{T}$. Unadapted non-terminals $A' \in N \setminus M$ are expanded in the standard way of PCFGs, as described in the previous section.

For each adapted non-terminal $A \in M$ there is a cache C_A which stores terminating tree fragments having A as their root. We denote the tree fragment in C_A that corresponds to the i^{th} expansion of A as z_i. A sequence of indices $\mathbf{z}_i = z_1, \ldots, z_i$ therefore assigns each individual expansions of A to some tree fragment in the cache.

Given a cache C_A that has n previously generated trees comprising m unique trees each used $n_1, \ldots, n_m$ times (where $n = \sum_k n_k$), the tree fragment for the next expansion of A, z_{n+1}, is sampled conditional on the previous assignments $\boldsymbol{z}_n$ according to

$$z_{n+1} \mid \boldsymbol{z}_n, a, b \sim \begin{cases} \dfrac{n_k - a}{n + b} & \text{if } 1 \leq z_{n+1} \leq m \\[2ex] \dfrac{am + b}{n + b} & \text{if } z_{n+1} = m + 1, \end{cases} \tag{5.1}$$

where a and b are the elements of $\boldsymbol{a}$ and $\boldsymbol{b}$ corresponding to A.

1. The first case denotes the situation where a previously cached tree is reused for this $n + 1^{\text{th}}$ expansion of A. This expands A with a fully terminating tree fragment in a single step, meaning that none of the nodes descending from A in the tree being generated are subject to further expansion.

2. The second case by-passes the cache and expands A according to the rules P_A and rule probabilities θ_A of the base grammar $\mathcal{G}_{\text{PCFG}}$. This samples a subtree with root A from the PYP base distribution, a process during which other

caches C_B (s.t. $B \in M, B \neq A$) may come into play when expanding descendants of A; thus a PY-CFAG can define a hierarchical stochastic process.

Both cases eventually result in a terminating tree-fragment for A, which is then added to the cache, updating the counts n, n_{z_n+1} and potentially m.

The adaptation does not affect the string language of $\mathcal{G}_S$, but it maps the distribution over trees to one that is distributed according to the PYP.

5.3.2.2 Recursion

Context-free grammars may, in general, include recursion in their rewrite rules, either directly, e.g. X → X Y, or indirectly, e.g. X → Y Z and Z → X Q.

A PY-CFAG that features this type of recursion among its adapted non-terminals M complicates sampler-based inference, since it can involve repeated sampling from the same CRP without being able to update its counts correctly. In theory, a Metropolis-Hastings accept/reject step could be used as remedy but it is not obvious what proposal distribution would lead to acceptable rejection rates. Thus we follow Johnson et al. (2007a) and restrict the adaptor grammars we use to exclude recursive rewrite rules among adapted non-terminals. (This is what the "$B \neq A$" condition in step 2 of the previous section alludes to.)

5.4 An idealised grammar for learning morphology

A unifying theme in this thesis is to show that specific though high-level intuitions about morphology can be encoded in probabilistic models to improve their quality and capabilities. The high-level intuition we want to employ here is that morphological derivation and inflection are often non-concatenative.

Context-free adaptor grammars (§5.3.2) are effective at modelling concatenative morphology (Johnson et al., 2007a; Johnson, 2008). They offer the convenience of being able to capture the essence of phenomena ranging from single-slot inflection to agglutinative derivation by writing down a few context-free rewrite rules:

$$\text{Word} \rightarrow (\text{Pre}^{*}\ \text{Stem Suf}^{*})^{+} \tag{5.2}$$

$$\text{Stem} \,|\, \text{Pre} \,|\, \text{Suf} \rightarrow \text{Morph} \tag{5.3}$$

$$\text{Morph} \rightarrow (\textit{terminal strings}) \tag{5.4}$$

Our objective is to be able to specify additional rewrite rules of the following kind that would capture non-concatenative phenomena as exemplified:

$$\text{Stem} \rightarrow \textbf{intercal}\,(\text{Root}, \text{Template}) \tag{5.5}$$

e.g. Arabic derivation *k·t·b* + *i·a* ⇒ *kitAb* (‘book’)

$$\text{Stem} \rightarrow \textbf{infix}\,(\text{Stem}, \text{Infix}) \tag{5.6}$$

e.g. Tagalog *sulat* (‘write’) ⇒ ***s**um**ulat*** (‘wrote’)

$$\text{Stem} \rightarrow \textbf{circfix}\,(\text{Stem}, \text{Circumfix}) \tag{5.7}$$

e.g. Indonesian *percaya* (‘to trust’) ⇒ ***ke**percaya**an*** (‘belief’)

The bold-faced “functions” combine the potentially discontiguous yields of the argument symbols into single contiguous strings, e.g. **infix**(s·ulat, um) produces stem *sumulat*.

The question is how to express the multi-argument functions appearing in the idealisation above in terms of a formal rewrite grammar that can function as an adaptor grammar. We propose the use of a *mildly context-sensitive grammar* formalism (Joshi, 1985) that will allow for a concrete definition of **intercal** that is a formal rewrite rule. The more powerful grammar formalism retains the modelling convenience, and is notably consistent with the universal *morphological rule constraint* proposed by McCarthy (1981, p. 405), whereby “morphological rules must be context-sensitive rewrite rules affecting no more than one segment at a time, and no richer type of rule is permitted in the morphology”. It will allow a discontiguous string like k·t·b to be dominated by a single non-terminal node in a parse tree, which is very useful for probabilistic modelling.

5.5 Simple range concatenating grammars

We apply *simple Range Concatenating Grammars* (SRCGs) (Boullier, 2000) to parse contiguous and discontiguous morphemes from an input string. These grammars are mildly context-sensitive, forming a superset of context-free grammars that retains parsing time-complexity that is polynomial in the input length.[8] An SRCG-rule operates on vectors of ranges over the input, in contrast to the way a CFG-rule operates on spans (single ranges). In this way a non-terminal symbol in an SRCG (CFG) derivation can dominate a subset (substring) of terminals in an input string.

5.5.1 Formalism

An SRCG $\mathcal{G}$ is a tuple (N, T, V, P, S), with finite sets of non-terminals (N), terminals (T) and variables (V), with a start symbol $S \in N$. A rewrite rule $p \in P$ of rank $r = \rho(p) \geq 0$ has the form

$$A(\alpha_1, \ldots, \alpha_{\psi(A)}) \rightarrow B_1(\beta_{1,1}, \ldots, \beta_{1,\psi(B_1)}) \ldots B_r(\beta_{r,1}, \ldots, \beta_{r,\psi(B_r)}),$$

where each $\alpha \in (T \cup V)^*$, each $\beta \in V$, and $\psi(A)$ is the number of arguments a non-terminal A has, called its *arity*. By definition, the start symbol has arity 1. Any variable $v \in V$ appearing in a given rule must be used exactly once on each side of the rule. Terminating rules are written with ϵ as the right-hand side and thus have rank 0.

A *range* is a pair of integers (i, j) denoting the substring $w_{i+1} \ldots w_j$ of a string $w = w_1 \ldots w_n$. A non-terminal becomes *instantiated* when its variables are bound to ranges through substitution. Variables within an argument imply concatenation and therefore have to bind to adjacent ranges.

An instantiated non-terminal A' is said to derive ϵ if the consecutive application of a sequence of instantiated rules rewrite it as ϵ. A string w is within the language

---

[8] Our formulation is in terms of SRCGs, which are equivalent in power to linear context-free rewrite systems (Vijay-Shanker et al., 1987) and multiple context-free grammars (Seki et al., 1991), all of which are weaker than (non-simple) range concatenating grammars (Boullier, 2000). A good overview of these formalisms and their parsing complexity is given by Kallmeyer (2010).

defined by a particular SRCG iff the start symbol S, instantiated with the exhaustive range $(0, w_n)$, derives ϵ.

An important distinction with regard to CFGs is that, due to the instantiation mechanism, the ordering of non-terminals on the right-hand side of an SRCG rule is irrelevant, i.e. $A(ab) \rightarrow B(a)C(b)$ and $A(ab) \rightarrow C(b)B(a)$ are the same rule.[9] Consequently, the isomorphisms of any given SRCG derivation tree all encode the same string, which is uniquely defined through the instantiation process.

5.5.2 Application to root-templatic morphology

A fragment of the idealised grammar schema from the previous section (§5.4) can be rephrased as an SRCG by writing the rules in the newly introduced notation, and supplying a definition of the **intercal** function as simply another rule of the grammar:

$$\text{Word}(abc) \rightarrow \text{Pre}(a)\ \text{Stem}(b)\ \text{Suf}(c)$$

$$\text{Stem}(abcde) \rightarrow \text{Root}(a, c, e)\ \text{Template}(b, d),$$

where $a, b, c, d, e \in V$. An instantiation of the latter rule with $w = \text{kitAb}$ is

$$\begin{aligned}\text{Stem}(\langle 0..1\rangle, \langle 1..2\rangle, \langle 2..3\rangle, \langle 3..4\rangle, \langle 4..5\rangle) \rightarrow\ & \text{Root}(\langle 0..1\rangle, \langle 2..3\rangle, \langle 4..5\rangle) \\ & \text{Template}(\langle 1..2\rangle, \langle 3..4\rangle)\end{aligned}$$

Given an appropriate set of grammar rules (as we present in §5.7), we can parse an input string to obtain a tree as shown in Figure 5.1. The overlapping branches of the tree demonstrate that this grammar captures something a CFG could not. From the parse tree one can read off the word's root morpheme and the template used.

5.5.3 Time complexity and tractability

Although SRCGs specify mildly context-sensitive grammars, each step in a derivation is context-free—a node's expansion does not depend on other parts of the tree. This property implies that a recognition or parsing algorithm can have a worst-case

---

[9] Certain ordering restrictions over the variables *within* an argument need to hold for an SRCG to indeed be a *simple* RCG (Boullier, 2000).

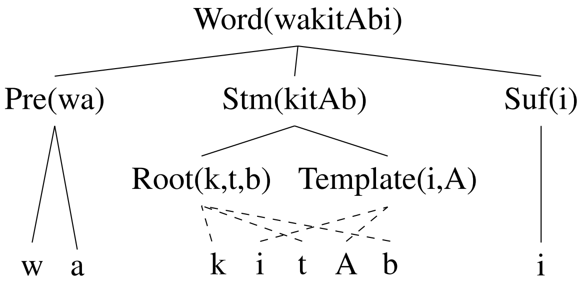


Figure 5.1: Example derivation for *wakitAbi* (and my book) using the SRCG fragment from §5.5.2. CFGs cannot capture such crossing branches.

time complexity that is polynomial in the input length n. The asymptotic expression is $\mathcal{O}(n^{(\rho+1)\psi})$ for arity ψ and rank ρ, which reduces to $\mathcal{O}(n^{3\psi})$ for a binarised grammar (Rodríguez and Satta, 2009; Kallmeyer, 2010, ch. 7); the formal basis for this result traces back to analyses by Seki et al. (1991); Boullier (2000), *inter alia*. We briefly state the intuition. For a binarised grammar, the worst-case occurs for a rule $A(\alpha_1, \ldots, \alpha_\psi) \to B(\beta_1, \ldots, \beta_\psi)\ C(\gamma_1, \ldots, \gamma_\psi)$, with each α_i consisting of two variables $u_i v_i$. In a bottom-up parser of SRCG (Kallmeyer, 2010), ψ ranges, hence 2ψ indices into the input string, have to be tracked for each right-hand side item. The adjacency in the arguments of A implies that the end point of the range bound to u_i is the same as the starting point of the range bound to v_i, so that the total number of *independent* indices are 3ψ, each ranging up to n.

To capture the maximal case of a stem comprising k discontiguous templatic character segments and $k-1$ interspersed root characters would require a grammar that has arity $\psi = k$. For Arabic, which has up to quadriliteral roots, hence $k \leq 5$, the time complexity would be at most $\mathcal{O}(n^{15})$. This is a daunting proposition for parsing, but we are careful to set up our application of SRCGs in such a way that this is not too big an obstacle:

Firstly, our grammars are defined over the characters that make up a word, and not over words that make up a sentence. As such, the input length n would tend to be shorter than when parsing full sentences from a corpus.

Secondly, we do type-based morphological analysis, a view supported by evidence from Goldwater et al. (2006), so each unique word in a dataset is only parsed once with a given grammar. The set of word types attested in the data sources of interest here is fairly limited, typically in the tens of thousands. For these reasons, our parsing and inference tasks turn out to be tractable despite the high time-complexity.

As a side remark, note that by refactoring rules the arity and rank can be balanced in a way that is optimal for parsing time-complexity, as characterised by Gildea (2010). It is thus conceivable that the appropriate refactoring of our grammars could bring down the complexity from what is given above.

5.6 Simple range concatenating adaptor grammars

We have argued that generative grammars are convenient for expressing linguistic intuitions of word formation (§5.4) and demonstrated how SRCGs can be used to encode root-templatic morphology (§5.5.2). In this section, we tie together SRCGs and adaptor grammars to obtain a framework capable of learning non-concatenative morphology in an unsupervised fashion.

5.6.1 Probabilistic SRCG

The probabilistic extension of SRCGs is similar to the probabilistic extension of CFGs to PCFGs, and has been used in other guises (Kato et al., 2006; Maier, 2010). Each rule $r \in P$ of the SRCG $\mathcal{G}_S$ (as defined in §5.5.1) has an associated probability θ_r such that $\sum_{r \in P_A} \theta_r = 1$. A random string in the language of the grammar can then be obtained through a generative procedure that begins with the start symbol S and iteratively expands it until deriving ϵ: At each step for some current symbol A, a rewrite rule r is sampled randomly from P_A in accordance with the distribution over rules and used to expand A. This procedure terminates when no further expansions are possible. Of course, expansions need to respect the range concatenating and ordering constraints imposed by the variables in rules. The expansions imply a chain of variable bindings going down the tree, and instantiation happens only when rewriting into ϵs but then propagates back up the tree.

The probability $P(w,t)$ of the resulting tree t and terminal string w is the product $\prod_r \theta_r$ over the sequence of rewrite rules used.

The invariance of SRCGs trees under isomorphism does not make the probabilistic model deficient. It adds a source of spurious ambiguity that we pre-empt in practice by requiring that grammar rules are specified in a canonical way which ensures a one-to-one correspondence between the order of nodes in a tree and of terminals in the yield.

5.6.2 PYSRCAG

A Pitman Yor simple Range Concatenating Adaptor Grammar (PYSRCAG) is specified as a tuple $\mathcal{G} = (\mathcal{G}_S, M, \mathbf{a}, \mathbf{b}, \boldsymbol{\alpha})$, where $\mathcal{G}_S$ is a probabilistic SRCG, M is the set of adapted non-terminals and the hyperparameters are analogous to those of the PY-CFAG (§5.3.2).

The inference procedure under our model is very similar to that of PY-CFAGs, so we restate the central aspects here but refer the reader to the original article by Johnson et al. (2007a) for further details. First, one may integrate out the adaptors to obtain a single distribution over the set of trees generated from a particular non-terminal. Thus, the joint probability of a particular sequence $\boldsymbol{z}$ for the adapted non-terminal A with cached counts $(n_1, \ldots, n_m)$ is

$$PY(\boldsymbol{z}|a,b) = \frac{\prod_{k=1}^{m} (a(k-1)+b) \ \prod_{j=1}^{n_k-1} (j-a)}{\prod_{i=0}^{n-1} (i+b)}. \tag{5.8}$$

Taking all the adapted non-terminals into account, the joint probability of a set of full trees $\boldsymbol{T}$ under the grammar $\mathcal{G}$ is

$$P(\boldsymbol{T}|\boldsymbol{a},\boldsymbol{b},\boldsymbol{\alpha}) = \prod_{A\in M} \frac{\mathrm{B}(\boldsymbol{\alpha}_A + \boldsymbol{f}_A)}{\mathrm{B}(\boldsymbol{\alpha}_A)} PY(\boldsymbol{z}(\boldsymbol{T})|\boldsymbol{a},\boldsymbol{b}), \tag{5.9}$$

where $\boldsymbol{f}_A$ is a vector of the usage counts of rules $r \in P_A$ across $\boldsymbol{T}$, and B is the Euler beta function.

The posterior distribution over a set of *strings* $\boldsymbol{w}$ is obtained by marginalising (5.9) over all trees that have $\boldsymbol{w}$ as their yields. This is intractable to compute directly, so instead we use MCMC techniques to obtain samples from that posterior using a component-wise Metropolis-Hastings sampler.[10] The sampler works by visiting each string w in turn and drawing a new tree for it under a proposal grammar $\mathcal{G}_Q$ and randomly accepting that as the new analysis for w according to the Metropolis-Hastings accept-reject probability. As proposal grammar, we use the analogous approximation of our $\mathcal{G}$ as Johnson et al. used for PCFGs, namely by taking a static snapshot $\mathcal{G}_Q$ of the adaptor grammar where additional rules rewrite adapted non-terminals as the terminal strings of their cached trees. Drawing a sample from the proposal distribution is then a matter of drawing a random tree from the parse chart of w under G_Q. The main distinction of the PYSRCAG therefore lies in the use of an SRCG parser to produce the chart.[11]

5.7 PYSRCAG model for root-templatic morphology

The previous sections have laid the ground-work for learning adaptor grammars that can cover discontiguous strings. This section provides the detail for how we apply PYSRCAGs to model Semitic morphology, covering both its concatenative and root-templatic aspects.

5.7.1 Words as concatenated morphemes

We start with a CFG-based adaptor grammar that models words as consisting of a stem and any number of prefixes and suffixes:[12]

$$\text{Word} \rightarrow \underline{\text{Pre}}^{*}\ \underline{\text{Stem}}\ \underline{\text{Suf}}^{*} \tag{5.10}$$

$$\underline{\text{Pre}} \mid \underline{\text{Stem}} \mid \underline{\text{Suf}} \rightarrow \text{Char}^{+} \tag{5.11}$$

---

[10] An alternative to MCMC is to do variational inference on top of the stick-breaking representation of the PYP (Cohen et al., 2010).

[11] We acknowledge the use of Mark Johnson's publicly available implementation of PY-CFAGs, which we extended as necessary for SRCGs.

[12] Adapted non-terminals are indicated by underlining and we use the following abbreviations: $\text{X} \rightarrow \text{Y}^{+}$ means one or more instances of Y and encodes the rules $\text{X} \rightarrow \text{Ys}$ and $\text{Ys} \rightarrow \text{Ys Y} \mid \text{Y}$. Similarly, $\text{X} \rightarrow \text{Y}^{*}\ \text{Z}$ allows zero or more instances of Y and encodes the rules $\text{X} \rightarrow \text{Z}$ and $\text{X} \rightarrow \text{Y}^{+}\ \text{Z}$. Further relabelling is added as necessary to avoid recursion among adapted non-terminals.

This fragment can be seen as building on the stem-and-affix adaptor grammar presented in (Johnson et al., 2007a) for morphological analysis of English, of which a later version also covers multiple affixes (Sirts and Goldwater, 2013). In the particular case of Arabic, multiple affixes are required to handle the attachment of particles and clitics onto base words.

5.7.2 Complex stems

In the preceding grammar, Stem is purely a sequence of characters without further cached sub-structure. Although there are certain word forms for which this is an appropriate view, this rule is obviously blind to the reuseable sub-structures inherent in root morphemes and templates.

Here, we make the logical extension of Stem to *complex stems*. To establish the necessary conventions, we start with a rule-set that is appropriate for modelling triliteral roots:

$$\underline{\text{Stem}}(abcdefg) \rightarrow \underline{\text{R3}}(b,d,f)\ \text{T4}(a,c,e,g) \tag{5.12}$$

$$\underline{\text{Stem}}(abcdef) \rightarrow \underline{\text{R3}}(a,c,e)\ \text{T3}(b,d,f) \tag{5.13}$$

$$\underline{\text{Stem}}(abcde) \rightarrow \underline{\text{R3}}(a,c,e)\ \text{T2}(b,d) \tag{5.14}$$

$$\underline{\text{Stem}}(abcd) \rightarrow \underline{\text{R3}}(a,c,d)\ \text{T1}(b) \tag{5.15}$$

$$\underline{\text{Stem}}(abc) \rightarrow \underline{\text{R3}}(a,b,c) \tag{5.16}$$

Some of these cases call for additional rules that permute the variables. For example, for handling other possible ways of forming a stem from a triliteral root and a single templatic character, rule (5.15) also has a variant $\underline{\text{Stem}}(abcd) \rightarrow \underline{\text{R3}}(a,b,d)$ $\text{T1}(c)$. In these and other rule-sets given subsequently, such permuted variants are suppressed but implied.

The justification for rule (5.16), which may look trivial, is that it is common in unvocalised text for the stem to be nothing more than the concatenation of the root characters. The inclusion of that rule is thus important for sharing statistical strength of a given root across all stems it may occur in, including ones that feature no templatic characters.

Note that although we provide the model with two sets of discontiguous non-terminals, R and T, we do not pre-specify their mapping onto terminal strings. No subdivision of the alphabet into vowels and consonants is hard-wired.

To complete the ruleset, a discontiguous non-terminal An is rewritten through recursion on its arity,[13]

$$\begin{aligned} \mathrm{A}n(v_1, \ldots, v_n) &\to \mathrm{A}l(v_1, \ldots, v_{n-1})\ \mathrm{Char}(v_n) && \textit{(recursive case; } 1 < l = n-1\textit{)} \\ \mathrm{A}1(v) &\to \mathrm{Char}(v) && \textit{(base case)} \end{aligned}$$

where $v_i \in V$ are variables and Char rewrites all terminals $t \in T$ as $\mathrm{Char}(t) \to \epsilon$.

5.8 Experiments

We evaluate our model on Modern Standard Arabic, Quranic Arabic and Hebrew. These languages are closely related in their morphology, and feature lexical cognates. But they are sufficiently different so that the transferral of rule-based morphological analysers from one to the other typically require manual intervention. A key question in this evaluation is therefore whether an appropriate instantiation of our adaptor grammars successfully generalises across these related languages.

We evaluate on three tasks: segmentation of a word into a linear sequence of morphemes, identification of a word's root, and the acquisition of lexica of stems, affixes and roots.

5.8.1 Evaluation data

Our models are unsupervised and therefore learn from raw text, but their evaluation requires annotated data as a gold-standard and these were derived as follows:

---

[13] Including the arity as part of the non-terminal symbol names forms part of our convention here to ensure that the grammar contains no cycles, a situation which would complicate inference under PYSRCAG.

Bw	Bw$'$	Root
wlAt$_{PRE}$ fHm$_{STM}$	walAta$_{PRE}$ foHam$_{STM}$	f·H·m
EArDy$_{STM}$	EAriDiyy$_{STM}$	E·r·D
w$_{PRE}$ jbA$_{STM}$ hm$_{SUF}$	wa$_{PRE}$ jabbA$_{STM}$ hum$_{SUF}$	j·b·n
fll$_{PRE}$ mkAtb$_{STM}$	falil$_{PRE}$ mukAtib$_{STM}$	k·t·b

Table 5.1: Example of words in our generated dataset Bw, shown in Buckwalter transliteration.

5.8.1.1 Arabic (MSA)

We created the dataset Bw by synthesising 50k morphotactically correct word types from the morpheme lexicons and consistency rules supplied with the Buckwalter Arabic Morphological Analyser (BAMA, Buckwalter, 2002).[14] This allowed control over the word shapes, which is important to focus the evaluation, while yielding reliable segmentation and root annotations. Bw has no vocalisation; we denote the corresponding *vocalised* dataset as Bw$'$. Example words are presented in Table 5.1.

5.8.1.2 Quranic Arabic

We extracted the roughly 18k word types from a morphologically analysed version of the Quran (Dukes and Habash, 2010). As an additional challenge, we left all given diacritics intact for this dataset, Qu$'$.

5.8.1.3 Hebrew

We leveraged the Hebrew *CHILDES* database as an annotated resource (Albert et al., 2013), specifically using both the adult and child utterances from the *Berman* and *Ravid* longitudinal corpora. 5k word types featuring at least one affix each were extracted to form our dataset Heb. For words marked as non-standard child language,

---

[14] We used BAMA version 2.0, LDC2004L02, and sampled word types having a single stem and at most one prefix, suffix or both, according to the following random procedure: Sample a shape according to (stem: 0.1, pre+stem: 0.25, stem+suf: 0.25, pre+stem+suf: 0.4). Sample uniformly at random (with replacement) a stem from the BAMA stem lexicon, and affix(es) from the ones consistent with the chosen stem. The BAMA lexicons contain elementary and compound affixes, so some of the generated words would permit a linguistic segmentation into multiple prefixes/suffixes. Nonetheless, we take as gold-standard segmentation precisely the items used by our generation procedure.

Dataset	Words	Stems	Roots	m/w	c/w
Bw	48428	24197	4717	2.3	6.4
Bw′	48428	30891	4707	2.3	10.7
Qu′	18808	12021	1270	1.9	9.9
Heb	5231	3164	492	2.1	6.7

Table 5.2: Corpus statistics in number of types, including average number of morphemes (m/w) and characters (c/w) per word type. The Roots column gives the number of distinct surface-realised roots of length 3 or 4.

we used the corrected forms provided by the database. Stressed and unstressed vowels were conflated to overcome some inconsistencies in the source data.

5.8.2 Model instantiations

We use as baselines adaptor grammars the context-free models that express morphemes as sequences of characters. We consider two variants that match the known characteristics of our datasets: **Concat** allows only up to one of each morpheme type in a word, while **MConcat** allows multiple prefixes and suffixes.

Using these concatenative grammars as starting points, we formulate more expressive grammars by adding rules that feature discontinuity. The canonical complex stem grammars aimed at triliteral roots are denoted as **Tpl** and **MTpl**, depending on the starting point grammar.

Informal inspection of the data revealed that cases exist where multiple characters intervene between root characters. We thus also experiment with a variant that allows the non-terminal T1 to be rewritten as up to three Char symbols. This model is denoted **Tpl3Ch**.

For efficiency reasons, the aforementioned models are limited to rules of arity at most 3. The grammars **Tpl+T4** and **TplR4** relax this constraint to include non-terminal categories T4 and $\underline{\text{R4}}$ & T4, respectively, in order to model quadriliteral roots as well.

These model definitions are given in more detail in Table 5.3.

5.8.2.1 Sampling details

Our PYSRCAGs have various hyperparameters. The PYP hyperparameters $\boldsymbol{a}$ and $\boldsymbol{b}$ are modelled by placing flat $\text{Beta}(1, 1)$ and vague $\text{Gamma}(10, 0.1)$ priors on them, respectively. Their values are then inferred using slice sampling (Neal, 2003; Johnson and Goldwater, 2009). The hyperparameters α are set to give symmetric Dirichlet prior distributions over the rule probabilities θ.

We start the sampler using batch initialisation, meaning the initial parse tree of each word is generated purely from the base distribution grammar and is not influenced by the PYP priors. Johnson and Goldwater (2009) found this approach to lead to higher posterior probabilities than initialising incrementally according to the model itself. Another option explored by that work is to include a type-based element into the sampler by occasionally resampling the cached trees themselves. This can improve mixing by updating the analyses of multiple words in a single step. Experimentation with these variations in initialisation and sampling was not central to our aim of applying SRCG-based adaptor grammars to root-templatic morphology, but it is conceivable that they may enable small improvements over our results for the same grammars we propose.

For each adaptor grammar, we collected 100 posterior samples after allowing 900 iterations of burn-in of the MCMC sampler described in §5.6.2. Morphological analyses were obtained from the collected samples in various ways, as set out per task in the following sections. To facilitate exposition, let $\mathcal{S}$ denote the $J = 100$ collected samples, where the j^{th} sample $\mathcal{S}^{(j)} = \{t_i^{(j)}\}$ consists of a parse tree $t_i^{(j)}$ for each word type w_i in the dataset, $1 \leq i \leq N$ and $1 \leq j \leq J$.

5.8.3 Task 1: Segmentation

5.8.3.1 Method

The segmentation of a word under our adaptor grammars is determined unambiguously by traversing the parse tree assigned to it. We split a word into morphemes

Concat =	$\begin{cases} \text{Word} \rightarrow \underline{\text{Pre}}\ \underline{\text{Stem}}\ \underline{\text{Suf}} \mid \underline{\text{Pre}}\ \underline{\text{Stem}} \mid \underline{\text{Stem}}\ \underline{\text{Suf}} \mid \underline{\text{Stem}} \\ \underline{\text{Pre}} \mid \underline{\text{Stem}} \mid \underline{\text{Suf}} \rightarrow \text{Char}^{+} \end{cases}$
MConcat =	**Concat** $\cup \ldots$ $\begin{cases} \text{Word} \rightarrow \text{Pres}\ \underline{\text{Stem}}\ \text{Sufs} \mid \text{Pres}\ \underline{\text{Stem}} \mid \underline{\text{Stem}}\ \text{Sufs} \\ \text{Pres} \rightarrow \underline{\text{Pre}}^{+} \\ \text{Sufs} \rightarrow \underline{\text{Suf}}^{+} \end{cases}$
Tpl =	$\begin{cases} \underline{\text{Stem}}(abcdef) \rightarrow \underline{\text{R3}}(a,c,e)\ \text{T3}(b,d,f) \\ \underline{\text{Stem}}(abcde) \rightarrow \underline{\text{R3}}(a,c,e)\ \text{T2}(b,d) \\ \underline{\text{Stem}}(abcd) \rightarrow \underline{\text{R3}}(a,c,d)\ \text{T1}(b) \\ \underline{\text{Stem}}(abc) \rightarrow \underline{\text{R3}}(a,b,c) \\ \underline{\text{R}}n \mid \text{T}n \rightarrow \text{(recursion described in §5.7.2)} \end{cases}$
Tpl+T4 =	**Tpl** $\cup \{\underline{\text{Stem}}(abcdefg) \rightarrow \underline{\text{R3}}(b,d,f)\ \text{T4}(a,c,e,g)\}$
Tpl3Ch =	**Tpl** $\cup \{\text{T1} \rightarrow \text{Char Char} \mid \text{Char Char Char}\}$
TplR4 =	**Tpl+T4** $\cup \{\underline{\text{Stem}}(abcdefgh) \rightarrow \underline{\text{R4}}(b,d,f,h)\ \text{T4}(a,c,e,g)\}$
MTpl =	**MConcat** $\cup$ **Tpl**

Table 5.3: Summary of model identifiers in terms of canonical grammar rules. Certain rule shapes involve a permutation in variables, but these are not shown explicitly.

along the boundaries of the substrings dominated by the non-terminals Stem, Pre, and Suf.

We use the maximum *a posteriori* (MAP) parse tree $\tilde{t}(w)$ of a word w, which is simply the tree that is assigned to that word most frequently in the collected posterior samples.[15] The joint probability of a word and tree can be factored as

$$P(w,t \mid \mathbf{a},\mathbf{b},\mathbf{w}) = P(w \mid t)\ P(t \mid \mathbf{a},\mathbf{b},\mathbf{w}), \tag{5.17}$$

where the first term is 1 if t yields w, and 0 otherwise; $\mathbf{a}$ and $\mathbf{b}$ are the hyperparameters, and $\mathbf{w}$ the observed data. The second term is the posterior distribution over trees (cf. Equation 5.9) and we can obtain an estimate of the probability of a tree under this distribution by aggregating over the collected samples $\mathcal{S}_i^{(1)},\ldots,\mathcal{S}_i^{(J)}$,

---

[15] An alternative method for obtaining a segmentation from the collected posterior samples is to use *maximum marginal decoding*, which marginalises out sub-structures which are irrelevant to segmentation (Johnson and Goldwater, 2009). In our experiments, this approach yielded very similar segmentation results as the MAP-based method, and we thus focus on the latter.

where i is such that $w_i = w$. That is,

$$P(t \mid \mathbf{a}, \mathbf{b}, \mathbf{w}) \approx \frac{1}{J} \sum_{j=1}^{J} \delta_t(t_i^j), \tag{5.18}$$

where δ is Kronecker-delta. Thus for a fixed word w, the MAP parse tree $\tilde{t}(w)$ is

$$\begin{aligned} \tilde{t}(w) &= \arg\max_t P(w, t \mid \mathbf{a}, \mathbf{b}, \mathbf{w}) && (5.19) \\ &= \arg\max_t P(t \mid \mathbf{a}, \mathbf{b}, \mathbf{w}) \quad (\text{fixed } w) && (5.20) \\ &\approx \arg\max_t \sum_{j=1}^{J} \delta_t(t_i^j). && (5.21) \end{aligned}$$

Segmentation quality is measured using the *segment border F1-score* (SBF, Sirts and Goldwater, 2013). This is a micro-averaged metric comparing the word-internal segmentation points of the predicted segmentation against the gold standard segmentation.[16]

As external baseline model we used **Morfessor** (Creutz and Lagus, 2007), which performs decently in unsupervised morphological segmentation of a variety of languages, but only handles concatenation.

5.8.3.2 Results

The introduction of complex-stem forming rules consistently brings large gains in segmentation quality compared to the purely concatenative baseline adaptor grammars (Table 5.4).

Across our data sets, the simplest templatic grammar featuring triliteral roots obtains an average relative increase in SBF of 21% (average SBF 47.8→58.0). These gains are not distributed uniformly across data sets and range from 12% to 29%. But regarding the question of whether a particular templatic adaptor grammar can deliver benefits across both Arabic and Hebrew, we conclude that the answer is in the positive for the segmentation task.

---

[16] We acknowledge the use of the evaluation script by Sharon Goldwater, obtainable from `http://homepages.inf.ed.ac.uk/sgwater/resources.html`.

	Bw$'$	Bw	Heb		Qu$'$
Morfessor	55.6	40.0	24.2	**Morfessor**	44.3
Concat	47.4	64.2	60.1	**MConcat**	19.6
Tpl	60.4	71.9	77.3	**MTpl**	22.5
Tpl3Ch	60.5	72.2	77.4	**MTpl3Ch**	25.7
Tpl+T4	64.5	71.6	77.1	**MTpl+T4**	24.8
TplR4	74.5	73.7	78.6		

Table 5.4: Segmentation quality in segment-border F-score (SBF).

Variation in SBF among the different variants of the triliteral grammars is small and mixed across data sets.

The vocalised Arabic data set Bw$'$ is most sensitive to changes in grammar expressivity. Introduction of the T4 non-terminal to the basic triliteral grammar increases SBF on Bw$'$ by 4 points, on top of which the most expressive grammar, **TplR4**, adds another 10 points. This amounts to a substantial total improvement of 57% by **TplR4** over the baseline **Concat**.

On the unvocalised Bw, grammar expressivity beyond **Tpl** brings smaller gains. This is consistent with the fact that Bw represents an easier segmentation task, since its unvocalised words are by definition shorter, on average, than those in Bw$'$, and cover fewer distinct contiguous morphemes.

When repeating the experiment for a version of the Hebrew training data that *includes* affixless words, SBF drops to the fifties. The overall trend that the templatic grammars improve segmentation remains, and the relative improvement is slightly larger in this setting.

A salient aspect of these results, when considered in conjunction with the root identification results to be presented in §5.8.4.3, is that templatic rules aid segmentation quality independent of root identification accuracy. In other words, our attempt at capturing discontiguous roots and templates benefits segmentation quality by allowing the model to capture discontiguous regularities at the sub-stem level, which is advantageous regardless of whether the captured elements actually correspond to the linguistically motivated analyses.

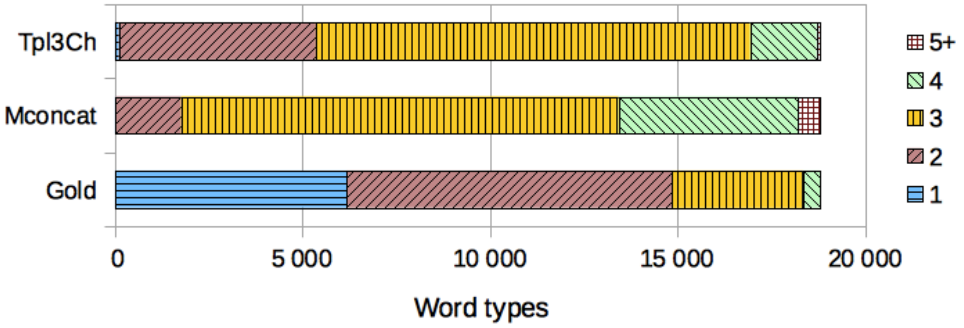


Figure 5.2: Distribution of number of morphemes per word for data set Qu', showing the large discrepancy between the gold standard and the induced segmentations.

5.8.3.3 Further analysis of Quran segmentation

The trend that templatic rules improved segmentation quality holds for the Quran data set (improvements of 15% to 31% relative to the baseline adaptor grammar), but absolute SBF for the adaptor grammars is disappointingly low for this data set.

Our analysis shows that all the evaluated adaptor grammars over-segmented heavily. As an illustration, the gold standard has on average 1.9 morphemes per word (m/w, Table 5.2), while the models hypothesise 2.8–3.2. The distribution of m/w is very different. The gold standard is dominated by words having one or two morphemes, giving a skewed distribution of m/w, while the adaptor grammars induce a more symmetric distribution where length three words dominate (Figure 5.2). If we evaluate only on the subset of words that have 3 or more morphemes according to the gold standard, segmentation quality is higher (**MConcat**: 32.7, **MTpl3Ch**: 38.6).

We investigated two hypotheses that could explain the low SBF scores on this data set. The first hypothesis is that the outcome is an artefact of our methodology for extracting segmentations from parse trees. As detailled in §5.8.3.1, we split words maximally along the elementary affix categories, Pre and Suf even though the grammars used for Qu' allow composite affixes. In Table 5.5, we designate this as the fine-grained grammar evaluated at the fine-grained level. An alternative is to split at the boundaries of the Stem category only and to ignore the internal structure

	Evaluation Coarse	 Fine
Concat (coarse grammar)	13.2	n/a
MConcat (fine grammar)	9.6	†19.6

Table 5.5: Effect of granularity choices in grammar design and evaluation when measuring segmentation quality on data set QU'. These results are for the strictly concatenative adaptor grammars. See §5.8.3.3 for details on "coarse" versus "fine" evaluation. †19.6 is highlighted as being the combination presented in Table 5.4.

of composite affixes. That is, evaluate the fine-grained grammar in a coarse-grained fashion. Finally, we can train a coarse-grained grammar (**Concat**) instead of a fine-grained one (**MConcat**), and evaluate coarsely.

The outcome of this two-way comparison, using only the concatenative baseline adaptor grammars for efficiency, is that all three valid combinations give low SBF scores (Table 5.5). We thus reject the hypothesis that these methodological choices by themselves explain the below-baseline performance of the adaptor grammars on QU'.

The second hypothesis we considered is that performance suffered because of a lack of hierarchical adaptation in the multi-affix adaptor grammars.[17] From Table 5.3, the fundamental word-forming rule for the multi-affix grammars is

$$\text{Word} \rightarrow \text{Pres}\ \underline{\text{Stem}}\ \text{Sufs},$$

where Pres and Sufs are *not* adapted (although $\underline{\text{Pre}}$ and $\underline{\text{Suf}}$ were adapted). We added adaptation to these categories, also introducing intermediate dummy rules to avoid ill-defined recursive adaptor grammar rules like $\underline{\text{Pres}} \rightarrow \underline{\text{Pres}}\ \underline{\text{Pre}}$. The segmentations induced by the resulting hierarchically adapted, concatenative grammar obtain an SBF of 22.2, marginally above the **MConcat** score of 19.6. The initial lack of hierarchical adaptation therefore affected segmentation quality adversely, but not sufficiently to fully explain the low segmentation quality on the Quran data set.[18]

---

[17] Sirts and Goldwater (2013) demonstrated that hierarchical adaptation can play a crucial role.

[18] The adaptation of affix non-terminals is nonetheless crucial. In preliminary experiments on BW, segmentation quality dropped by about 50% when using unadapted non-terminal categories Pre and Suf.

Given the low segmentation quality on the Quran, it is not included in the subsequent tasks.

5.8.4 Task 2: Root identification

The next evaluation task is that of obtaining the root morpheme, if any, given a surface word type as input. We describe two methods for root identification, followed by details of the baselines we compare to.

5.8.4.1 Root extraction method

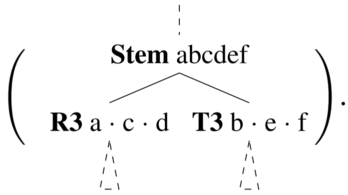
Our root-templatic adaptor grammars are fundamentally unsupervised. The only built-in bias they have for capturing roots under the R non-terminals instead of the T non-terminals is that we only adapted the former, in particular R3 and R4, as set out in Table 5.3. Root extraction *Method A* thus hypothesises as root morpheme the characters governed by R3 or R4 in the parse tree, depending on whether the evaluation is on triliterals or quadriliterals. A shortcoming of this method is that it would miss cases where the root happens to be governed by a T non-terminal, which is also licensed. The more lenient extraction *Method B* thus considers both types of non-terminals, possibly hypothesising two root candidates for a word. As with the segmentation task, we use the MAP parse tree $\tilde{t}(w)$ (cf. Equation 5.21) as input to these root extraction methods.[19] Figure 5.3 provides an example of these root extraction methods, and illustrates the evaluation calculation we define below.

Given an input word w, we denote the set of k-literal roots hypothesised as $\tilde{h}_k\big(\tilde{t}(w)\big) \in \big\{\emptyset, \{r_1^h\}, \{r_1^h, r_2^h\}\big\}$. We focus on $k \in \{3, 4\}$.

Similarly, we write the set of correct roots of the word w as $g_k(w) = \{r_j^g\}$. Roots that contain weak radicals are filtered out, since our models deal purely with surface-realised phenomena and thus cannot express such roots. Owing to this filtering and the fact that some word forms legitimately do not derive from a root, $g_k(w)$ can be empty. Our evaluation skips such words.

---

[19] The posterior distribution over roots conditioned on a word form, as estimated from the collected samples, is sharply peaked and in most cases uni-modal, so that a MAP estimate is reasonable.

Let w =abcdef and the inferred parse tree $\tilde{t}(w) =$ .

	Method A	Method B
	$\tilde{h}^A_3(\tilde{t}(w)) = \{acd\}$ $\tilde{h}^A_4(\tilde{t}(w)) = \emptyset$	$\tilde{h}^B_3(\tilde{t}(w)) = \{acd, bef\}$ $\tilde{h}^B_4(\tilde{t}(w)) = \emptyset$
Suppose $g_3(w) = \{acd\}$.	$\tilde{h}^A_3 \cap g_3 = \{acd\}$ $\therefore$ exact_match$_3(w) = 1$	$\tilde{h}^B_3 \cap g_3 = \{acd\}$ $\therefore$ exact_match$_3(w) = 1$
Suppose $g_3(w) = \{bef\}$.	$\tilde{h}^A_3 \cap g_3 = \emptyset$ $\therefore$ exact_match$_3(w) = 0$	$\tilde{h}^B_3 \cap g_3 = \{bef\}$ $\therefore$ exact_match$_3(w) = 1$

Figure 5.3: Illustration of the root extraction methods and scoring descibed in §5.8.4.1. For brevity, we present a single learning outcome for the word w (above table) and consider two alternative gold standard references $g_3(w)$ (bottom two rows).

The evaluation metric is exact match accuracy. A hypothesised root is counted as a match if it appears among the word's gold roots. When multiple roots are hypothesised we check if any one matches a gold root. To be precise, define the exact match score between hypothesised and gold standard roots for a word w as

$$\text{exact\_match}_k(w) = \begin{cases} 1 & \text{if } \tilde{h}_k\big(\tilde{t}(w)\big) \cap g_k(w) \neq \emptyset \\ 0 & \text{otherwise.} \end{cases} \tag{5.22}$$

The accuracy across the N' filtered test word types is $\frac{1}{N'} \sum_{i=1}^{N'} \text{exact\_match}_k(w_i)$.[20]

This metric is therefore recall-based. A precision-based evaluation would be overly harsh, since our root identification methods posit a root for most words. That is, the templatic adaptor grammars tend to rely heavily on the complex stem-forming rules, despite the presence of the simple rule, $\underline{\text{Stem}} \rightarrow \text{Char}^+$.

5.8.4.2 Baselines

We devised three straightforward randomised baseline methods to compare our model-based prediction accuracies. They all involve incremental deletion of characters

---

[20] These definitions are aimed at precise exposition, belying the fact that the ambiguity observed in our evaluation is very low; the vast majority of words have a single gold root, if any, and for hypothesised roots it is only Method B that occasionally posits more than one candidate.

from a word, chosen uniformly at random, until a string of k characters remain. The **uniform guesser** baseline applies this reduction method directly to the surface word form, while three further baselines operate on a constrained guessing space:

Stem-constrained: The gold standard stem of a word is taken as input, thereby establishing the difficulty of identifying a root under a perfect segmentation model.

Character-constrained: Starts with the surface word form, but strips out characters never appearing as roots in the data set. This baseline thus has external knowledge not encoded in our models, but has no sense of word structure.

Doubly-constrained: Uses both aforementioned constraints. This would often leave little ambiguity as to the identity of the root, and could be regarded as a benchmark rather than baseline.

Given the non-determinism of these baselines, we report the average of five repetitions of each.

5.8.4.3 Results

The results for root identification are given in Table 5.6 and Table 5.7.

The performance of the baseline methods confirms the intuition that root identification is harder in the presence of vocalisation.[21] It is consequently also for vocalised Arabic (BW$'$) that identification Method A performs poorly. The more liberal Method B succeeds in correctly identifying the triliteral root for up to one in five word types in BW$'$ that meet the evaluation conditions specified in §5.8.4.1. This is still low accuracy, but exceeds the character-constrained baseline. Since that baseline gauges the extent to which vocalisation complicates root identification, we conclude from this outcome that our models go beyond merely overcoming the presence of vocalisation. The difference is explained by the model's segmentational

---

[21] Hebrew morphology and orthography are less complicated than Arabic (Daya et al., 2008), which is why the baseline accuracies for the vocalised HEB are generally higher than for BW$'$.

	BW′	BW	HEB
Random baselines			
Uniform guesser	1.5	8.6	3.7
Stem-constrained	6.6	43.6	14.2
Character-constrained	10.3	10.3	30.3
Doubly constrained	41.7	45.2	79.9
Method A: Extract <u>R3</u>			
Tpl	0.8	66.6	72.9
Tpl3Ch	0.7	64.5	9.2
Tpl+T4	0.7	67.1	74.2
TplR4	0.9	64.0	69.5
Method B: Extract <u>R3</u> and T3			
Tpl	17.2	66.9	72.9
Tpl3Ch	0.8	64.5	9.2
Tpl+T4	19.8	67.4	74.7
TplR4	13.9	64.1	69.5
Evaluated (N')	27783	27494	2738

Table 5.6: Triliteral root identification accuracy. Exact matches of extracted and gold roots expressed as a percentage of the number of word types N' for which the true analysis contains a strong triliteral root.

	BW′	BW	HEB
Random baselines			
Uniform guesser	0.7	8.4	2.1
Stem-constrained	2.7	40.7	3.2
Character-constrained	10.5	10.8	15.8
Doubly constrained	39.4	40.6	66.3
TplR4			
Method A: Extract <u>R4</u>	0.1	10.4	0.0
Method B: Extract <u>R4</u> and T4	14.1	10.4	0.0
Evaluated (N')	2295	2276	19

Table 5.7: Quadriliteral root identification accuracy. Exact matches of extracted and gold roots expressed as a percentage of the number of word types N' for which the true analysis contains a strong quadriliteral root.

component. Nonetheless, the regularities leveraged in the process do not necessarily coincide with the linguistically correct roots.

On Hebrew and unvocalised Arabic, there is no meaningful difference between Methods A and B.

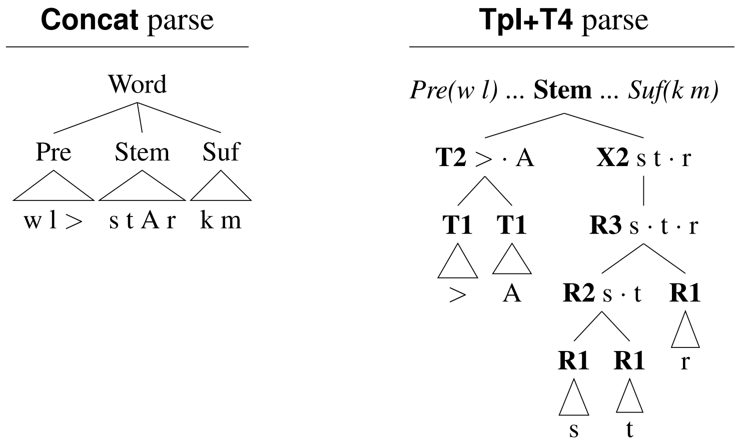
For triliteral root identification, the templatic grammar **Tpl+T4** that includes four-character templates performs best, with accuracies of 74% on Hebrew and 67% on unvocalised Arabic. In the latter data set there are about 9000 words for which the stem and root are equivalent; excluding those words from the evaluation decreases the accuracy from 67% to 57% accuracy. This is still substantially above the competitive Doubly constrained baseline. The fact that **Tpl+T4** outperforms **Tpl3Ch** shows that, for these two data sets, it is more important to capture templatic structures of the form VCVCVCV than to handle forms like CVVCVC. On Hebrew and unvocalised Arabic, this adaptor grammar jointly solves the task of segmentation and root identification to a fairly high standard.

Our model capable of quadrilteral root identification, **TplR4**, obtained accuracies in the region of the Character-constrained baseline for the two Arabic data sets. Closer analysis shows that this low accuracy is not due to worse segmentation performance on those words containing quadriliteral roots. Instead, we find that even if the model segments a word correctly, it often accounts for the stem sufficiently using the R3 or T3 categories, so that our method extracts no quadriliteral root. For cases where a quadriliteral root is hypothesised by our extraction methods, we find that they often match the gold root partially. In this sense, the exact match metric is relatively harsher than for evaluating triliteral root extraction.

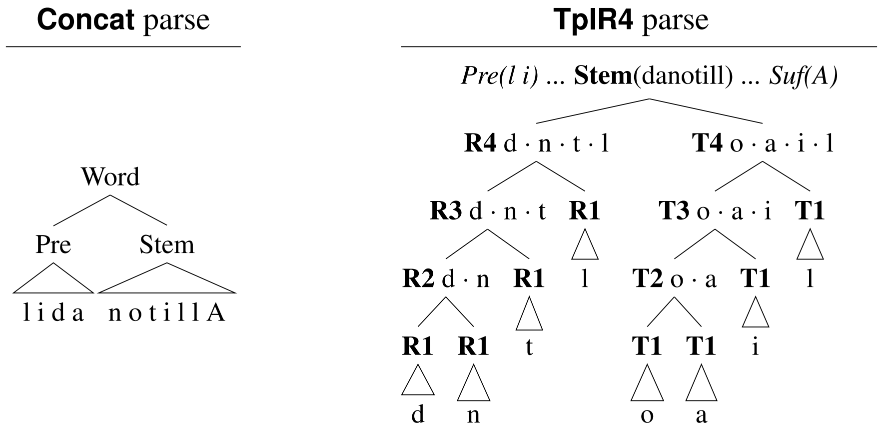
In summary, these results show that a given instantiation of our root-templatic adaptor grammars can correctly identify triliteral and quadriliteral roots of a small portion of words, but that more refined grammar engineering would be needed to increase the accuracy. Example analyses are provided in Figure 5.4 on p. 122.

5.8.5 Task 3: Morpheme lexicon induction

The previous two tasks considered the challenging setting of predicting a single correct segmentation and root for a given word type. In this section, we turn to another dimension of morphology learning by evaluating the morpheme lexica inferred by the adaptor grammars. By morpheme lexica we mean the sets of strings identified by



(a) Unvocalised Arabic input word “wl>stArkm”.
Reference analysis: wl␣>**st**A**r**␣km



(b) Vocalised Arabic input word “lidanotillA”. Reference analysis: li␣**d**a**n**o**t**i**l**lA

Figure 5.4: Parse tree examples for a word from the Arabic data sets BW and BW$'$ where the concatenative adaptor grammar segmented incorrectly (left).
The templatic grammars (right) correctly identified the triliteral and quadriliteral roots, and fixed the segmentation of (a). In (b), the templatic grammar improved over the baseline by finding the correct prefix but falsely posited a suffix.
(Unimportant subtrees are elided for space, while the yields of discontiguous constituents are indicated next to their symbols, with dots marking gaps. Crossing branches are not drawn but should be inferrable. Root characters are bold-faced in the reference analyses. The non-terminal X2 in (a) is part of a number of implementation-specific helper rules that ensure the appropriate handling of partly contiguous roots.)

a model as being members of a given morphological category.[22] The ability to derive such lexica from unannotated data can be useful for the construction of linguistic resources in low-resource or undocumented languages, and is relevant in computational studies of language acquisition (de Marcken, 1996; Goldwater, 2007).

Purely concatenative models of morphology can only infer lexica of prefixes, suffixes and stems. Our approach to modelling explicitly the transfixal formation of Semitic stems thus have two potential benefits: it can improve the quality of the aforementioned lexica, but it also enables the acquisition of the lexicon of root morphemes. The aim here is to test the extent to which these two potential benefits are borne out by the SRCG-based adaptor grammars on our Hebrew and Arabic data sets.

We evaluate two methods of obtaining morpheme lexica from the adaptor grammar posterior samples:

1. use the MAP parse of each word type as before (§5.8.3.1), but aggregate morphemes across word types;
2. marginalise directly over both word types and posterior samples to induce probabilistic lexica.

5.8.5.1 MAP-based morpheme lexica

We take the MAP parse trees under a given adaptor grammar as starting point, and predict the lexicon of a particular morphological category (stem, affix, root, etc.) as the unique set of strings appearing under that category across the word types in the data set.[23] Prefixes, stems and suffixes are identified using the same method of traversing the parse trees as in the segmentation task, and we discuss the root lexicon shortly.

The evaluation metric is the F-score when comparing a hypothesised lexicon against the corresponding gold lexicon.

---

[22] For simplicity, we use the term “string” in this section to mean both contiguous character sequences as well as non-contiguous tuples of characters.

[23] A given string can therefore appear in multiple lexica, which fits the linguistic reality.

The primary result is that, consistent with our expectations, the models capable of forming complex stems obtain a marked improvement in prefix, stem and suffix lexicon F-score compared to the baseline concatenative adaptor grammar (Table 5.8). The amount of improvement grows with the expressivity of the templatic grammar, most pronouncedly for the vocalised Arabic data set.

For unvocalised Arabic, the templatic grammars improve over the baseline by recovering more of the correct stems—recall goes from 32% under **Concat** to 47% under **TplR4** while precision remains at 74%. This equates to an additional 3673 correctly inferred stems. On vocalised Arabic and Hebrew, the stem F-score improvement is much more substantial (roughly 20% to 60%) and arises from increases in both precision and recall; precision reaches about 70% and recall 54% for these two vocalised data sets when using **TplR4**. This grammar thus induces stem lexica at quite high precision for three different data sets.

In contrast to stems, the affix lexica are very noisy. The concatenative models tend to over-generate to obtain very high recall but low precision. The templatic grammars balance out that trade-off to some extent.

To obtain a root lexicon based on the MAP derivations for each data set we leveraged the outcome of the earlier root identification experiment (§5.8.4.3). For BW and HEB, we take the R3 yields, while for BW' we take the T3 yields. Variation among the different templatic grammars was negligible, so we report the results only for **Tpl** in Table 5.9. The lexica of roots identified for the two Arabic data sets in this fashion contained just over 50% legitimate roots. The high recall of 80% on the unvocalised version equates to recovering 2602 of the 3254 surface-realised triliteral roots present in that data set.

5.8.5.2 Probabilistic morpheme lexica

The preceding set-based evaluation of the morpheme lexica imposes hard decisions about category membership and does not make use of the fact that the adaptor grammars induce a probability distribution over strings. In this section we measure mor-

	Bw′			Bw			Heb		
	Pre	Stem	Suf	Pre	Stem	Suf	Pre	Stem	Suf
Concat	15.0	20.2	25.4	32.8	44.1	40.3	18.7	20.9	29.2
Tpl	24.7	39.4	35.2	45.9	54.7	47.9	35.1	59.6	52.9
TplR4	37.8	60.3	47.0	53.0	57.7	51.9	38.0	62.4	55.2

Table 5.8: Morpheme lexicon induction quality. F1-scores for lexicons induced from the maximum *a posteriori* parse of each different dataset.

	P	R	F
Bw′ (T3)	54.9	47.9	51.2
Bw (R3)	51.6	80.0	62.7
Heb (R3)	25.3	55.3	34.8

Table 5.9: Precision, recall and F-score for root lexicon induction using the grammar **Tpl**.

pheme lexicon induction performance by considering the probability with which a string falls under a particular non-terminal category in an adaptor grammar.

From our posterior samples we can estimate the joint probability of a string s appearing under grammar category C by summing over all word types and posterior samples:

$$P(s, C) \approx \frac{1}{JN} \sum_{j=1}^{J} \sum_{i=1}^{N} \begin{cases} 1 & \text{if non-terminal C dominates } s \text{ in analysis } t_i^{(j)} \\ 0 & \text{otherwise.} \end{cases}$$

We can thus produce a list of roots ranked by their joint probability values under a given adaptor grammar. As an example, we present in Table 5.10 the top five triliteral roots obtained from the Hebrew data set in this fashion. The first four of these correspond to correct Hebrew roots, demonstrating that the SRCG-based adaptor grammars can learn linguistically relevant, discontiguous entities from unannotated data. In the same table, we present some example usages of the roots. As a concrete example of how the model successfully encodes the original intuition that discontiguous morphemes can be shared across word forms, consider that it correctly associates the root *lbS* with such orthograpically distinct word forms as *labaSt*, *tilbeSi* and *lehalbiS*, which are not linked by a shared stem.

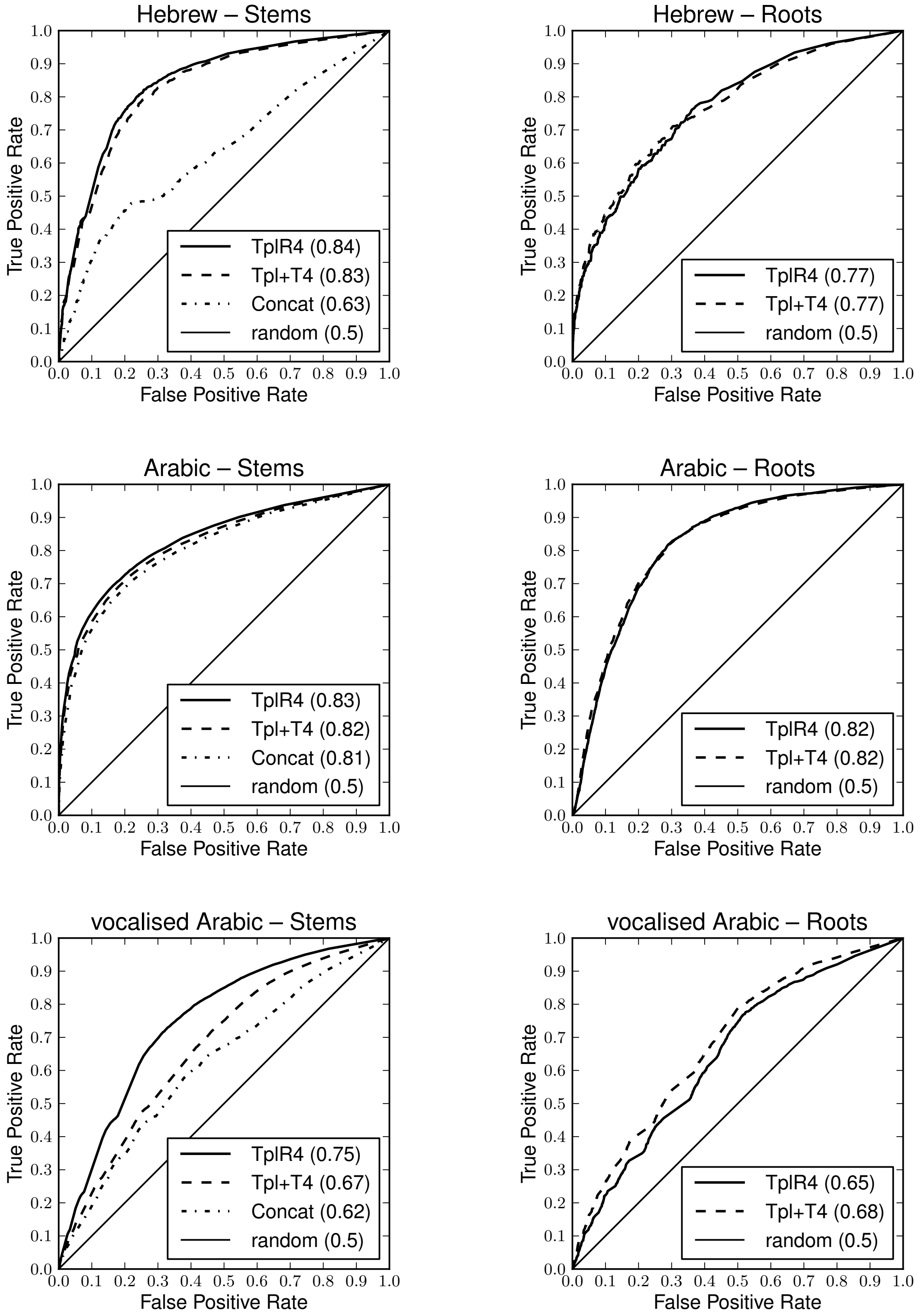
ROC analysis A snapshot of the top-ranked items does not quantify the quality of a morpheme lexicon very broadly. But it illustrates a simple sense in which we can combine a decision rule with an adaptor grammar to obtain a discrete classifier of strings. Instead of the decision rule "top n items from ranked list", we can also base the binary decision of whether a particular string s is part of a morphological category C directly on the probability $P(s|C)$ using a threshold τ between 0 and 1. Let a predicted lexicon L_τ^C be the set of strings for which $P(s|C) \geq \tau$.

This allows using receiver operating characteristics (ROC) analysis (Egan, 1975; Fawcett, 2006) to characterise the trade-off between true positive and false positive predictions across different decision thresholds τ. A ROC plot is obtained by plotting the number of true positives (e.g. true stems) versus the number of false positives (e.g. false stems) for L_τ^C as τ is varied from 0 to 1. A coin-tossing random baseline classifier traces a diagonal line ($y = x$) through the origin. Better classifiers give rise to points in ROC space toward the upper left of the plot. Finally, the area under a curve (AUC) on the ROC plot gives a summary of the general predictive performance of a model. Because the ROC space is the unit square, AUC is a number between 0 and 1. AUC is equivalent to the probability that a model would rank a randomly chosen positive instance higher than a randomly chosen negative instance (Fawcett, 2006). ROC plots and AUC are non-parametric methods, and therefore give robust results insensitive to internal differences between models or skews in the number of true versus negative instances.

Figure 5.5 shows the ROC plots and AUC values for inducing stem and root lexica from our data sets using different adaptor grammars. The first main outcome is that the prediction of stems benefits from the use of the root-templatic SRCG rules, consistent with the results in the MAP-based evaluation. This effect is again most pronounced for the vocalised data sets BW$'$ and HEB. The second main result is that the prediction of root lexica for HEB and BW is substantially better than the random baseline: the SRCG-based models would rank a true root above a non-root string with probability 0.77 and 0.82 on these respective data sets. For the vocalised BW$'$, the corresponding value is 0.68. As a means for predicting whether or not particular strings are root morphemes of the language, our method thus performs better than might be expected from the low root-identification results in the per-word analysis in §5.8.4.3.

				Mistaken Examples	
$P(s, R3)$	**Root** s	**Correct Examples**		Bad Segmentation	Bad Root
0.00649	spr ✓	**s**a**p**a**r**␣ti	ye␣**s**a**pr**␣u	me⊗**s**a**pr**␣im	hi⊗**st**a**p**⊗ar↑t
0.00527	lbs ✓	**l**a**b**a**S**␣t	ti␣**lb**e**S**␣i	le↑ha⊗**lb**i**S**	
0.00524	ptx ✓	**p**a**t**a**x**␣ti	ti␣**pt**e**x**␣i	ni⊗**pt**a**x**␣at	li␣**p**t**o**a**x**
0.00521	rxc ✓	yi␣**rx**a**c**	**r**o**x**e**c**␣et	le↑hit⊗**r**a**x**e**c**	
0.00510	!al ×			le↑ha⊗**!al**⊗ot	**!a**c**l**⊗an↑it

Table 5.10: Top five Hebrew triliteral roots hypothesised by the grammar **Tpl+T4** as ranked by model probability, along with examples of their occurrence in the analysed words. The first four are legitimate roots, while the fifth is not a valid root.
The middle two columns give examples of words where the model found the correct segmentation points (␣) and the predicted root characters coincide with the true root characters (**bold**). The penultimate column shows examples where the root is found despite segmentation mistakes, marking false positive (⊗) and false negative (↑) segmentation points. Examples in the final column feature segmentation and root mistakes—false positive root characters appear in **grey**, while false negatives are boxed.



Hebrew – Stems

Hebrew – Roots

Arabic – Stems

Arabic – Roots

vocalised Arabic – Stems

vocalised Arabic – Roots

Figure 5.5: ROC curves for predicting the stem lexicon (left) and root lexicon (right) for the Hebrew and Arabic data sets, as described in §5.8.5.2. The area under each curve (AUC) is given in parentheses, as computed with the trapezium rule.

5.9 Summary

This chapter introduced a new approach to the unsupervised learning of non-concatenative morphology. We showed how the simple Range Concatenating Grammar formalism can be used to encode non-concatenative morphological processes which do not involve character deletion or insertion when composing morphemes into a surface word form. This approach can account for infixation and circumfixation, but our focus was to instantiate and evaluate the general idea on the root-template-based morphology employed in Semitic languages. For a learning framework, we made the natural extension of non-parametric Bayesian adaptor grammars to the SRCG formalism, which makes it possible to model explicitly the tendency of morphemes (contiguous and discontiguous ones) to be reused in different words. Aside from our application to morphology, this learning approach could also be useful in other applications of SRCGs.

Our experimentation on Hebrew, Quranic Arabic and two versions of a synthetic data set containing morphotactically valid standard Arabic words shows that modelling discontiguous sub-structures of words improves the segmentation of words into morphemes. A simple root identification method correctly identified triliteral roots for about two-thirds of the words where they are expected, for two of the aforementioned data sets. On vocalised Arabic, absolute performance across the different tasks was generally lower, and very low for the Quran. Finally, we found that our SRCG-based models could be used to infer a lexicon of triliteral root morphemes. This mixed outcome flows from the fact that we deliberately sought to limit the amount of language-specific knowledge built into the grammars in order to test to what extent a high-level notion of intercalated morphology is beneficial in these kind of unsupervised models. For the Hebrew and synthetic Arabic data sets, this was balanced against exercising more control over the word shapes we included in the evaluation, whereas for the Quran we applied no informed preprocessing or standardisation whatsoever.

Chapter 6

Conclusion

This dissertation presented original work on the problem of capturing sub-word structure in probabilistic models of language. The central approach was to use the essence of certain morphological processes as a basis for extending three different types of models in order to make them better suited to morphologically rich languages. We applied this approach both to language models (LMs) which define probabilities of word sequences, and to unsupervised models that seek to induce the morphemes of individual words from unannotated text.

In chapter 3 we presented a hierarchical Bayesian model of word sequences which accounts for productive compound formation. We exploited the right-headedness of German compounds to structure the statistical dependencies between a compound word and its immediately preceding context, while using an embedded language model to generate the modifiers within each compound independently of the word's context. The main findings from our evaluation were as follows:

1. The compounding language model succeeded in reducing the data sparsity due to compounding by expressing compounds in terms of their parts—in our data set, the number of distinct elements in the base distribution at the root of the hierarchical model is reduced to less than half the number of unsegmented word types. The model obtained small improvements in test set perplexity compared to a baseline language model that views compounds as atomic elements.

2. The use of the compounding language model as a feature in a machine translation system led to translations that were judged to be equivalent in overall quality with that from a baseline language model, as measured with BLEU. Finer-grained measurement showed that the compounding model did, however, improve the translation system's precision in outputting compound words,

3. Structuring the statistical dependencies according to the right-headedness of German compounds played a key role in the aforementioned outcomes. This was established by observing a degradation in perplexity and BLEU when using a model variant that posits left-headedness.

In chapter 4 we shifted the attention to another type of language model to consider how sub-word structure could be leveraged in another setting. Based on automatically induced distributed feature representations of words and smooth scoring functions, distributed language models (DLMs) circumvent the need for smoothing by means of back-off as employed in traditional n-gram LMs, and as we refined further in the compounding model. Our contribution in this regard is two-fold: We show that DLMs can benefit substantially from explicitly modelling sub-word structure, and we determine that it is possible and beneficial to use a normalised DLM as part of a machine translation decoder.

We formulated a variant of the log bilinear (LBL) language model (Mnih and Teh, 2012) that incorporates a simple method of composing word vectors from morpheme vectors. Morpheme vectors are learnt as part of the model, leveraging a separate unsupervised model to obtain the morphological segmentation itself. The method is inspired directly by recent positive results in modelling compositional morphology (Luong et al., 2013; Lazaridou et al., 2013) and the observation that word-based DLMs already capture non-trivial, morphologically relevant regularities (Mikolov et al., 2013b). The main findings from our monolingual evaluation are as follows:

1. The morpheme-based LBL achieved substantial reductions in test set perplexity for six languages of varying morphological complexity. The model af-

fects a soft tying of parameters across words that share morphemes, which contributed to the largest perplexity improvements being generally on rarer words. This is an important outcome for modelling morphologically richer languages, where rare words tend to form a larger portion of the vocabulary.

2. The availability of morpheme vectors allowed for the composition of high-quality word vectors for out-of-vocabulary words, as measured in a word similarity rating task on multiple individual languages.

The integration of correctly normalised LBLs into a machine translation decoder was made tractable by partitioning the vocabulary into word classes and decomposing the probability model accordingly. This is a known method of decreasing the computational cost involved in computing probabilities with DLMs. To our knowledge, this work is however the first to deploy it successfully in translation—previous uses of DLMs in translation have been limited to rescoring n-best lists or lattices, or have used unnormalised DLMs. Our evaluation findings were as follows:

1. Use of a LBL LM feature in the log-linear translation model (in addition to a modified Kneser-Ney (MKN) LM feature) provided consistent improvements in BLEU when translating into six languages of varying morphological complexity, compared to a baseline system using only the MKN LM.

2. The morpheme-based LBL provided further small increases in BLEU, though the improvements cannot be separated entirely from the instability of the optimiser used to tune the translation model feature weights.

As a third and final instance of capturing sub-word structure in probabilistic models of language, we switched perspective to the question of *acquiring* such sub-word structure in the first place, rather than relying on an external source as we did in the sequence-based models. Our contribution in this regard is to suggest that simple Range Concatenating Grammars (SRCGs) are useful for encoding both concatenative and non-concatenative morphological processes in a single formalism. The grammar-based approach allows for unsupervised learning to be done using the non-parametric Bayesian framework of adaptor grammars, which we instantiate for the SRCG formalism. We applied this approach to Semitic morphology, where words are often formed by first interspersing the characters of root morphemes into templatic structures and then attaching further affixes in a concatenative fashion. As such, it offers a useful test-bed for the notion that joint modelling of concatenative and non-concatenative morphological processes could have a mutually disambiguating effect in morphological analysis. Our evaluation gave rise to the following main findings:

1. Modelling discontiguous sub-word structure improved the segmentation of Arabic and Hebrew words into linear morphemes, as measured by the F-score in comparison to baseline grammars that ignore discontiguous sub-structure. This holds for both orthographic variants of our Arabic data set (with and without vocalisation).
2. A simple method for extracting root morphemes from the inferred parse trees of words produced mixed results. On two of the data sets it correctly identified about two-thirds of triliteral roots for the words where such roots are expected, but for the other data sets and for the identification of quadriliteral roots in all our data sets, the accuracy was low.
3. The intercalating adaptor grammars also produced drastic F-score increases for the lexica of stems induced from data, relative to the purely concatenating baseline grammars. This is related to the improved segmentation performance.
4. Considered as binary classifiers of strings, the intercalating adaptor grammars obtained moderately high performance in discriminating between string tuples that are triliteral roots and those that are not.

A characteristic of our unsupervised morphology learning work was that we erred on the side of underspecification when it came to engineering rules to fit the data sets and languages used, and this is reflected in the mixed results. Our primary intention was to test the extent to which a high-level notion of non-concatenative morphology could reasonably be expressed in adaptor grammars. This paves the way for interesting future applications, and creates a framework within which it is straightforward to encode finer-grained knowledge of particular non-concatenative processes.

In conclusion, this dissertation showed that the explicit inclusion of sub-word structure in probabilistic language models is beneficial for morphologically complex languages in settings ranging from unsupervised morphology learning and word similarity rating to the estimation of word-sequence probabilities and their application in machine translation.

6.1 Extensions and further research

Possible directions for further research emanating from the work presented in this dissertation include the following:

1. The LMs developed in Chapters 3 and 4 are in principle capable of providing probabilities for OOV words by leveraging their sub-word representations encoded in the models. For a machine translation system to benefit maximally from that ability, the system itself also requires the ability to hypothesise novel word forms in its translation output. An interesting direction for future research is to combine our LMs with the recently proposed methodology of Chahuneau et al. (2013a) where novel inflections of words can be hypothesised by the translation system.

2. Both LMs made use of deterministic segmentations of words into sub-parts. An alternative would be to learn the segmentation jointly with the rest of the model, e.g. as done by (Mochihashi et al., 2009) in the context of Chinese word segmentation.

3. The method from Chapter 4 for composing word vectors from morpheme vectors through addition is readily implementable in bilingual or multilingual distributed models (e.g. Zou et al., 2013). Aside from the parameter tying effect

that should also be useful for rare words in that setting, an interesting question is whether a morpheme-based bilingual model can capture the tendency of a given language to encode certain grammatical properties with distinct words while another language uses bound morphemes for the same.

4. Our approach to modelling discontiguous sub-word structure in Chapter 5 presents interesting opportunities for extending the line of work that approaches unsupervised morphology learning from the perspective of language acquisition. Frank et al. (2013) recently considered the mutually disambiguating roles of morphology and syntax in this context, and a similar study on languages employing non-concatenative morphology could benefit from our work.

5. The use of SRCGs in Chapter 5 affords modelling convenience but comes in exchange for high polynomial parsing time complexity. As with adaptor grammars generally, the emphasis is on being able to explore different model assumptions easily. Yet full recursive power is not necessary for many morphological processes, so that a relevant question is whether finite-state transformations or approximations could be used to cast a given adaptor grammar into a more efficient computational framework. There is a rich literature on this for context-free grammars (e.g. Nederhof, 2000).

6. The burden of defining the appropriate set of rewrite rules for the base distribution grammar could be overcome by using a semi-supervised approach that searches for optimality over different instantiations of so-called metagrammars (Sirts and Goldwater, 2013).

References

H. Adel, N. T. Vu, and T. Schultz. 2013. Combination of Recurrent Neural Networks and Factored Language Models for Code-Switching Language Modeling. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 206–211, Sofia, Bulgaria, August. Association for Computational Linguistics.

A. Albert, B. MacWhinney, B. Nir, and S. Wintner. 2013. The Hebrew CHILDES corpus: transcription and morphological analysis. *Language Resources and Evaluation*, 47(4):973–1005.

A. Alexandrescu and K. Kirchhoff. 2006. Factored Neural Language Models. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, New York City, USA, June. Association for Computational Linguistics.

A. Allauzen, N. Pechéux, Q. K. Do, M. Dinarelli, T. Lavergne, A. Max, H.-S. Le, and F. Yvon. 2013. LIMSI@ WMT13. In *Proceedings of the Eighth Workshop on Statistical Machine Translation*, pages 62–69, Sofia, Bulgaria, August. Association for Computational Linguistics.

M. Baroni and J. Matiasek. 2002. Predicting the components of German nominal compounds. In *Proceedings of the 15th European Conference on Artificial Intelligence (ECAI)*, pages 470–474, Lyon, France, July.

M. Baroni and R. Zamparelli. 2010. Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1183–1193, Cambridge, MA.

M. Baroni, J. Matiasek, and H. Trost. 2002. Unsupervised discovery of morphologically related words based on orthographic and semantic similarity. In *Proceedings of the ACL-02 Workshop on Morphological and Phonological Learning*, pages 48–57. Association for Computational Linguistics, July.

M. J. Beal. 2003. *Variational Algorithms for Approximate Bayesian Inference*. Ph.D. thesis, University of London.

K. R. Beesley and L. Karttunen. 2003. *Finite state morphology*, volume 18. CSLI Publications Stanford.

Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin. 2003. A Neural Probabilistic Language Model. *Journal of Machine Learning Research*, 3:1137–1155.

A. Berton, P. Fetter, and P. Regel-Brietzmann. 1996. Compound Words in Large-Vocabulary German Speech Recognition Systems. In *Proceedings of Fourth International Conference on Spoken Language Processing. ICSLP '96*, volume 2, pages 1165–1168. IEEE.

J. A. Bilmes and K. Kirchhoff. 2003. Factored Language Models and Generalized Parallel Backoff. In *Proceedings of Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics: short papers*. Association for Computational Linguistics.

P. Blunsom and T. Cohn. 2011. A Hierarchical Pitman-Yor Process HMM for Unsupervised Part of Speech Induction. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 865–874, Portland, Oregon, USA, June. Association for Computational Linguistics.

P. Blunsom, T. Cohn, S. Goldwater, and M. Johnson. 2009. A Note on the Implementation of Hierarchical Dirichlet Processes. In *Proceedings of the ACL-IJCNLP 2009 Conference: Short Papers*, pages 337–340. Association for Computational Linguistics, August.

J. A. Botha. 2012. Hierarchical Bayesian Language Modelling for the Linguistically Informed. In *Proceedings of the Student Research Workshop at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 64–73, Avignon, France. Association for Computational Linguistics.

J. A. Botha and P. Blunsom. 2013. Adaptor Grammars for Learning Non-Concatenative Morphology. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, Seattle, Washington, USA, October.

J. A. Botha and P. Blunsom. 2014. Compositional Morphology for Word Representations and Language Modelling. In *Proceedings of the 31st International Conference on Machine Learning, ICML*, pages 1899–1907, Beijing, China, June.

J. A. Botha, C. Dyer, and P. Blunsom. 2012. Bayesian Language Modelling of German Compounds. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING)*, Mumbai, India.

A. Boudlal, R. Belahbib, A. Lakhouaja, A. Mazroui, A. Meziane, and M. Bebah. 2009. A Markovian approach for Arabic Root Extraction. *The International Arab Journal of Information Technology*, 8(1):91–98.

P. Boullier. 2000. A cubic time extension of context-free grammars. *Grammars*, 3 (2-3):111–131.

T. Brants, A. C. Popat, P. Xu, F. J. Och, and J. Dean. 2007. Large language models in machine translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, volume 1, pages 858–867. Association for Computational Linguistics, June.

P. F. Brown, P. V. DeSouza, R. L. Mercer, V. J. Della Pietra, and J. C. Lai. 1992. Class-Based n-gram Models of Natural Language. *Computational Linguistics*, 18 (4):467–479.

T. Buckwalter. 2002. Buckwalter Arabic Morphological Analyzer Version 1.0. Technical report, Linguistic Data Consortium.

J. A. Bullinaria and J. P. Levy. 2007. Extracting Semantic Representations from Word Co-occurrence Statistics: A Computational Study. *Behavior research methods*, 39(3):510–26, August.

Ç. Çöltekin. 2010. A Freely Available Morphological Analyzer for Turkish. In *Proceedings of the 7th International Conference on Language Resources and Evaluation (LREC2010)*, pages 820–827.

V. Chahuneau, E. Schlinger, N. A. Smith, and C. Dyer. 2013a. Translating into Morphologically Rich Languages with Synthetic Phrases. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1677–1687, Seattle, Washington, USA. Association for Computational Linguistics.

V. Chahuneau, N. A. Smith, and C. Dyer. 2013b. Knowledge-rich morphological priors for Bayesian language models. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1206–1215, Atlanta, Georgia. Association for Computational Linguistics.

C. Chelba, T. Mikolov, M. Schuster, Q. Ge, T. Brants, P. Koehn, and T. Robinson. 2014. One Billion Word Benchmark for Measuring Progress in Statistical Language Modeling. In *Proceedings of the 15th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pages 2635–2639, Singapore.

S. F. Chen and J. Goodman. 1998. An Empirical Study of Smoothing Techniques for Language Modeling. Technical report, Center for Research in Computing Technology, Harvard University, Cambridge, MA.

S. F. Chen and J. Goodman. 1999. An empirical study of smoothing techniques for language modeling. *Computer Speech & Language*, 13:359–394, October.

D. Chiang. 2007. Hierarchical Phrase-Based Translation. *Computational Linguistics*, 33(2):201–228.

A. Clark. 2001a. Unsupervised induction of stochastic context-free grammars using distributional clustering. In *Proceedings of the ACL 2001 Workshop on Computational Natural Language Learning (ConLL)*, pages 105–112, Toulouse, France. Association for Computational Linguistics.

A. Clark. 2001b. Partially Supervised Learning of Morphology with Stochastic Transducers. In *Proceedings of Natural Language Processing Pacific Rim Symposium*, pages 341–348, Tokyo, Japan, November.

A. Clark. 2002. Memory-Based Learning of Morphology with Stochastic Transducers. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 513–520.

J. H. Clark, C. Dyer, A. Lavie, and N. A. Smith. 2011. Better Hypothesis Testing for Statistical Machine Translation: Controlling for Optimizer Instability. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 176–181, Portland, Oregon, USA.

S. B. Cohen, D. M. Blei, and N. A. Smith. 2010. Variational Inference for Adaptor Grammars. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 564–572, Los Angeles, California. Association for Computational Linguistics.

Y. Cohen-Sygal and S. Wintner. 2006. Finite-state registered automata for non-concatenative morphology. *Computational Linguistics*, 32(1):49–82.

T. Cohn, P. Blunsom, and S. Goldwater. 2010. Inducing Tree-Substitution Grammars. *Journal of Machine Learning Research*, 11:3053–3096.

R. Collobert and J. Weston. 2008. A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning. In *Proceedings of the 25th International Conference on Machine Learning*, Helsinki, Finland.

M. Creutz and K. Lagus. 2007. Unsupervised Models for Morpheme Segmentation and Morphology Learning. *ACM Transactions on Speech and Language Processing*, 4(1):1–34.

K. Darwish. 2002. Building a shallow Arabic morphological analyzer in one day. In *Proceedings of the ACL-02 Workshop on Computational Approaches to Semitic Languages*, pages 47–54. Association for Computational Linguistics.

E. Daya, D. Roth, and S. Wintner. 2008. Identifying Semitic Roots: Machine Learning with Linguistic Constraints. *Computational Linguistics*, 34(3):429–448.

R. G. de Almeida and G. Libben. 2005. Changing morphological structures: The effect of sentence context on the interpretation of structurally ambiguous English trimorphemic words. *Language and Cognitive Processes*, 20(1-2):373–394.

C. G. de Marcken. 1996. *Unsupervised Language Acquisition*. Ph.D. thesis, Massachusetts Institute of Technology.

A. N. de Roeck and W. Al-Fares. 2000. A morphologically sensitive clustering algorithm for identifying Arabic roots. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 199–206. Association for Computational Linguistics.

J. Duchi, E. Hazan, and Y. Singer. 2011. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *Journal of Machine Learning Research*, 12:2121–2159.

K. Duh, G. Neubig, K. Sudoh, and H. Tsukada. 2013. Adaptation Data Selection using Neural Language Models: Experiments in Machine Translation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 678–683, Sofia, Bulgaria.

K. Dukes and N. Habash. 2010. Morphological Annotation of Quranic Arabic. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*.

C. Dyer. 2009. Using a maximum entropy model to build segmentation lattices for MT. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational*

Linguistics, pages 406–414, Boulder, Colorado. Association for Computational Linguistics.

C. Dyer, A. Lopez, J. Ganitkevitch, J. Weese, F. Ture, P. Blunsom, H. Setiawan, V. Eidelman, and P. Resnik. 2010. cdec: A Decoder, Alignment, and Learning Framework for Finite-State and Context-Free Translation Models. In *Proceedings of the ACL 2010 System Demonstrations*, pages 7–12, Uppsala, Sweden. Association for Computational Linguistics.

C. Dyer, V. Chahuneau, and N. A. Smith. 2013. A Simple, Fast, and Effective Reparameterization of IBM Model 2. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 644–648, Atlanta, Georgia, USA. Association for Computational Linguistics.

J. P. Egan. 1975. *Signal detection theory and ROC analysis*. Series in Cognition and Perception. Academic Press, New York, NY.

T. Fawcett. 2006. An introduction to ROC analysis. *Pattern Recognition Letters*, 27 (8):861–874, June.

T. S. Ferguson. 1973. A Bayesian analysis of some nonparametric problems. *Annals of Statistics*, (1):209–230.

R. Finkel and G. Stump. 2002. Generating Hebrew verb morphology by default inheritance hierarchies. In *Proceedings of the ACL Workshop on Computational Approaches to Semitic Languages*. Association for Computational Linguistics.

L. Finkelstein, E. Gabrilovich, Y. Matias, E. Rivlin, Z. Solan, G. Wolfman, and E. Ruppin. 2002. Placing Search in Context: The Concept Revisited. *ACM Transactions on Information Systems*, 20(1):116–131.

M. Forsberg and A. Ranta. 2004. Functional Morphology. In *Proceedings of the Ninth ACM SIGPLAN International Conference on Functional Programming*, pages 213–223, Utah, September. Association for Computing Machinery.

S. Frank, F. Keller, and S. Goldwater. 2013. Exploring the utility of joint morphological and syntactic learning from child-directed speech. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, Seattle, Washington, USA. Association for Computational Linguistics.

M. A. Fullwood and T. J. O'Donnell. 2013. Learning non-concatenative morphology. In *Proceedings of the Fourth Annual Workshop on Cognitive Modeling and Computational Linguistics*, pages 21–27, Sofia, Bulgaria. Association for Computational Linguistics.

M. Gasser. 2009. Semitic morphological analysis and generation using finite state transducers with feature structures. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 309–317. Association for Computational Linguistics.

P. Geutner. 1995. Using Morphology Towards Better Large-Vocabulary Speech Recognition Systems. In *Proceedings of the 1995 International Conference on Acoustics, Speech, and Signal Processing*, pages 445–448. IEEE.

D. Gildea. 2010. Optimal Parsing Strategies for Linear Context-Free Rewriting Systems. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 769–776. Association for Computational Linguistics.

W. R. Gilks, S. Richardson, and D. J. Spiegelhalter. 1996. *Markov Chain Monte Carlo in Practice*. Chapman & Hall/CRC Interdisciplinary Statistics.

J. Goldsmith and T. Reutter. 1998. Automatic Collection and Analysis of German Compounds. In F. Busa F. et al., editor, *The Computational Treatment of Nominals*, pages 61–69. Universite de Montreal, Canada.

S. Goldwater. 2007. *Nonparametric Bayesian Models of Lexical Acquisition*. Ph.D. thesis, Brown University.

S. Goldwater, T. L. Griffiths, and M. Johnson. 2006. Interpolating Between Types and Tokens by Estimating Power-Law Generators. In *Advances in Neural Information Processing Systems*, pages 459–466.

S. Goldwater, T. L. Griffiths, and M. Johnson. 2011. Producing Power-Law Distributions and Damping Word Frequencies with Two-Stage Language Models. *Journal of Machine Learning Research*, 12:2335–2382.

I. J. Good. 1953. The Population Frequencies of Species and the Estimation of Population Parameters. *Biometrika*, 40(3):237–264.

J. Goodman. 2001. Classes for Fast Maximum Entropy Training. In *Proceedings of the 2001 International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 561–564. IEEE.

J. T. Goodman. 1998. *Parsing Inside-Out*. Ph.D. thesis, Harvard University.

I. Gurevych. 2005. Using the Structure of a Conceptual Network in Computing Semantic Relatedness. In *Proceedings of the Second International Joint Conference on Natural Language Processing: Full Papers*, pages 767–778.

M. U. Gutmann and A. Hyvärinen. 2012. Noise-Contrastive Estimation of Unnormalized Statistical Models , with Applications to Natural Image Statistics. *The Journal of Machine Learning Research*, 13:307–361.

N. Habash and O. Rambow. 2005. Arabic Tokenization, Part-of-Speech Tagging and Morphological Disambiguation in One Fell Swoop. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 573–580, Ann Arbor, June. Association for Computational Linguistics.

N. Habash and O. Rambow. 2006. MAGEAD: a morphological analyzer and generator for the Arabic dialects. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, volume M, pages 681–688, Sydney, Australia, July. Association for Computational Linguistics.

N. Habash and F. Sadat. 2006. Arabic preprocessing schemes for statistical machine translation. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 49–52. Association for Computational Linguistics.

N. Habash, O. Rambow, and G. Kiraz. 2005. Morphological Analysis and Generation for Arabic Dialects. In *Proceedings of the ACL Workshop on Computational Approaches to Semitic Languages*, pages 17–24. Association for Computational Linguistics, June.

N. Y. Habash. 2010. Introduction to Arabic natural language processing. *Synthesis Lectures on Human Language Technologies*, 3(1):1–187.

H. Hammarström and L. Borin. 2011. Unsupervised Learning of Morphology. *Computational Linguistics*, 37(2):309–350.

S. Hassan and R. Mihalcea. 2009. Cross-lingual Semantic Relatedness Using Encyclopedic Knowledge. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1192–1201. Association for Computational Linguistics.

K. Heafield. 2011. KenLM: Faster and Smaller Language Model Queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 187–197. Association for Computational Linguistics.

K. Heafield, I. Pouzyrevsky, J. H. Clark, and P. Koehn. 2013. Scalable Modified Kneser-Ney Language Model Estimation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 690–696, Sofia, Bulgaria, August. Association for Computational Linguistics.

K. M. Hermann and P. Blunsom. 2013. The Role of Syntax in Vector Space Models of Compositional Semantics. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 894–904, Sofia, Bulgaria.

T. Hirsimäki, M. Creutz, V. Siivola, M. Kurimo, S. Virpioja, and J. Pylkkönen. 2006. Unlimited vocabulary speech recognition with morph language models applied to Finnish. *Computer Speech and Language*, 20(4):515–541, October.

E. H. Huang, R. Socher, C. D. Manning, and A. Y. Ng. 2012. Improving Word Representations via Global Context and Multiple Word Prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 873–882. Association for Computational Linguistics.

S. Huang and S. Renals. 2009. A parallel training algorithm for hierarchical Pitman-Yor process language models. In *Proceedings of Interspeech*, volume 9, pages 2695–2698.

S. Huang and S. Renals. 2010. Hierarchical Bayesian Language Models for Conversational Speech Recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(8):1941–1954, November.

Y. Huang, M. Zhang, and C. L. Tan. 2011. Nonparametric Bayesian Machine Transliteration with Synchronous Adaptor Grammars. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 534–539. Association for Computational Linguistics.

P. Ircing, P. Krbec, J. Hajic, S. Khudanpur, F. Jelinek, J. Psutka, and W. Byrne. 2001. On large vocabulary continuous speech recognition of highly inflectional language – Czech. In *Proceedings of the 7th European Conference on Speech Communication and Technology (EUROSPEECH)*, volume 1.

H. Ishwaran and L. F. James. 2001. Gibbs Sampling Methods for Stick-Breaking Priors. *Journal of the American Statistical Association*, 96(453):161–173.

H. Ishwaran and L. F. James. 2003. Generalized weighted Chinese restaurant processes for species sampling mixture models. *Statistica Sinica*, 13:1211–1235.

F. Jelinek and R. L. Mercer. 1980. Interpolated estimation of Markov source parameters from sparse data. In *Proceedings of the Workshop on Pattern Recognition in Practice*, pages 381–397, Amsterdam, The Netherlands.

M. Johnson. 2008. Unsupervised word segmentation for Sesotho using Adaptor Grammars. In *Proceedings of the Tenth Meeting of ACL Special Interest Group on*

Computational Morphology and Phonology, pages 20–27. Association for Computational Linguistics.

M. Johnson and S. Goldwater. 2009. Improving nonparameteric Bayesian inference: Experiments on unsupervised word segmentation with adaptor grammars. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 317–325. Association for Computational Linguistics.

M. Johnson, T. L. Griffiths, and S. Goldwater. 2007a. Adaptor Grammars: A Framework for Specifying Compositional Nonparametric Bayesian Models. In *Advances in Neural Information Processing Systems*, volume 19, page 641. MIT.

M. Johnson, T. L. Griffiths, and S. Goldwater. 2007b. Bayesian inference for PCFGs via Markov chain Monte Carlo. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics*, number April, pages 139–146, Rochester, NY. Association for Computational Linguistics.

A. K. Joshi. 1985. Tree adjoining grammars: How much context-sensitivity is required to provide reasonable structural descriptions? In D. R. Dowty, L. Karttunen, and A. M. Zwicky, editors, *Natural Language Parsing*, chapter 6, pages 206–250. Cambridge University Press.

C. Joubarne and D. Inkpen. 2011. Comparison of Semantic Similarity for Different Languages Using the Google N-gram Corpus and Second-Order Co-occurrence Measures. In *Proceedings of the Canadian Conference on Advances in Artificial Intelligence*, pages 216–221. Springer-Verlag.

N. Kalchbrenner and P. Blunsom. 2013. Recurrent Continuous Translation Models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1700–1709, Seattle, Washington, USA. Association for Computational Linguistics.

L. Kallmeyer. 2010. *Parsing beyond context-free grammars*. Springer Verlag.

L. Kataja and K. Koskenniemi. 1988. Finite-state Description of Semitic Morphology: A Case Study of Ancient Akkadian. In *Coling Budapest 1988 Volume 1: International Conference on Computational Linguistics*, pages 313–315.

Y. Kato, H. Seki, and T. Kasami. 2006. Stochastic Multiple Context-Free Grammar for RNA Pseudoknot Modeling. In *Proceedings of the International Workshop on Tree Adjoining Grammar and Related Formalisms*, pages 57–64, July.

S. Katz. 1987. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 35(3):400–401, March.

M. Kay. 1987. Nonconcatenative Finite-State Morphology. In *Proceedings of the third conference of the European Chapter of the Association for Computational Linguistics*.

G. A. Kiraz. 2000. Multitiered Nonlinear Morphology Using Multitape Finite Automata: A Case Study on Syriac and Arabic. *Computational Linguistics*, 26(1): 77–105, March.

K. Kirchhoff, J. Bilmes, and K. Duh. 2008. Factored Language Models Tutorial. Technical Report UWEETR-2008-0004, Department of Electrical Engineering, University ofWashington, Seattle.

R. Kneser and H. Ney. 1995. Improved Backing-off for m-gram Language Modelling. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 181–184.

P. Koehn. 2010. *Statistical Machine Translation*. Cambridge University Press.

P. Koehn and K. Knight. 2003. Empirical Methods for Compound Splitting. In *Proceedings of the 10th Conference of the European Chapter of the Association for Computational Linguistics*, pages 187–193. Association for Computational Linguistics.

P. Koehn, A. Arun, and H. Hoang. 2008. Towards better Machine Translation Quality for the German-English Language Pairs. In *Third Workshop on Statistical Machine Translation*, pages 139–142. Association for Computational Linguistics, June.

K. Koskenniemi. 1984. A general computational model for word-form recognition and production. In *Proceedings of the 10th International Conference on Computational Linguistics and 22nd Annual Meeting of the Association for Computational Linguistics*, pages 178–181. Association for Computational Linguistics.

M. Kurimo, S. Virpioja, V. T. Turunen, G. W. Blackwood, and W. Byrne. 2010. Overview and Results of Morpho Challenge 2009. In C. Peters, G. M. Nunzio, M. Kurimo, T. Mandl, D. Mostefa, A. Peñas, and G. Roda, editors, *Multilingual Information Access Evaluation I. Text Retrieval Experiments*, volume 6241 of *Lecture Notes in Computer Science*, pages 578–597. Springer Berlin / Heidelberg.

K. Lari and S. J. Young. 1990. The estimation of stochastic context-free grammars using the Inside-Outside algorithm. *Computer Speech & Language*, 4(1):35–56.

A. Lavie, A. Itai, U. Ornan, and M. Rimon. 1988. On the Applicabtlity of Two Level Morphology to the Inflection of Hebrew Verbs. Technical Report CS0513, Computer Science Department, Israeli Institute of Technology, Technion.

A. Lazaridou, M. Marelli, R. Zamparelli, and M. Baroni. 2013. Compositionally Derived Representations of Morphologically Complex Words in Distributional Semantics. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1517–1526, Sofia, Bulgaria. Association for Computational Linguistics.

H.-S. Le, I. Oparin, A. Allauzen, J.-L. Gauvain, and F. Yvon. 2011. Structured Output Layer Neural Network Language Model. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5524–5527, Prague, Czech Republic. IEEE.

Y. K. Lee, A. Haghighi, and R. Barzilay. 2011. Modeling syntactic context improves morphological segmentation. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning*. Association for Computational Linguistics.

A. Levenberg, C. Dyer, and P. Blunsom. 2012. A Bayesian Model for Learning SCFGs with Discontiguous Rules. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 223–232, July.

P. Liang, B. Taskar, and D. Klein. 2006. Alignment by Agreement. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 104–111, New York City, USA, June. Association for Computational Linguistics.

M.-T. Luong, R. Socher, and C. D. Manning. 2013. Better Word Representations with Recursive Neural Networks for Morphology. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*. Association for Computational Linguistics.

W. Maier. 2010. Direct Parsing of Discontinuous Constituents in German. In *Proceedings of the NAACL HLT 2010 First Workshop on Statistical Parsing of Morphologically-Rich Languages*, pages 58–66. Association for Computational Linguistics.

J. J. McCarthy. 1981. A Prosodic Theory of Nonconcatenative Morphology. *Linguistic Inquiry*, 12(3):373–418.

T. Mikolov, M. Karafiát, L. Burget, J. Černocký, and S. Khudanpur. 2010. Recurrent neural network based language model. In *Proceedings of Interspeech*, pages 1045–1048, September.

T. Mikolov, S. Kombrink, L. Burget, J. Černocký, and S. Khudanpur. 2011. Extensions of Recurrent Neural Network Language Model. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Prague, Czech Republic.

T. Mikolov, K. Chen, G. Corrado, and J. Dean. 2013a. Efficient Estimation of Word Representations in Vector Space. In *Proceedings of the International Conference on Learning Representations*. arXiv:1301.3781.

T. Mikolov, W.-t. Yih, and G. Zweig. 2013b. Linguistic Regularities in Continuous Space Word Representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751. Association for Computational Linguistics.

E. Minkov, K. Toutanova, and H. Suzuki. 2007. Generating Complex Morphology for Machine Translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 128–135, June.

J. Mitchell and M. Lapata. 2008. Vector-based Models of Semantic Composition. In *Proceedings of ACL-08: HLT*, pages 236–244, Columbus, Ohio, June. Association for Computational Linguistics.

J. Mitchell and M. Lapata. 2010. Composition in distributional models of semantics. *Cognitive science*, 34(8):1388–429, November.

A. Mnih and G. Hinton. 2007. Three New Graphical Models for Statistical Language Modelling. In *Proceedings of the 24th International Conference on Machine Learning*, pages 641–648, Corvallis, Oregon.

A. Mnih and G. Hinton. 2008. A Scalable Hierarchical Distributed Language Model. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems*, pages 1081–1088.

A. Mnih and Y. W. Teh. 2012. A fast and simple algorithm for training neural probabilistic language models. In *Proceedings of the 29th International Conference on Machine Learning*, Edinburgh, Scotland.

A. Mnih, Z. Yuecheng, and G. Hinton. 2009. Improving a statistical language model through non-linear prediction. *Neurocomputing*, 72(7-9):1414–1418.

D. Mochihashi, T. Yamada, and N. Ueda. 2009. Bayesian unsupervised word segmentation with nested Pitman-Yor language modeling. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 100–108, Suntec, Singapore. Association for Computational Linguistics.

F. Morin and Y. Bengio. 2005. Hierarchical probabilistic neural network language model. In R. G. Cowell and Z. Ghahramani, editors, *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics*, pages 246–252. Society for Artificial Intelligence and Statistics.

A. E.-D. Mousa, M. A. B. Shaik, R. Schlüter, and H. Ney. 2011. Morpheme Based Factored Language Models for German LVCSR. In *Interspeech*, pages 1445–1448.

V. Nair and G. Hinton. 2010. Rectified Linear Units Improve Restricted Boltzmann Machines. In *Proceedings of the 27th International Conference on Machine Learning*, pages 807–814, Haifa, Israel.

R. Neal. 1993. Probabilistic inference using Markov chain Monte Carlo methods. Technical Report CRG-TR-93-1, Department of Computer Science, University of Toronto.

R. M. Neal. 2003. Slice Sampling. *The Annals of Statistics*, 31(3):705–741.

M.-J. Nederhof. 2000. Practical Experiments with Regular Approximation of Context-free Languages. *Computational Linguistics*, 26(1).

G. Neubig, M. Mimura, S. Mori, and T. Kawahara. 2010. Learning a Language Model from Continuous Speech. In *Interspeech*, pages 1053–1056, Chiba, Japan, September.

H. Ney, U. Essen, and R. Kneser. 1994. On structuring probabilistic dependences in stochastic language modelling. *Computer Speech & Language*, 8(1):1–38.

F. J. Och. 2003. Minimum Error Rate Training in Statistical Machine Translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 160–167.

M. Pachitariu and M. Sahani. 2013. Regularization and nonlinearities for neural language models: when are they needed? *arXiv preprint arXiv:1301.5650.*

K. Papineni, S. Roukos, T. Ward, W.-j. Zhu, T. J. Watson, and Y. Heights. 2001. Bleu: A Method for Automatic Evaluation of Machine Translation. Technical report, IBM.

R. Pascanu, T. Mikolov, and Y. Bengio. 2012. On the difficulty of training recurrent neural networks. In *Proceedings of the 30th International Conference on Machine Learning*.

J. Pitman. 1995. Exchangeable and partially exchangeable random partitions. *Probability Theory and Related Fields*, 102:145–158.

J. Pitman and M. Yor. 1997. The Two-Parameter Poisson-Dirichlet Distribution Derived from a Stable Subordinator. *The Annals of Probability*, 25(2):855–900.

H. Poon, C. Cherry, and K. Toutanova. 2009. Unsupervised morphological segmentation with log-linear models. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 209–217. Association for Computational Linguistics.

P. Rodrigues and D. Ćavar. 2007. Learning Arabic Morphology Using Statistical Constraint-Satisfaction Models. In E. Benmamoun, editor, *Perspectives on Arabic Linguistics: Proceedings of the 19th Arabic Linguistics Symposium*, pages 63–75, Urbana, IL, USA. John Benjamins Publishing Company.

C. G. Rodríguez and G. Satta. 2009. An Optimal-Time Binarization Algorithm for Linear Context-Free Rewriting Systems with Fan-Out Two. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 985–993. Association for Computational Linguistics.

R. Rosenfeld, S. F. Chen, and X. Zhu. 2001. Whole-sentence exponential language models: a vehicle for linguistic-statistical integration. *Computers Speech & Language*, 15(1).

H. Rubenstein and J. B. Goodenough. 1965. Contextual Correlates of Synonymy. *Communications of the Association for Computing Machinery*, 8(10):627–633, October.

H. Schmid and F. Laws. 2008. Estimation of Conditional Probabilities With Decision Trees and an Application to Fine-Grained POS Tagging. In *Proceedings of the 22nd International Conference on Computational Linguistics*, pages 777–784, Manchester, UK. Association for Computational Linguistics.

P. Schone and D. Jurafsky. 2000. Knowledge-Free Induction of Morphology Using Latent Semantic Analysis. In *Fourth Conference on Computational Natural Language Learning and the Second Learning Language in Logic Workshop*, pages 67–72, Lisbon, Portugal.

H. Schwenk. 2004. Efficient Training of Large Neural Networks for Language Modeling. In *Proceedings of the IEEE Joint Conference on Neural Networks*, volume 4, pages 3059–3064. IEEE.

H. Schwenk and P. Koehn. 2008. Large and Diverse Language Models for Statistical Machine Translation. In *Proceedings of the Third International Joint Conference on Natural Language Processing: Volume-II*.

H. Schwenk, D. Dechelotte, and J.-L. Gauvain. 2006. Continuous Space Language Models for Statistical Machine Translation. In *Proceedings of COLING/ACL*, pages 723–730, Sydney, Australia. Association for Computational Linguistics.

H. Schwenk, A. Rousseau, and M. Attik. 2012. Large, Pruned or Continuous Space Language Models on a GPU for Statistical Machine Translation. In *In Proceedings of the NAACL-HLT 2012 Workshop: Will We Ever Really Replace the N-gram Model? On the Future of Language Modeling for HLT*, pages 11–19. Association for Computational Linguistics.

H. Seki, T. Matsumura, M. Fujii, and T. Kasami. 1991. On multiple context-free grammars. *Theoretical Computer Science*, 88(2):191–229.

J. Sethuraman. 1994. A Constructive Definition of Dirichlet Priors. *Statistica Sinica*, 4:639–650.

K. Sirts and S. Goldwater. 2013. Minimally-Supervised Morphological Segmentation using Adaptor Grammars. *Transactions of the Association of Computational Linguistics*, 1:255–266.

O. Smrž. 2007. *Functional Arabic Morphology: Formal System and Implementation*. Ph.D. thesis, Charles University, Prague.

B. Snyder and R. Barzilay. 2008. Unsupervised Multilingual Learning for Morphological Segmentation. In *Proceedings of ACL-HLT*, pages 737–745.

A. Stolcke. 2002. SRILM – An extensible language modeling toolkit. In *Proceedings of the International Conference on Spoken Language Processing*, pages 901–904.

A. Stolcke and S. M. Omohundro. 1994. Inducing probabilistic grammars by Bayesian model merging. In *Grammatical inference and applications*, pages 106–118. Springer.

G. Stump. 2001. *Inflectional Morphology*. Cambridge studies in linguistics. Cambridge University Press, Cambridge, UK.

S. Stymne. 2009. A comparison of merging strategies for translation of German compounds. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 61–69. Association for Computational Linguistics, April.

Y. W. Teh. 2006a. A hierarchical Bayesian language model based on Pitman-Yor processes. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 985–992. Association for Computational Linguistics.

Y. W. Teh. 2006b. A Bayesian Interpretation of Interpolated Kneser-Ney. Technical Report TRA2/06, School of Computing, National University of Singapore.

P. D. Turney and P. Pantel. 2010. From Frequency to Meaning: Vector Space Models of Semantics. *Journal of Artificial Intelligence Research*, 37:141–188.

L. van der Maaten and G. Hinton. 2008. Visualizing Data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605.

A. Vaswani, Y. Zhao, V. Fossum, and D. Chiang. 2013. Decoding with Large-Scale Neural Language Models Improves Translation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, Seattle, USA. Association for Computational Linguistics.

D. Vergyri, K. Kirchhoff, K. Duh, and A. Stolcke. 2004. Morphology-based Language Modeling for Arabic Speech Recognition. In *International Conference on Spoken Language Processing*, volume 6.

K. Vijay-Shanker, D. J. Weir, and A. K. Joshi. 1987. Characterizing structural descriptions produced by various grammatical formalisms. In *Proceedings of the 25th Annual Meeting of the Association for Computational Linguistics*, pages 104–111.

S. Virpioja, J. J. Väyrynen, M. Creutz, and M. Sadeniemi. 2007. Morphology-Aware Statistical Machine Translation Based on Morphs Induced in an Unsupervised Manner. In *Proceedings of the Machine Translation Summit XI*, pages 491–498, September.

R. Wang, M. Utiyama, I. Goto, E. Sumita, H. Zhao, and B.-L. Lu. 2013. Converting Continuous-Space Language Models into N-gram Language Models for Statistical Machine Translation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 845–850.

L. Wasserman. 2007. *All of Nonparametric Statistics*. Springer Texts in Statistics.

E. Williams. 1981. On the Notions “Lexically Related” and “Head of a Word”. *Linguistic Inquiry*, 12(2):245–274.

F. Wood and Y. W. Teh. 2009. A Hierarchical Nonparametric Bayesian Approach to Statistical Language Model Domain Adaptation. In *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics*, pages 607–614, Clearwater Beach, Florida, USA.

Y. Wu, X. Lu, H. Yamamoto, S. Matsuda, C. Hori, and H. Kashioka. 2012. Factored Language Model based on Recurrent Neural Network. In *Proceedings of COLING*, pages 2835–2850.

M. Xiao and Y. Guo. 2013. Domain Adaptation for Sequence Labeling Tasks with a Probabilistic Language Adaptation Model. In *Proceedings of the 30th International Conference on Machine Learning*, Atlanta, Georgia.

W. Xu and A. I. Rudnicky. 2000. Can Artificial Neural Networks Learn Language Models? In *Proceedings of the International Conference on Statistical Language Processing,*.

T. Zesch and I. Gurevych. 2006. Automatically creating datasets for measures of semantic relatedness. In *Proceedings of the Workshop on Linguistic Distances*, pages 16–24. Association for Computational Linguistics.

G. Zipf. 1932. *Selective Studies and the Principle of Relative Frequency in Language*. Harvard University Press, Cambridge, MA, USA.

W. Y. Zou, R. Socher, D. Cer, and C. D. Manning. 2013. Bilingual Word Embeddings for Phrase-Based Machine Translation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1393–1398, Seattle, USA. Association for Computational Linguistics.