Charles University in Prague

Faculty of Mathematics and Physics

MASTER THESIS

rijksuniversiteit groningen

faculteit der letteren

Manh-Ke Tran

Unsupervised and Semi-Supervised Multilingual Learning for Resource-Poor Languages

Institute of Formal and Applied Linguistics

Supervisor of the master thesis:	RNDr. Daniel Zeman, Ph.D.
	Marco A. Wiering, Assistant professor
Study programme:	Informatika N1801
Specialization:	Mathematical Linguistics

Prague 2012

I declare that I carried out this master thesis independently, and only with the cited sources, literature and other professional sources.

I understand that my work relates to the rights and obligations under the Act No. 121/2000 Coll., the Copyright Act, as amended, in particular the fact that the Charles University in Prague has the right to conclude a license agreement on the use of this work as a school work pursuant to Section 60 paragraph 1 of the Copyright Act.

In .... date .... Signature of the author

mo:ru:ɐ - *the peak of manhood, a mixture of bravery and kindness.*

This thesis is dedicated to my Small-B family and to my parents.

Acknowledgements

This thesis would not be possible without the help of many wonderful people.

First and foremost, I would like to thank to my supervisor Daniel Zeman for his guidance and for his endless support. Daniel has been patient explaining to me many morphological issues that occur in different languages. His door was always open, and he was willing to discuss whenever I stepped in. Daniel has encouraged me in my research and given me freedom to explore every direction. Daniel is the best supervisor I could possibly have.

I also would like to thank to my co-supervisor Marco A. Wiering. He inspired my interest in deep learning, which I indirectly used in my thesis, during his Introduction to Machine Learning class in Groningen.

I thank Dr. Žabokrtský for serving as my consultant His early feedback on my draft has shaped this thesis. I would miss the deadline if he did not constantly remind me about it.

I learned a lot about doing research from Gertjan van Noord. Gertjan has shaped my scientific thinking by his strict criteria and helpful feedback. I enjoyed long conversations with Gisela Redeker when I was looking for a potential topic for my thesis.

I am in debt to Gosse Bouma, my coordinator in Groningen. Without him, I could not be able to complete my study with all the formal processes which I need to go through. My thanks also go to Katrien Colman, whom I have not met, for helping me with my graduation procedure.

I spent an unforgettable year in Groningen. Many thanks to Ha Thanh Le and Raushanna Sarkeyeva for every moment we shared, to Nadine Haeusler for your kindness and sweetness, to Anna Logaki for our joyful study sessions, salsa dance, and garden library. All of you taught me what it meant to miss somebody. I would also like to thank many other members of Winschoterdiep family, Dorina Zhupa, Natasha Pudjiadi, Sebastian Tillmann, Nikos Kapitsinis, Jaime Alhambra Cordoba, Mariana Ruiz, and Anuka Matata for making Groningen my second home. I will not forget Nguyen Tuan Anh's family for letting me stay over at your place for few weeks, and Jelke Bloem for your study notes and your companion whenever I came back to Groningen.

I had a great time doing my summer internship in Trento, Italy. Many thanks to Sara Tonelli and Emanuele Pianta with whom I worked on readability measurement for Italian. Octavian Popescu has inspired me to travel into the wild. Thank you.

I owe my thanks to Joachim Daiber and Milos Stanojevic. My second year of the Master's program in Prague would not have been the same without the two of them. I enjoyed countless number of nerdy conversations among us, and the weekend times when we searched for some places with good music and drinks.

I would like to thank to the Erasmus Mundus European Masters Program in Language and Communication Technologies (LCT) for giving me an oppoturnity to study in Europe, to experience new things, and to meet many delightful people.

Last but not least, I would like to thank my parents Tran Van Tuong and Thai Thi Xuan Binh for their unconditional love and encouragement. Thank you for believing in me and supporting me in whatever I do.

Abstract

Název práce: Neřízené a polořízené vícejazyčné učení pro jazyky s nedostatkem zdrojů

Autor: Manh-Ke Tran

Katedra: Ústav formální a aplikované lingvistiky

Vedoucí diplomové práce: RNDr. Daniel Zeman, Ph.D., Ústav formální a aplikované lingvistiky & Marco A. Wiering, Assistant professor, Artificial Intelligence department, University of Groningen

Abstrakt: Práce se zaměřuje na neřízenou morfologickou segmentaci, jednu ze základních úloh počítačového zpracování přirozeného jazyka. V této úloze je cílem rozložit slova na morfémy. Popisuji a reimplementuji model navržený v Lee *et al.* (2011) a vyhodnocuji ho na 4 jazycích. Navíc navrhuji generativní model, který dokáže využít reprezentaci slov jako přídavné rysy. Slovní reprezentace jsou rovněž získávány neřízeným způsobem pomocí strojového učení a neuronového jazykového modelu. Pokusy ukazují, že s využitím těchto přídavných rysů celková úspěšnost neřízeného modelu vzrůstá.

Klíčová slova: neřízené učení, morfematická segmentace

Title: Unsupervised and Semi-Supervised Multilingual Learning for Resource-Poor Languages

Author: Manh-Ke Tran

Institute: Institute of Formal and Applied Linguistics

Supervisor: RNDr. Daniel Zeman, Ph.D., Institute of Formal and Applied Linguistics & Marco A. Wiering, Assistant professor, Artificial Intelligence department, University of Groningen

Abstract: This thesis focuses on unsupervised morphological segmentation, the fundamental task in NLP which aims to break words into morphemes. I describe and re-implement a model proposed in Lee *et al.* (2011) and evaluate it on 4 languages. Moreover, I present a generative model that could use word representation as extra features. The word representations are leant in unsupervised manner using neural language model. The experiment shows that using extra features improves the performance of the unsupervised model.

Keywords: unsupervised learning, morphological segmentation

Contents

List of Figures

Chapter 1

Introduction

1.1 Unsupervised Morphological Learning

1.1.1 Morphology

> "I never heard of *Uglification*," Alice ventured to say. "What is it?" The Gryphon lifted up both its paws in surprise. "Never heard of uglifying!" it exclaimed. "You know what to beautify is, I suppose?" "Yes," said Alice doubtfully: "it means to make prettier." "Well, then," the Gryphon went on, "if you don't know what to uglify is, you are a simpleton."
>
> LEWIS CARROLL, Alices Adventures in Wonderland, 1865.

In linguistics, morphology refers to the study of the internal structure of words, and of the process by which words are formed. Words are made up of *morphemes*, the smallest semantically meaningful units in a language. There are two types of morphemes, *free morphemes* and *bound morphemes*. A free morpheme can stand alone by itself as a word in the language, whereas bound morpheme can only occur as part of a larger word.

The atomic core of a word is a morpheme *root*. A root may or may not occur alone as a word, for example the root `ling` in "linguist". A *stem* is a word without *inflectional affixes*. A stem is often a result of compounding a root with other affixes, for example the word "unbearable" is formed by putting prefix `un`, stem `bear`, and suffix `able` together.

Affixes are bound morphemes which always appear attached to a root or a stem. A morpheme that occurs before a root or a stem is called *prefix*, a morpheme that occurs after a root or a stem is called *suffix*. In some languages, morphemes can be inserted into other morphemes, or attached to a root/stem both initially and finally. These morphemes are called *infixes*, the former, and *circumfixes*, the latter.

Bound morphemes can be classified into two categories: inflectional morphemes and derivational morphemes. Generally, there is a distinction between *inflectional morphology* and *word formation*. Inflectional morphology deals with the various realizations of the same lexeme, depending on its grammatical function, such as tense, number, gender and so forth. Inflectional morphemes never change the grammatical category of the stems to which they are attached. For example, suffixes `-s` and `-es` can be added to singular nouns to form plural nouns. Word formation deals with creating new lexemes from existing ones either by derivational rule, or compounding rule. Unlike inflectional morphemes when derivational morphemes attach to stems, new words with new meaning are formed. The Mock Turtle added `-ify` to the adjective "ugly" to form a verb "uglify" - means "to make ugly," then he went even further by adding `-cation` to form a noun - means "the process of making ugly." Compounding, or composition, on the other hand, refers to the process of constructing new words by puting existing lexemes (free morphemes) together. For example, words like "Batman", "Watchmen", and "Sabretooth" are formed by compounding process.

1.1.2 Unsupervised Morphological Learning

There are three common tasks for morphological learning:

1. Morphological segmentation.
2. Identification of morphologically related word forms.
3. Morphological analysis.

Under unsupervised setting, the third task is considered as the most challenge task. The output of a morphological analysis stem not only contains a list of ordered morphemes for a given word but also a label (syntactic class) for each morpheme. The second task is especially useful for many information retrieve systems. There is some significant results for this task, for example Dreyer & Eisner (2011) developed a model that could organize words into structured inflectional paradigms.

The focal point of this thesis is the first task, namely unsupervised morphological learning. That is, given a collection of raw (unannotated) natural language text data, I develop a statistical model, which could learn automatically the morphological structure of the language of the input with minimal supervision.

The model in this work is devoted to concatenate morphology (i.e. morphemes are put together.)

1.2 Motivation

Unsupervised morphological learning poses many interesting problems for researchers across different fields, from computational linguistics, cognitive science to machine learning.

In computational linguistics context, having morphological analysis of words could help other downstream NLP applications to battle data sparsity problem, especially for morphologically rich languages. Toutanova *et al.* (2008) improved the quality of statistical machine translation over both phrasal and syntax-based SMT by applying models that predict word forms from their stems. Cowan & Collins (2005) showed that exploiting morphology leads to the improvement of Spanish syntactic parser.

In cognitive science context, a powerfully computational model could shed light on how the child accomplishes the immense task of language acquisition. Unsupervised morphological learning, or more generally, unsupervised linguistic structure learning, can be considered as "the problem of induction," a famous puzzle that philosophers have inquired for over two thousand years, from Plato and Aristotle through Hume, Whewell, and Mill to Carnap, Quine, Goodman, and others in the 20th century. Computational models, which take reverse-engineering human learning and cognitive development approaches, as Tenenbaum *et al.* (2011) pointed out, can help to address some of the deepest questions about the nature and origins of human thought.

In machine learning context, unsupervised induction is more challenging in term of modeling and evaluation. Many powerful machine learning techniques have been developed to make use of unannotated data. Smith & Eisner (2005) proposed contrastive estimation, a technique that exploits implicit negative evidence to move the probability mass to the observed data. This technique, then, has been used successfully in log-linear models proposed by Poon *et al.* (2009) for unsupervised morphological segmentation task.

Last but not least, the ultimate motivation of this thesis is to build a morphological segmentation tool for poor-resource languages, for which few or no linguistically annotated resources are available.

1.3 Thesis outline

The thesis is organized as follows. Chapter 2, reviews some related works. Each of them took different approach which employed many interesting ideas from both linguistics and machine learning point of views. Chapter 2 also provides some background of Bayesian inference to prepare for the presentation of the models in this work. Chapter 3, presents two common evaluation methods for unsupervised morphological segmentation task that I use to evaluate the results along with the paired significance tests method to show the significant improvement is not due to luck. Chapter 4 describes the model proposed by Lee *et al.* (2011) and the results of using the models for various languages. Chapter 5 applies the idea of using word representations as extra features for existing NLP systems. This idea has been exploited successfully for many supervised learning tasks, however there is a limited number of works that exploits this direction for unsupervised learning. Chapter **??** summarizes the contribution of the thesis and discusses the limitations and directions for future work.

Chapter 2

Background

2.1 Previous Work

In the absence of labels, unsupervised learning must rely on a strong prior hypothesis that reflects prior knowledge about the task. In unsupervised morphological learning, a common-used hypothesis is the Minimum Description Length (MDL) principle Rissanen (1989), which favors compact representations of lexicon and corpus.

Creutz (2006) developed Morfessor, a language-independent, data-driven method for the unsupervised morphological segmentation. Morfessor has been applied successfully for various languages. Among different versions of Morfessor, Morfessor Baseline Creutz (2003); Creutz & Lagus (2002) is the oldest version and Morfessor Categories-MAP (Morfessor CatMAP for short) Creutz & Lagus (2005a) is the latest version.

Morfessor Baseline is based on the idea of language model. Given a corpus, it learns the optimal lexicon and segmentation by using MAP estimation:

$$\arg\max_{M} P(M|\text{corpus}) = \arg\max_{M} P(\text{corpus}|M)P(M) \tag{2.1}$$

where M is the language model for morphemes.

The prior probability $P(M)$ is the product of probability distributions $P(f)$ over morpheme frequency and probability distributions $P(l)$ over morpheme length.

$$P(M) = \prod_{i=1}^{M} P(f_{\sigma_i}) \times \prod_{i=1}^{M} P(l_{\sigma_i}) \tag{2.2}$$

where $\{\sigma_1, ..., \sigma_M\}$ is the set of morphemes in M. Let f_{σ_i} and l_{σ_i} denote frequency and length of morpheme σ_i respectively. Morfessor Baseline models frequency explicitly by choosing Zipf distribution for $P(f)$, and it selects Gamma distribution for morpheme length $P(l)$.

Likelihood $P(\text{corpus}|M)$ in Morfessor Baseline simply is the product of frequencies of all morphemes in the corpus.

$$P(\text{corpus}|M) = \prod_{j=1}^{W} \prod_{k=1}^{n_j} P(\sigma_{jk}) \tag{2.3}$$

here W is the size of the corpus (token-level), n_j is the number of morphemes in the j^{th} word, σ_{jk} is the k^{th} morpheme in n_j morphemes, and

$$P(\sigma_i) = \frac{f_{\sigma_i}}{\sum_{j=1}^{n} f_{\sigma_i}}$$

Morfessor Baseline employs MDL by taking frequency into account. MB seeks for the optimal set of morphemes by keeping the most frequent word types unsplit and splitting rare word types excessively.

While Morfessor Baseline ignores context dependency between morphemes (it treats "`s wing`" and "`wing s`" equally), Morfessor CatMAP makes use of this dependency by using Hidden Markov Model (HMM) to model transition probabilities between morpheme categories and emission probabilities of morphemes from categories. In Morfessor CatMAP, the MAP estimate needed to be maximized is similar to the MAP equation in Morfessor Baseline:

$$\operatorname*{arg\,max}_{\text{lexicon}} P(\text{lexicon}|\text{corpus}) = \operatorname*{arg\,max}_{\text{lexicon}} P(\text{corpus}|\text{lexicon})P(\text{lexicon}) \tag{2.4}$$

Morfessor CatMAP differs from Morfessor Baseline in the way it defines prior probability $P(\text{lexicon})$ and likelihood probability $P(\text{corpus}|\text{lexicon})$. Every morpheme in lexicon is considered as a set of *form* and *meaning*. The probability of the from of a morpheme depends on whether it is represented as a string, a letter or a concatenation of two sub-morphemes. The probability of the meaning of a morpheme depends on its frequency, its length and its context (defined through left and right perplexity). The likelihood probability $P(\text{corpus}|\text{lexicon})$ employs a first-order HMM to model the agreement between words and their category as well as inter-word syntax.

$$P(corpus|lexicon) = \prod_{j=1}^{W} \left[P(C_{j1}|C_{j0}) \prod_{k=1}^{n_j} P(\sigma_{jk}|C_{jk}) P(C_{j(k+1)}|C_{jk}) \right] \tag{2.5}$$

where C_{jk} denotes the category of k^{th} morpheme σ_{jk} in j^{th} word with n_j segments.

Lignos (2010) presented MORSE (**MO**Rphological **S**parsity **E**mbiggens **L**earning) system in Morpho Challenge 2010, which attained impressive performance. The

MORSE system is fairy simple, it learns the transformation rules from minimal word-pairs in training data by updating repeatedly Base, Derived, and Unmodeled word sets. Base word set is the set consists of stems that the system has predicted so far. Derived word set is the set of words that can be derived from Base by applying learned transformation rules. Unmodeled word set is the set of words that have not been moved to Base and Derived word sets yet.Lignos (2010) employed the compounding model of Koehn & Knight to refine the set of learned morphemes $S = \{\sigma_1, ..., \sigma_n\}$

$$\arg\max_{S} \left(\prod_{\sigma_i \in S} \text{count}(\sigma_i) \right)^{\frac{1}{n}} \tag{2.6}$$

Algorithm 1 MORSE algorithm

Add all the words to Unmodeled word set.
for $t = 1 \to T$ **do**
  Score suffixes and transformation rules and select the best transformation rules
  Move the words used in selected transfrom
  Performing Base Inference, inferring new bases and adding them the learned transfroms
  Optionally perform compounding for the current iteration
end for
Optionally perform compounding after learning is complete

Poon *et al.* (2009) proposed a log-linear model that could incorporate simple exponential priors inspired by MDL, and overlapping features. The key component of the model is a morpheme-context model, which can capture rich segmentation regularities by looking at the context patterns. Context of a morpheme is represented using n-grams before and after that morpheme, for some constant n. For instance, Arabic word `w-vlAv-wn` (hyphens indicate morpheme boundaries) has three bigram context features `##_vl`, `#w_wn`, and `Av_##` corresponding to three morphemes `w`, `vlAv`, and `wn` respectively. Formally, the model defines a joint probability distribution over a set of types[1] W and a segmentation S as follow:

$$P_\theta(W, S) = \frac{1}{Z} u_\theta(W, S) \tag{2.7}$$

[1]Authors reported that in their experiment, learning and inference using word types give better result than using tokens.

where Z is the normalizing constant and

$$u_\theta(W,S) = \exp\left(\sum_\sigma \lambda_\sigma f_\sigma(S) + \sum_c \lambda_c f_c(S) + \alpha \cdot \sum_{t\in\{-,0,+\}} \sum_{\sigma\in L_t} l(\sigma) + \beta \sum_{w\in W} \frac{s(w)}{l(w)}\right) \tag{2.8}$$

in which, σ is a morpheme string; c is a morpheme-context; L_-, L_0, and L_+ are sets of prefix, stem, and suffix lexicons induced by S; $l(w)$ denotes length of a string w; $s(w)$ denotes number of morphemes in w given S.

Poon *et al.* (2009) used `DELORTRANS1` (deleting any character or transposing any pair of adjacent characters) to obtain a set of neighborhoods of the observed data. These neighborhoods served as pseudo-negative examples to move probability mass to the observed data using contrastive estimation Smith & Eisner (2005).

While log-linear model has been successfully applied for Arabic language, reducing F1 error by 11% compared to Morfessor, it does not make use of the connection between part-of-speech (POS) categories and morphological properties. Lee *et al.* (2011) proposed a generative model which utilized this tight connection without assuming access to full-fledged syntactic information. This model captured two aspects of the morpho-syntactic connection:

- Morphological consistency within POS categories. Words that belong to the same syntactic category tend to have similar affixes.
- Morphological realization of grammatical agreement. Grammatical agreement can be expressed via correlated morphological markers. In Penn Arabic treebank corpus, exact suffix matching of adjacent words has 94% precision at the token-level.

Since the work in this thesis is based on this model, I will spend chapter 4 to go into technical details of the model.

The review would not be completed without mentioning the model proposed by Goldwater *et al.* (2006). This model extends standard generative models with an adaptor that captures one of the most striking properties of natural languages: the power-law distribution in the frequencies of word tokens or Zipf's law. The model, which is referred as a two-stage language model, contains a generator and an adapter. The generator generates words by first, generating inflectional class for the words then, stems and suffixes are generated conditionally on the class. The adapter produces the power-law distribution using Pitman-Yor process Pitman & Yor (1997). Operating on tokens level, this model allows different tokens of the same type to have different analyses.

A larger body of work in unsupervised learning recently devotes to unsupervised multilingual learning. It has been showed that unsupervised multilingual learning has pushed the state-of-the-art in language technology to new limits Snyder & Barzilay (2010). The key idea of unsupervised multilingual learning is to explore the deep links among human languages. A common approach for multilingual learning is to use knowledge of source languages to guide learning algorithm on target languages. The knowledge can be transferred through heuristic "projection" Yarowsky & Ngai (2001) or constraints in learning Das & Petrov (2011); McDonald *et al.* (2011); Naradowsky & Toutanova (2011); Täckström *et al.* (2012) or inference Cohen *et al.* (2011). Another direction of research in unsupervised multilingual learning is to learn a joint model exploiting hypothesis that cross-lingual variations in linguistic forms correspond to systematic variations in ambiguity Snyder & Barzilay (2008); Snyder *et al.* (2008, 2009).

In unsupervised morphological learning, Snyder & Barzilay (2008) modeled both abstract morphemes (cross-lingual morpheme patterns) as well as stray morphemes (morphemes that appear in one language without their counterparts in other language) using a hierarchical Bayesian model. Given a parallel corpus, a distribution $\mathcal{A}$ over bilingual morpheme pairs, a distribution $\mathcal{E}$, and a distribution $\mathcal{F}$ over stray morphemes in each language are drawn from Dirichlet processes. To find the set of morphemes which yields a high joint probability, Snyder & Barzilay (2008) performed Gibbs sampling over all possible draws of the distributions $\mathcal{A}$, $\mathcal{E}$, and $\mathcal{F}$. This model not only can induce morpheme segmentations for each language but also can discover abstract bilingual morphemes like (`un`, `ne`) for English-Czech language pair or (`im`, `un`) for English-German language pair[1].

Treating morphological analysis as a structured prediction problem, Kim *et al.* (2011) defined a morphological space, in which each language is resided as a datapoint. They employed a fairy simple set of morphological features for any labeled language:

- Number of unique stems.
- Number of unique suffixes.
- Number of unique deletion rules. There are three type of deletion rules: deletion of final vowels (`..V#` $\rightarrow$ `..#`), deletion of penultimate vowels (`..VC#` $\rightarrow$ `..C#`), and removals or additions of final accent marks (e.g. `..š#` $\rightarrow$ `...s#`).
- Entropy of stems.
- Entropy of suffixes.
- Entropy of deletion rules.

[1] I implemented this idea in my model using Chinese Restaurant Process, however, it is only good at finding bilingual abstract prefixes.

- Percentage of unsegmented word types.
- Percentage of segmented word types which employ a deletion rule.

Given annotated languages serving as training examples, Kim *et al.* (2011) developed a structured nearest neighbor prediction method which searches for the best morphological analysis for each unlabeled language by minimizing its distance to each of the training languages. The limitation of this method is that currently it only works for nominal inflectional suffix morphology, on which a small set of deletion rules can apply[1].

2.2 Computational preliminaries

In this section I review some basic ideas of Bayesian inference, particularly focusing on two important prior distributions, namely, Multinomial distribution and Dirichlet distribution. These distributions play a crucial role in simplifying the inference formula, which makes it easier for sampling algorithms such as Gibbs sampling. I will establish a short-cut for sampling equations used later on by deriving a generic formula for a joint distribution which takes Multinomial distribution as prior.

2.2.1 Bayesian Inference

In any generative model, Bayesian inference plays an important part for updating beliefs about latent variables given observed data. At the heart of Bayesian inference is Bayes rule:

$$P\left(h|d\right) = \frac{P\left(d|h\right)P\left(h\right)}{\sum_{h' \in H} P\left(d|h'\right)P\left(h'\right)} \propto P\left(d|h\right)P\left(h\right) \tag{2.9}$$

$P(h)$ is the *prior probability* which encodes the learner's degree of belief in a hypothesis without any knowledge of the observations. $P(h|d)$ is the *posterior probability*, which measures how expected the data are under hypothesis h , relative to all other hypotheses h' in hypothesis space $\mathcal{H}$.

The goal of learning is to select the most probable hypothesis $\hat{h}$ given the observed data. In case prior knowledge is not provided, Maximum-Likelihood estimation (MLE) is a common method that used to select such a hypothesis $\hat{h}$.

MLE assumes that all hypotheses are equally probable a priori, then the posterior probability (probability of a hypothesis h given the observed data d)

[1] I tried to reproduce their experiment, but at the step of computing morphological features, my result is far closer to what they reported in their paper.

is proportional to the likelihood $P(d|h)$. Learning hypothesis $\hat{h}$ is equivalent to choosing the single hypothesis with the highest likelihood:

$$\hat{h} = \arg\max_{h} P\left(d|h\right) \tag{2.10}$$

In context of unsupervised learning, many successful generative models have imposed strong constraints on the priors. Successful generalization depends on taking the right constrains. A often-used constraint is Minimum Description Length (MDL) principle, a mathematical formalization of Occam's Razor which favors simpler hypotheses over more complex ones. Under prior constraints, Maximum a Posteriori (MAP) is a method that provides a principled way to compare hypotheses with different numbers of parameters, and to select the most probable one:

$$\hat{h} = \arg\max_{h} P\left(d|h\right) P\left(h\right) \tag{2.11}$$

2.2.2 Conjugate Priors

MDL constraint has been used successfully in many unsupervised learning tasks, especially in unsupervised morphological learning as I mentioned earlier. However, the choice of prior constraints can greatly affect the complexity of the models. Within Bayesian statistics, certain kinds of distributions have been widely used as priors because of their convenient mathematical properties. To illustrate some of these properties, and to prepare for the presentation of the model in this thesis, I will take Dirichlet distribution, a prior over categorical as the example.

Consider a random variable that can take on one of K possible outcomes $\{1, ..., K\}$, in which the probability of outcome $k \in \{1, ..., K\}$ is θ_k. Let $\boldsymbol{x} = \{x_1, ..., x_n\}$ be the set of outcomes sampled from this categorical distribution (i.e. $x_i \in \{1, ..., K\}$ and $P(x_i = k) = \theta_k$). This can be expressed as follows:

$$x_i|\boldsymbol{\theta} \sim Cat(\boldsymbol{\theta}) \tag{2.12}$$

where $\boldsymbol{\theta} = \{\theta_1, .., \theta_k\}$

The Dirichlet prior is a distribution over parameter space $\boldsymbol{\theta}$. Using a Dirichlet prior over a categorical distribution thus gives a model:

$$x_i|\boldsymbol{\theta} \sim Cat(\boldsymbol{\theta}) \tag{2.13}$$

$$\boldsymbol{\theta}|\boldsymbol{\beta} \sim Dir(\boldsymbol{\beta}) \tag{2.14}$$

Recall the difinition of the Dirichlet distribution:

$$P(\boldsymbol{\theta}|\boldsymbol{\beta}) = \frac{1}{\mathrm{B}(\boldsymbol{\beta})} \prod_{k=1}^{K} \theta_k^{\beta_k - 1}$$

with

$$\mathrm{B}(\boldsymbol{\beta}) = \frac{\prod_{k=1}^{K} \Gamma(\beta_k)}{\Gamma\left(\sum_{k=1}^{K} \beta_k\right)}$$

where $\beta_k > 0$. $\mathrm{B}(\boldsymbol{\beta})$ is the normalizing constant, which is expressed in terms of the Gamma function $\Gamma(z) = \int_0^\infty t^{z-1} e^{-t} dt$ for $z > 0$.

Using Bayes' rule to estimate the underlying parameter $\boldsymbol{\theta}$ of the categorical distribution given a collection of n samples $\{x_1, ..., x_n\}$:

$$\begin{aligned} P(\boldsymbol{\theta}|\boldsymbol{x}, \boldsymbol{\beta}) &\propto P(\boldsymbol{x}|\boldsymbol{\theta}) P(\boldsymbol{\theta}|\boldsymbol{\beta}) \\ &\propto \prod_{i=1}^{n} P_{\boldsymbol{\theta}}(x_i) \prod_{k=1}^{K} \theta_k^{\beta_k - 1} \\ &= \prod_{k=1}^{K} \theta_k^{n_k} \prod_{k=1}^{K} \theta_k^{\beta_k - 1} \\ &= \prod_{k=1}^{K} \theta_k^{n_k + \beta_k - 1} \end{aligned} \tag{2.15}$$

Choosing Dirichlet distribution $Dirichlet(\boldsymbol{\beta})$ as the prior over categorical parameters leads the posterior $P(\boldsymbol{\theta}|\boldsymbol{x}, \boldsymbol{\beta})$ to having the form of another Dirichlet distribution, with parameters $n_k + \beta_k$. A prior is called *conjugate prior* for the likelihood if the posterior distribution is in the same analytical form as the prior probability distribution.

2.2.3 Point estimation

MAP estimate, as discussed in 2.2.1, of the posterior in equation 2.15 results in

$$\theta_k = \frac{n_k + \beta_k - 1}{n + \sum_{k=1}^{K} (\beta_k - 1)} \tag{2.16}$$

The form of equation 2.16 is equivalent to the maximum likelihood estimate of $\boldsymbol{\theta}$ with observed counts $\{n_1 + \beta_1 - 1, ..., n_K + \beta_K - 1\}$.

Goldwater (2007) pointed out a problem with MAP when any β_k is less than one. Follow the example in Goldwater (2007), assume that we are interested in

Table 2.1: A toy probabilistic grammar

θ_x	$S \to X$
θ_y	$S \to Y$
$1 - \theta_x - \theta_y$	$S \to B$
1	$X \to a$
1	$Y \to a$
1	$B \to b$

learning syntactic rule probabilities for parsing. Data d contains only two strings a and b, probabilistic grammar rules are given in table 2.1.

We initialize all production rules with uniform probability $\theta_x = \theta_y = \frac{1}{3}$ and use symmetric Dirichlet prior for $\boldsymbol{\theta} = (\theta_x, \theta_y)$ by setting $\beta_x = \beta_y = \beta = 0.2$. Expectation-Maximization (EM) computes the expected counts n_x and n_y for rules $S \to X$ and $S \to Y$ are both 0.5, and the expected count n_b for rule $S \to B$ is 1. From equation 2.15, posterior probability of $\boldsymbol{\theta}$ given data d and hyperparameter β is proportional to:

$$\begin{aligned} P(\theta|d,\beta) &\propto \theta_x^{n_x+\beta-1}\theta_y^{n_y+\beta-1}(1-\theta_x-\theta_y)^{n_b+\beta-1} \\ &= \theta_x^{-0.3}\theta_y^{-0.3}(1-\theta_x-\theta_y)^{0.2} \end{aligned} \tag{2.17}$$

This posterior probability function is maximized when $\theta_x \to 0$ and $\theta_y \to 0$. Thus, it makes the string a unparseable.

2.2.4 Inference via sampling

The drawback of point estimation methods is that they simply disregard the knowledge about a whole distribution. As an example, assume that we want to predict the outcome for a new observation x_{n+1} in 2.2.2 using posterior information 2.15. The conditional distribution of x_{n+1} given all previous observations is derived by integrating over all possible values of $\boldsymbol{\theta}$:

$$\begin{aligned}
P(x_{n+1}=j|\boldsymbol{x},\boldsymbol{\beta}) &= \int_{\Delta} P(x_{n+1}=j|\boldsymbol{\theta})P(\boldsymbol{\theta}|x,\beta)d\boldsymbol{\theta} \\
&= \int_{\Delta} \theta_j \frac{\Gamma\left(n+\sum_{k=1}^{K}\beta_k\right)}{\prod_{k=1}^{K}\Gamma\left(n_k+\beta_k\right)} \prod_{k=1}^{K} \theta_k^{n_k+\beta_k-1} d\boldsymbol{\theta} \\
&= \frac{\Gamma\left(n+\sum_{k=1}^{K}\beta_k\right)}{\prod_{k=1}^{K}\Gamma\left(n_k+\beta_k\right)} \int_{\Delta} \theta_j^{n_j+\beta_j} \prod_{k\neq j} \theta_k^{n_k+\beta_k-1} d\boldsymbol{\theta} \\
&= \frac{\Gamma\left(n+\sum_{k=1}^{K}\beta_k\right)}{\prod_{k=1}^{K}\Gamma\left(n_k+\beta_k\right)} \times \frac{\Gamma\left(n_j+\beta_j+1\right)\prod_{k\neq j}\Gamma\left(n_k+\beta_k\right)}{\Gamma\left(n+1+\sum_{k=1}^{K}\beta_k\right)} \\
&= \frac{n_j+\beta_j}{n+\sum_{k=1}^{K}\beta_k}
\end{aligned} \tag{2.18}$$

where Δ denotes the probability simplex, i.e. the set of all possible $\boldsymbol{\theta}$ such that $\theta_1 + ... + \theta_K = 1$.

We integrate out $\boldsymbol{\theta}$, in the final formula, there is no more $\boldsymbol{\theta}$. I briefly explain how maths works cleanly in 2.18. From 2.15 we know $P(\boldsymbol{\theta}|\boldsymbol{x},\boldsymbol{\beta}) = c\prod_{k=1}^{K}\theta_k^{n_k+\beta_k-1}$, and because $P(\boldsymbol{\theta}|\boldsymbol{x},\boldsymbol{\beta})$ is the form of a Dirichlet distribution, so we know the value of normalizing constant c. Applying the property of Dirichlet distribution:

$$\int_{\Delta} \prod_{k=1}^{K} \theta_k^{\beta_k-1} d\theta = \mathrm{B}(\boldsymbol{\beta})$$

where $\sum_{k=1}^{K}\theta_k = 1$, we have:

$$\int_{\Delta} \theta_j^{n_j+\beta_j} \prod_{k\neq j} \theta_k^{n_k+\beta_k-1} d\boldsymbol{\theta} = \frac{\Gamma\left(n_j+\beta_j+1\right)\prod_{k\neq j}\Gamma\left(n_k+\beta_k\right)}{\Gamma\left(n+1+\sum_{k=1}^{K}\beta_k\right)}$$

The last line of 2.18 is obtained by using the property of Gamma function $\Gamma(x+1) = x\Gamma(x)$.

Equation 2.18 with hyperparameters β_k allows x_{n+1} can select any outcome $k \in \{1, 2, ..., K\}$ where the most probable outcome has highest probability and the most improbable outcome has lowest non-zero probability.

In general, dealing with the whole distribution, we are interested in calculating the expected value of a function $f(z)$, where z is a random variable.

$$E\left[f\left(z\right)\right] = \int f\left(z\right) p\left(z\right) dz \tag{2.19}$$

In Bayesian inference, often $p(z)$ is the prior probability and $f(x)$ is the likelihood function. We can rewrite 2.19 as:

$$E_{p(z)}\left[f\left(z\right)\right] = \lim_{N \to \infty} \frac{1}{N} \sum_{t=1}^{N} f(z^{(t)}) \tag{2.20}$$

In practice, we approximate 2.20 by sampling only finite number of times, T:

$$E_{p(z)}\left[f\left(z\right)\right] \approx \frac{1}{T} \sum_{t=1}^{T} f(z^{(t)}) \tag{2.21}$$

Now, the crucial point is to get sample $z^0, z^1, ..., z^T$ from distribution $p(z)$. We need a function g that walks through probabilistic space, and at state z^t it walks to the next state $z^{t+1} := g(z^t)$ with probability $P_{trans}\left(z^{(t+1)} | z^{(0)}, z^{(1)}, ..., z^{(t)}\right)$. For simplicity, we use Markov property:

$$P_{trans}\left(z^{(t+1)} | z^{(0)}, z^{(1)}, ..., z^{(t)}\right) = P_{trans}\left(z^{(t+1)} | z^{(t)}\right) \tag{2.22}$$

In the following, I will discuss Gibbs sampling, a technique that allows us to design such a function g.

2.2.5 Gibbs sampling

We want to approximate equation 2.21 by sampling $z^{(0)}, z^{(1)}, z^{(1)}, ..., z^{(T)}$ according to $p(z)$. Let z be a point in $K > 1$ dimensions. The basic idea of Gibbs sampling is walking to the next state in K dimensions by making a probabilistic choice for each of the K dimensions, where each choice depends on the other $K - 1$ dimensions.

$$P(Z_i | z_1^{(t+1)}, ..., z_{i-1}^{(t+1)}, z_{i+1}^{(t)}, ..., z_K^{(t)}) = \frac{P(z_1^{(t+1)}, ..., z_{i-1}^{(t+1)}, z_i^{(t)}, z_{i+1}^{(t)}, ..., z_K^{(t)})}{P(z_1^{(t+1)}, ..., z_{i-1}^{(t+1)}, z_{i+1}^{(t)}, ..., z_K^{(t)})} \tag{2.23}$$

The point $z^{(t+1)} = g(z^{(t)})$ is computed as $\left\langle z_1^{(t+1)}, ..., z_K^{(t+1)} \right\rangle$.

Algorithm 2 Gibbs sampling algorithm

```
z^(0) := ⟨z_1^(0), ..., z_K^(0)⟩
for t = 1 → T do
    for i = 1 → K do
        z_i^(t+1)
    end for
end for
```

2.2.6 Maximum Marginal Decoding

Typically, the output of the algorithm is the last sample in a stream of samples from the posterior distribution produced by Gibbs sampler. Because Gibbs sampler makes a probabilistic choice for each state, it might introduce variace and noise in its output. *Maximum marginal decoding* (MM) is a technique which assigns to each latent variable the value with the highest marginal probability, thus MM maximizes the expected number of correct assignments and reduces noise. Stallard *et al.* (2012) applied MM for the model of Lee *et al.* (2011) and obtained state-of-the-art unsupervised morphological segmentation for Arabic. They found that MM not only dramatically reduces the output variance of Gibbs sampling but also reduces noise from spurious affixes when the model is trained on a large corpus.

MM algorithm is quite straightforward: Draw N independent Gibbs samplers, and for each word type, select the most frequent segmentation.

Chapter 3

Evaluation Metrics

3.1 Evaluation for unsupervised morphological segmentation

One difficulty in evaluating morphological segmentation is that unsupervised systems usually decompose word into morphemes while gold standard contains full analysis. To illustrate this point, take "`knives`" as an example of a word that needs to be segmented. Since unsupervised systems do not have access to linguistically motivated morpheme labels as well as language-specific knowledge, they typically cut the word into morphemes without modifying any morpheme in the result. Such a system often decomposes "`knives`" as "`kniv - es`" instead of the conventional analysis "`knife_N + Plural`", in gold standard. Nevertheless, most recent papers have used Precision, Recall, and F-measure to evaluate performance of unsupervised systems. Two evaluation methods are proposed, one compares directly the proposed segmentation, while the other compares indirectly. We describe both methods in following subsections.

3.1.1 Morpho Challenge Evaluation

Creutz & Lagus (2005b) used precision, recall, and the harmonic mean F-measure to evaluate on discovered morpheme boundaries. Precision is the fraction of correctly discovered morpheme boundaries in all discovered morpheme boundaries by the algorithm. Recall is the fraction of correctly discovered morpheme boundaries in all suggested morpheme boundaries. F-measure is given by:

$$\text{F-measure} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{3.1}$$

These measures are widely used to evaluate performance of unsupervised morphological segmentation algorithms. They are used to compare the result of participants in Morpho Challenge 2005[1], 2007[2], 2008[3], 2009[4], and 2010[5], a series of workshops on semi-supervised and unsupervised methods for morphological analysis.

In Morpho Challenge, the result is evaluated on a sample of a large number of word pairs, where both words in a word pair share at least one gold standard morpheme in common. A system which has highest F-measure is the best system.

- *Precision* is calculated as follows: A number of word forms will be sampled from the result file such that for each morpheme in these words, another word having the same morpheme will be chosen randomly if such a word exists. Hence, we obtain a number of word-pairs, such that two words in a word-pair share at least one morpheme in common. These word-pairs will be compared against gold standard. We give one point for a correct word-pair, and the final point for each sampled word form is normalized to one. Precision is then computed by taking the total number of points divided by the total number of sampled words. For example, assume that the proposed analysis of the word "`abyss`" is "`abys - s`". By sampling the result file, assume that we find "`abys - s - es`" and "`mountain - s`" which share morpheme "`abys`" and "`s`" with "`abys - s`" respectively. According to gold standard, the correct analyses of these words are "`abyss_N`", "`abyss_N + PL`", and "`mountain_N + PL`". The pair "`abys - s, abys - s - es`" is correct (common `abyss_N`), but the pair "`abys - s, mountain - s`" is incorrect (no common morpheme in gold standard). Thus precision for the word "`abyss`" is $1/2 = 50\%$.

- *Recall* is calculated analogously to precision with word forms randomly sampled from gold standard.

In order to compare our results, we adopt the evaluation procedure used in Morpho Challenge.

3.1.2 EMMA

Spiegler & Monson (2010) proposed an alternative evaluation called EMMA[6] (an

[1] http://www.cis.hut.fi/morphochallenge2005/
[2] http://www.cis.hut.fi/morphochallenge2007/
[3] http://www.cis.hut.fi/morphochallenge2008/
[4] http://www.cis.hut.fi/morphochallenge2009/
[5] http://research.ics.aalto.fi/events/morphochallenge2010/
[6] The script is availabe to download at http://www.cs.bris.ac.uk/Research/MachineLearning/Morphology/

Evaluation **M**etric for **M**orphological **A**nalysis), which has been used in Morpho Challenge 2010.

The key idea of EMMA is that it does not directly compare discovered and answer analyses, instead, it seeks a one-to-one relabeling of discovered morphemes that renders them as similar as possible to the answer. The final measures (Precision, Recall, and F-measure) are then computed on the approximated isomorphism. To achieve this goal, EMMA finds the optimal maximum matching in a bipartite graph $\mathcal{G} = \{\mathcal{D}, \mathcal{A}; \mathcal{E}\}$, where $\mathcal{D}$ is the set of all unique morphemes in discovered analysis, $\mathcal{A}$ is the set of all unique morphemes in the answer analyses, and the set of edges $e(d_i, a_j) \in \mathcal{E}$ such that each edge has one vertex in $\mathcal{D}$ and the other in $\mathcal{A}$.

A *maximum matching* $\mathcal{M} \subset \mathcal{E}$ is a matching where there is no other $\mathcal{M}' \subset \mathcal{E}$ such that $|\mathcal{M}'| > |\mathcal{M}|$. Let $w(d_i, a_j)$ be the weight assigned to the edge $e(d_i, a_j) \in \mathcal{E}$. The goal of EMMA is to find such an optimal assignment $\mathcal{M}$ satisfying:

$$\mathcal{M} = \arg\max_{\mathcal{M}} \sum_{e(d_i,a_j)\in\mathcal{M}} w(d_i, a_j) \tag{3.2}$$

Given a maximum matching optimal assignment $\mathcal{M}$ of discovered and answer morphemes, EMMA computes *Precision*, *Recall*, and *F-measure* as follows:

Let w_k be the k^{th} word in vocabulary V. Let $D_{k,r}$ be the r^{th} discovered analysis of w_k with $1 \leq r \leq m_k$, and let $A_{k,s}$ be the s^{th} answer analysis of w_k with $1 \leq s \leq n_k$. Furthermore, let $D^*_{k,r}$ denote the set of discovered morphemes of r^{th} analysis for word w_k, in which a morpheme $d_{i,r}$ is replaced by a morpheme $a_{j,s}$ if $e(d_{i,r}, a_{j,s}) \in \mathcal{M}$.

$$\begin{aligned}
\text{Precision} &= \frac{1}{|V|}\sum_{k}^{|V|}\frac{1}{m_k}\sum_{s}^{n_k}\sum_{r}^{m_k} b_{r,s}\frac{|A_{k,s}\cap D^*_{k.r}|}{|D^*_{k.r}|} && (3.3)\\
\text{Recall} &= \frac{1}{|V|}\sum_{k}^{|V|}\frac{1}{n_k}\sum_{s}^{n_k}\sum_{r}^{m_k} b_{r,s}\frac{|A_{k,s}\cap D^*_{k.r}|}{|A_{k.s}|} && (3.4)\\
\text{F-measure} &= \frac{2\cdot\text{Precision}\cdot\text{Recall}}{\text{Precision}+\text{Recall}} && (3.5)
\end{aligned}$$

where $b_{r,s} = 1$ if the assignment between $D_{k,r}$ and $A_{k,s}$ is found in $\mathcal{M}$, otherwise, $b_{r,s} = 0$.

3.2 Statistical significance testing

Using evaluation metric like *F-measure* to compare two systems is not enough. When one system appears to outperform the other, we want to know whether the improvement is real or it just happens by chance. Statistical significance tests give us a systematic way of quantifying the probability that the observed increase in the test score on a test set is due to luck. If that probability is low, we believe that the improvement is real, if it is high, either there is no improvement, or the data are insufficient to reflect the true improvement in system quality.

3.2.1 Hypothesis tests

When comparing a new system A to a baseline system B, we want to know if A outperforms B on some large population of data given that A wins B by a metric gain $\delta(x)$ on a small sample test set $x = x_1, ..., x_n$. Hypothesis testing guards against the case that the victory of A over B is due merely to chance. The particular hypothesis to be tested is called the *null hypothesis*, denoted H_0, which assumes that A *is no better than* B *on the population as a whole*. The ultimate goal of hypothesis testing is to accept or reject H_0 by estimating this likelihood, written $p(\delta(X) > \delta(x)|H_0)$, where X is a random variable over possible test sets of size n that we could have drawn, and $\delta(x)$ is a constant, the observed metric gain. Small value of $p(\delta(X) > \delta(x)|H_0)$ suggests the null hypothesis is false. We refer to $p(\delta(X) > \delta(x)|H_0)$ as p-value(x). Typically p-value(x) < 0.05 is considered "sufficiently good" to reject H_0.

In most cases p-value(x) is not easy to compute and must be approximated. Among various approximation schemes, paired-bootstrap Efron & Tibshirani (1993) is one of the most widely used in NLP community Berg-Kirkpatrick *et al.* (2012); Bisani & Ney (2004); Koehn (2004); Och (2003). Berg-Kirkpatrick *et al.* (2012) demonstrated that paired-bootstrap can be applied to a range of NLP tasks including text summarization, dependency parsing, machine translation, word alignment, and constituency parsing. Koehn (2004) showed that bootstrap can give us assurances that the differences between two translation systems is real even with only 300 sentences as test data.

3.2.2 The Bootstrap

The bootstrap draws many simulated test sets $x^{(i)}$ from x by sampling n items from x with replacement for each $x^{(i)}$, then it approximates p-value(x) by counting how often A beats B at least by $\delta(x)$ in sample test sets $x^{(i)}$. Algorithm 3 describes the bootstrap procedure used in Berg-Kirkpatrick *et al.* (2012).

There is a little bit difference in algorithm 3 compared to the algorithm used in Koehn (2004). Koehn (2004) increased counter s under condition $\delta(x^{(i)}) < 0$. As explained in Berg-Kirkpatrick *et al.* (2012), sample $x^{(i)}$ are drawn from x, so the mean of $\delta(x^{(i)})$ will be around $\delta(x)$. Therefore, system A will beat system B on about half of $x^{(i)}$. The solution for this problem is re-centering of the

Algorithm 3 The bootstrap procedure

Draw b bootstrap samples $x^{(i)}$ of size n by sampling with replacement from x.
Initialize $s = 0$.
for $i = 1 \rightarrow b$ **do**
  if $\delta(x^{(i)}) > 2\delta(x)$ **then** $s = s + 1$
  end if
end for
Estimate p-value$(x) \approx \frac{s}{b}$

mean: how often A does *more than* $\delta(x)$ *better than expected*. Thus, the condition $\delta(x^{(i)}) > 2\delta(x)$ comes from the fact that we expect A beats B by $\delta(x)$. Berg-Kirkpatrick *et al.* (2012) also noted that if the mean of $\delta x^{(i)}$ is $\delta(x)$, and if the distribution of $\delta x^{(i)}$ is symmetric, then these two versions will be equivalent.

Chapter 4

Unsupervised Morphological Segmentation

4.1 Modeling Syntax in Unsupervised Morphological Segmentation

In this section I review the state of the art unsupervised morphological segmentation model proposed by Lee *et al.* (2011). I also reimplement their model and perform a set of experiments and evaluate the results of the model on 4 languages: English, Turkish, Tamil, and Telugu.

Lee *et al.* (2011) introduced a model for unsupervised morphological segmentation that captures two prominent linguistic relations between morphology and syntax.

1. Morphological consistency within POS categories.

2. Morphological realization of grammatical agreement.

The former morpho-syntax relation captures the intuition that words belonging to the same syntactic category tend to choose similar affixes. The later relation holds for certain languages, for example in Arabic, the grammatical agreement is commonly realized using matching suffixes, for example bigrams *(adjective, noun)* in Arabic often have the same ending. While this assumption may not hold for other languages, I still describe it in this section.

4.1.1 High-level generative story

Given a corpus of unannotated and unsegmented sentences as input, the model provides a generative story explaining how the corpus was probabilistically created. The model consists of four components:

1. **Lexicon Model** generates morpheme lexicon $\mathbf{L}$ using parameters $\boldsymbol{\gamma}$. Set of lexicon $\mathbf{L}$ consists of three separate subsets: prefixes, stems, and suffixes which are generated in a hierachical fashion.

2. **Segmentation Model** generates word-types $\mathbf{W}$, their segmentations $\mathbf{S}$, and their syntactic categories $\mathbf{T}$ conditionally on $\mathbf{L}$.

3. **Token-POS Model** generates unsegmented tokens $\mathbf{w}$ and their parts-of-speech $\mathbf{t}$ from standard first-order HMM.

4. **Token-Seg Model** generates token segmentations $\mathbf{s}$ from a first-order Markov chain that has dependencies between adjacent segmentations.

The complete picture of this generative story is given in the following equation:

$$
\begin{aligned}
P(\mathbf{w}, \mathbf{s}, \mathbf{t}, \mathbf{W}, \mathbf{S}, \mathbf{T}, \mathbf{L}, \boldsymbol{\Theta}, \boldsymbol{\theta} | \boldsymbol{\gamma}, \boldsymbol{\alpha}, \boldsymbol{\beta}) \;=\; & P(\mathbf{L}|\boldsymbol{\gamma}) && (4.1)\\
& P(\mathbf{W}, \mathbf{S}, \mathbf{T}, \boldsymbol{\Theta} | \mathbf{L}, \boldsymbol{\gamma}, \boldsymbol{\alpha}) && (4.2)\\
& P_{pos}(\mathbf{w}, \mathbf{t}, \boldsymbol{\theta} | \mathbf{W}, \mathbf{S}, \mathbf{T}, \mathbf{L}, \boldsymbol{\alpha}) && (4.3)\\
& P_{seg}(\mathbf{s} | \mathbf{W}, \mathbf{S}, \mathbf{T}, \mathbf{L}, \boldsymbol{\beta}, \boldsymbol{\alpha}) && (4.4)
\end{aligned}
$$

where $\boldsymbol{\gamma}$, $\boldsymbol{\Theta}$, $\boldsymbol{\theta}$, $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$ are hyperparameters whose roles will be explained shortly.

4.1.2 Submodels and sampling equations

Now I will describe these four components of the model in details, and derive the sampling equation for each of them.

4.1.2.1 Lexicon Model

Lexicon Model is designed to encode MDL constraint as the priors. It prefers short morphemes and a compact set of morpheme lexicon $\mathbf{L}$. First, it draws each morpheme σ in the master lexicon $\mathbf{L}^*$ according to geometric distribution.

$$|\sigma| \sim \text{Geometric}(\gamma_l)$$

where hyperparameter γ_l is specified beforehand.

The choice of the distribution depends on our knowledge of the languages. For example, morphemes in Telugu often have 2 to 8 characters, thus, we can choose gamma distribution i.e. $|\sigma| \sim \text{Gama}(k, \theta)$ (Figure 4.1) instead of geometric distribution to encode this knowledge.

Having master lexicon L^*, then lexicon model draws sets of morphemes for the prefix L_-, the stem L_0, and the suffix L_+ lexicons from morphemes in L^*. By

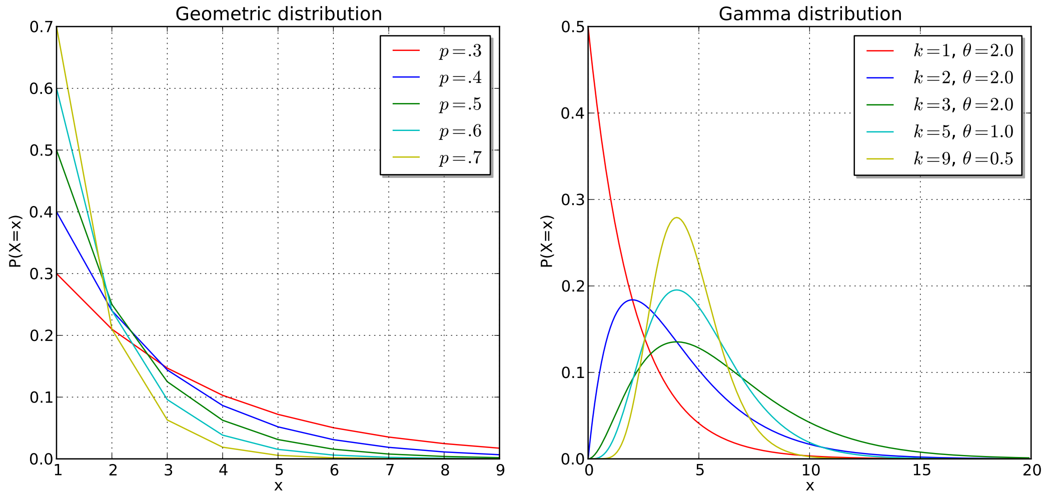

Figure 4.1: Geometric distribution and gamma distribution as the choice of priors

this hierarchical design, the morphemes can be shared among the lower-level lexicons. Therefore, the model also works for compound words. Technically speaking, assume that we allow only one stem in a word, if the morpheme "`moon`" is generated in L^*, then it can be used to generate suffixes or prefixes for "`moonshine`", "`moonstruck`", "`moonwalk`" and so forth. So far, the model biases toward short morphemes, to favor compact lexicons, model assigns lower probability to bigger morpheme set. This can be done using geometric distribution again:

$$\begin{aligned} \text{prefix}: &\quad |L_-| \sim \text{Geometric}(\gamma_{l_-}) \\ \text{stem}: &\quad |L_0| \sim \text{Geometric}(\gamma_{l_0}) \\ \text{suffix}: &\quad |L_+| \sim \text{Geometric}(\gamma_{l_+}) \end{aligned}$$

Let (S, T) denote the hypothesis that segments word-type Wi with segmentation S and tags it with POS tag T. Let $\mathbf{L} = (L^*, L_-, L_0, L_+)$ be the minimal lexicon under this hypothesis. The probability of hypothesis $(S, T, s = S, t = T, \boldsymbol{L})$ is proportional to:

$$\prod_{\sigma \in L^*} \gamma_l (1-\gamma_l)^{|\sigma|} \times \gamma_-(1-\gamma_-)^{|L_-|} \times \gamma_0(1-\gamma_0)^{|L_0|}\gamma_+(1-\gamma_+)^{|L_+|} \tag{4.5}$$

Starting with every word-type as a morpheme, if a hypothesis introduces a new morpheme σ_- as a suffix it has to pay an additional cost $(1-\gamma_-)\times\gamma_l(1-\gamma_l)^{|\sigma_-|}$ compared to the hyhothesis that introduces none.

In practice, we assign $\gamma_0 \ll min\{\gamma_-, \gamma_+\}$. By doing this, we capture the fact that the set of prefixes and suffixes are much smaller than the set of stems.

To sum up, the model penalizes hypothesis for increasing the size of lexicons while encouraging it to make a reasonable segmentation.

4.1.2.2 Segmentation Model

Segmentation Model captures the agreement between morphology and syntactic class. The model generates each word-type independently using morphemes in stem and affix lexicons, such that each word-type has only one stem and affixes attached to the stem are generated conditionally on the syntactic classes. In their preliminary experiments, Lee *et al.* (2011) found that the model performed worst when stems are generated conditioned on the tag. Lee *et al.* (2011) argued that the connection between affixes and POS tag is stronger than the connection between stems and POS tag. In the following, I describe the generative process in the segmentation model.

First, the model generates categorical distribution parameters for the POS tag from symmetric Dirichlet prior:

$$\Theta_T \sim \text{Dirichlet}(\alpha_T, \{1, ..., K\})$$

where α_T is the concentration parameter and K is the number of tags, which is fixed and set beforehand.

For each tag $T \in \{1, ..., K\}$, the model generates parameters for categorial distribution from Dirichlet prior for the prefix and suffix lexicons. Categorical distribution parameters for stem lexicon are generated (from symmetric Dirichlet prior) independently from tag T:

$$\begin{aligned}
\Theta_{-|T} &\sim \text{Dirichlet}(\alpha_-, L_-) \\
\Theta_0 &\sim \text{Dirichlet}(\alpha_0, L_0) \\
\Theta_{+|T} &\sim \text{Dirichlet}(\alpha_+, L_+)
\end{aligned}$$

For each word-type Wi, the number of morphemes in its segmentation S is drawn from truncated geometric distribution which allows maximum m morphemes per word-type:

$$|S| \sim \text{Truncated-Geometric}(\gamma_{|S|}) = \frac{\gamma_{|S|}(1-\gamma_{|S|})^{|S|}}{\sum_{j=1}^{m} \gamma_{|S|}(1-\gamma_{|S|})^j}$$

Once the number of morphemes is sampled, the model randomly picks one morpheme as stem from uniform distribution, the prefixes and suffixes are then determined according to the position of the stem.

Next, the model draws syntactic category T of word-type Wi from categorical distribution:

$$T \sim \mathrm{Cat}(\Theta_T)$$

Afterward, the model generates stem σ_0, prefixes σ_-, and suffixes σ_+ independently:

$$\begin{aligned}
\sigma_0 &\sim \mathrm{Cat}(\Theta_0) \\
\sigma_-|T &\sim \mathrm{Cat}(\Theta_{-|T}) \\
\sigma_+|T &\sim \mathrm{Cat}(\Theta_{+|T})
\end{aligned}$$

Recall equation 2.18 for computing the posterior $P(x_{n+1} = j|\boldsymbol{x}, \boldsymbol{\beta})$ for a new observation x_{n+1} given previous observations $\boldsymbol{x} = x_1, ..., x_n$ drawn from categorical distribution with hyperparameters $\boldsymbol{\beta}$:

$$P(x_{n+1} = j|\boldsymbol{x}, \boldsymbol{\beta}) = \frac{n_j + \beta_j}{n + \sum_{k=1}^{K} \beta_k}$$

Using this formula, the probability of generating tag T, stem σ_0, prefix σ_-, and suffix σ_+ for word-type Wi is computed as the product of the following equations:

$$P(t_i = T|\boldsymbol{T}^{-i}, \alpha_T) = \frac{n_T^{-i} + \alpha_T}{N^{-i} + \alpha_T K} \tag{4.6}$$

$$P(\sigma_0|\boldsymbol{L}^{-i}, \alpha_0) = \frac{n_{\sigma_0}^{-i} + \alpha_0}{N_0^{-i} + \alpha_0|L_0|} \tag{4.7}$$

$$P(\sigma_-|\boldsymbol{L}^{-i}, \alpha_-) = \frac{n_{\sigma_-|T}^{-i} + \alpha_-}{N_{-|T}^{-i} + \alpha_-|L_-|} \tag{4.8}$$

$$P(\sigma_+|\boldsymbol{L}^{-i}, \alpha_+) = \frac{n_{\sigma_+|T}^{-i} + \alpha_+}{N_{+|T}^{-i} + \alpha_+|L_+|} \tag{4.9}$$

where the superscript $-i$ indicates that the relative counts exclude the word type Wi. n_T^{-i} is the number of word-types with tag T, N^{-i} is the number of word-types excluding word-type Wi, $n_{\sigma_0}^{-i}$ is the number of stems σ_0 in the stem lexicon L_0, N_0^{-i} is the total number of stems, $n_{\sigma_-|T}^{-i}$ is the number of prefixes σ_- associated with word-types tagged with tag T, $N^{-i}_{-|T}$ is the number of prefixes in all word-types that has tag T. The notions for suffixes are analogous to the notions for prefixes.

The final sampling equation is then given as:

$$\frac{\gamma_{|S|}(1-\gamma_{|S|})^{|S|}}{\sum_{j=0}^{m}\gamma_{|S|}(1-\gamma_{|S|})^{j}} \times \frac{n_T^{-i}+\alpha_T}{N^{-i}+\alpha_T K} \times \frac{n_{\sigma_0}^{-i}+\alpha_0}{N_0^{-i}+\alpha_0|L_0|} \times \frac{n_{\sigma_-|T}^{-i}+\alpha_-}{N_{-|T}^{-i}+\alpha_-|L_-|} \times \frac{n_{\sigma_+|T}^{-i}+\alpha_+}{N_{+|T}^{-i}+\alpha_+|L_+|} \tag{4.10}$$

4.1.2.3 Token-POS model

Token-POS model plays a role as an unsupervised POS type-based tagger. The model generates tokens $\boldsymbol{w}$ and their POS tags $\boldsymbol{t}$ with probability:

$$P(\boldsymbol{w},\boldsymbol{t}|\mathbf{W},\mathbf{T},\boldsymbol{\theta}) = \prod_{w_i,t_i} P(t_{i-1}|t_i,\theta_{t|t})P(w_i|t_i,\theta_{w|t})$$

Transition probabilities and emission probabilities are specified by a collection of categorical parameters $\boldsymbol{\theta} = \{\theta_{(T,k)}\} \cup \{\theta_{(E,k)}\}$, where $\{\theta_{(T,k)}\}$ is the set of K transition distributions, each over K tags and $\{\theta_{(E,k)}\}$ is the set of K emission distributions, each over the set of word-types.

$$\begin{aligned} \theta_{t|t} &\sim Dirichlet(\alpha_{t|t}, \{1,...,K\}) \\ \theta_{w|t} &\sim Dirichlet(\alpha_{w|t}, \boldsymbol{W_t}) \end{aligned}$$

where $\boldsymbol{W_t}$ is the set of word-types that are generated by tag t.

Using the formula for a general type-based sampler in Liang *et al.* (2010), the sampling equation for this model is given by

$$\frac{\alpha_{w|t}^{(m^i)}}{(M_t^{-i}+\alpha_{w|t}|\boldsymbol{W_t}|)^{(m^i)}} \times \prod_{t=1}^{K}\prod_{t'=1}^{K}\frac{(m_{t'|t}^{-i}+\alpha_{t|t})^{(m^i_{t'|t})}}{(M_t^{-i}+\alpha_{t|t})^{(m^i_{t'|t})}} \tag{4.11}$$

where $\alpha^{(m)} = \alpha(\alpha+1)...(\alpha+m-1)$ is the ascending factorial. M_t^{-i} is the number of tokens having tag t, m^i is the number of token w_i, and $m^i_{t'|t}$ is the number of tokens t-to-t' transitions. Note that all the counts for tokens that belong to word-type Wi are excluded.

The first term is the emission probability and the second term is the transition probability with parameters $\boldsymbol{\theta}$ marginalized out.

4.1.2.4 Token-Seg model

Although Lee *et al.* (2011) demonstrated that Token-Seg model improved greatly the performance of the unsupervised morphological segmentation system for Arabic, the model is only suitable for certain language family. It is designed to capture the morpho-syntatic agreement between adjacent tokens which is often realized by matching the last suffixes. Let $\boldsymbol{s}$ denote a sequence of segmentations, and let s_i be the segmentation of i^{th} word in the data. The probability of drawing $\boldsymbol{s}$ is given by

$$P_{\text{seg}}(\boldsymbol{s}|\mathbf{W}, \mathbf{S}, \mathbf{T}, \mathbf{L}, \boldsymbol{\beta}, \boldsymbol{\alpha}) = \prod_{(s_{i-1}, s_i)} p(s_i|s_{i-1}) \tag{4.12}$$

The model is designed in such a way that it encourages adjacent tokens exhibiting morpho-syntactic agreement by having the same final suffix while it penalizes the case when adjacent tokens have the same ending but different final suffixes. To achieve this goal, the model first computes n, the length of the longest final suffix in pair of segmentations (s_{i-1}, s_i), and sets the last n characters of each word as its *ending*. A simple matching method then serves as a proxy for morpho-syntactic agreement between the two words. Finally, the model defines a probability distribution over pair $(s_i|s_{i-1})$

$$p(s_i|s_{i-1}) = \begin{cases} \beta_1, & \text{if same endings and same final suffix} \\ \beta_2, & \text{if same endings but different final suffixes} \\ \beta_3 & \text{otherwise} \end{cases}$$

where $\beta_1 + \beta_2 + \beta_3 = 1$ and $\beta_1 > \beta_3 > \beta_2$.

The sampling equation for word-type Wi has the form:

$$\beta_1^{m^i_{\beta_1}} \beta_2^{m^i_{\beta_2}} \beta_3^{m^i_{\beta_3}} \tag{4.13}$$

in which, $m^i_{\beta_1}$ is the number of transitions where word-type Wi occurs such that Wi and its neighbor have the same final suffix. $m^i_{\beta_2}$ and $m^i_{\beta_3}$ are read analogously.

4.1.3 Training procedure

The model is trained stage by stage, the next stage adds a new submodel and uses the previous stage for initialization.

4.2 Experimental Setup

4.2.1 Performance metrics

In order to compare with other works, I evaluate the segmentation results using the evaluation scheme in Morpho Challenge (MC for short), and the EMMA method. The scripts for evaluating are obtained at http://research.ics.aalto.fi/events/morphochallenge/.

4.2.2 Data

I evaluate the model on 4 languages: English, Turkish, Tamil, and Telugu. I collect word lists[1] and gold standard segmentations for English and Turkish from the series of the Morpho Challenge[2]. For each word list, I randomly select 70,000 word types as training data.

For Tamil and Telugu, I use the same data as Ramasamy *et al.* (2012). They randomly selected articles from monolingual section of Tamil and Telugu in EMILLE corpus Xiao *et al.* (2004) and transliterated the them into the Latin script. For each language, they created a word list from real sentences in EMILLE corpus and manually annotated every word in the list to obtain gold standard segmentations.

Table 4.1: Gold standard segmentations statistics

Language	#word-types	#morphemes	#unique morphemes
English	2,545	5,884	2,191
Turkish	2,867	20,227	1,760
Tamil	1,080	2,641	848
Telugu	997	1,732	1,266

4.2.3 Software

I implemented the model[3] described above in Julia[4]. I also obtained implementations of various sytems participated in Morpho Challenge for the comparison,

[1] Because the model is fully unsupervised, I only take the word lists which contain words and their frequency as my inputs.

[2] http://research.ics.aalto.fi/events/morphochallenge/

[3] https://github.com/ketranm/morpho-segmentation

[4] http://julialang.org/

including Morfessor Categories-MAP, Morfessor Baseline[1] and MORSEL[2]. These systems were ranked among the best systems in Morpho Challenge.

4.2.4 Submodels and prameters setting

As mentioned in the previous section, Token-Seg model was designed for Arabic, the language that morpho-syntactic agreement can be realized using matching suffixes. This observation has not been seen in 4 languages to be evaluated, so I exclude Token-Seg model.

In my preliminary experiments, adding Token-POS model does not improve F1-score. Lee *et al.* (2011) also reported similar result in their experiment for Arabic using paired t-test. Thus, I only use lexicon model and segmentation model.

In all the experiments, I set $\gamma_l = \frac{1}{1.1}$ (for the length of morphemes), $\gamma_{|S|} = \frac{1}{2}$ (for the number of morphemes of each word), $\gamma_- = \gamma_+ = \frac{1}{1.1}$ (for the size of the prefix and the suffix lexicons) to favor small sets of affixes, and $\gamma_0 = \frac{1}{10{,}000}$ (for the size of s the stem lexicon). To prefer sparse distributions in segmentation model, I set concentration parameters $\alpha_T = \alpha_- = \alpha_+ = \alpha_0 = 0.1$. Number of POS tags is set to 5.

4.2.5 Baselines

I run experiments with Mofessor Cat-MAP, Morfessor Baseline, and MORSEL on the same dataset for each language and use the results as the baselines.

4.2.6 Unrealistic setting

The "unrealistic experiments" is set up to evaluate the robustness of the model. Under this setting, I train the model on gold standard datasets (only word types in gold standard, the model does not access segmentation information). The training data in this case is much smaller. Because the computation is cheaper for small training data, I will apply maximum marginal decoding (MM) technique by drawing 15 independent Gibbs samplers.

4.3 Results

Table 4.2 and table 4.3 show the results of evaluation using MC method and EMMA method respectively.

[1] http://www.cis.hut.fi/projects/morpho/
[2] https://github.com/ConstantineLignos/MORSEL

Table 4.2: Results of evaluation with MC method

Language	Model	Precision	Recall	F1
English	MORSEL	57.64%	53.43%	55.45%
	Morfessor Baseline	55.10%	57.94%	**56.48%**
	Morfessor-CatMAP	31.88%	33.26%	32.55%
	Lexicon	60.36%	38.26%	46.83%
	+Segmentation	59.54%	43.74%	50.43%
Turkish	MORSEL	72.95%	17.72%	28.51%
	Morfessor Baseline	80.25%	16.32%	27.12%
	Morfessor-CatMAP	76.31%	24.66%	**37.27%**
	Lexicon	70.84%	18.74%	29.64%
	+Segmentation	72.31%	18.40%	29.34
Tamil	MORSEL	54.14%	18.52%	27.60%
	Morfessor Baseline	60.43%	31.74%	41.62%
	Morfessor-CatMAP	51.15%	45.43%	**48.12%**
	Lexicon	69.51%	22.56%	34.07%
	+Segmentation	67.87%	23.68%	35.11%
Telugu	MORSEL	36.31%	2.58%	4.81%
	Morfessor Baseline	24.89%	54.32%	**34.14%**
	Morfessor-CatMAP	13.66%	53.96%	21.80%
	Lexicon	28.36%	30.16%	29.23%
	+Segmentation	29.49%	34.29%	31.71%

Table 4.3: Results of evaluation with EMMA method

Language	Model	Precision	Recall	F1
English	MORSEL	84.15%	72.72%	78.02%
	Morfessor Baseline	79.91%	78.56%	**79.23%**
	Morfessor-CatMAP	85.52%	69.09%	76.27%
	Lexicon	84.08%	72.11%	77.64%
	+Segmentation	83.75%	73.26%	78.15%
Turkish	MORSEL	85.98%	29.60%	44.04%
	Morfessor Baseline	87.30%	30.31%	45.00%
	Morfessor-CatMAP	84.90%	35.67%	**50.24%**
	Lexicon	82.26%	33.53%	47.64%
	+Segmentation	82.43%	33.90%	48.04%
Tamil	MORSEL	84.95%	63.40%	72.61%
	Morfessor Baseline	85.00%	67.25%	75.09%
	Morfessor-CatMAP	80.17%	73.59%	**76.74%**
	Lexicon	92.46%	63.76%	75.47%
	+Segmentation	92.60%	64.35%	75.93%
Telugu	MORSEL	98.14%	80.79%	**88.62%**
	Morfessor Baseline	70.89%	92.47%	80.25%
	Morfessor-CatMAP	56.30%	93.23%	70.20%
	Lexicon	78.13%	88.70%	83.08%
	+Segmentation	77.87%	88.44%	82.82%

The F1 score evaluated with EMMA method for Telugu gives highest value for MORSEL system while MC method gives lowest value. Why does the contradiction appear? Table 4.4 shows that in gold standard datasets, the number of unique morphemes is often smaller than the number of word types for all the languages except for Telugu. It implies that not many morphemes in Telugu gold standard dataset have been reused.

Table 4.4: Segmentations statistics of gold standard datasets

Language	Model	#types	#morph	#unique morph
English	MORSEL	2,545	5,620	2,103
	Morfessor Baseline	2,545	5,994	2,118
	Morfessor-CAT	2,545	5,680	2,593
	Lexicon	2,545	4,029	2,263
	+Segmentation	2,545	4,153	2,256
	Gold standard	2,545	5,884	2,191
Turkish	MORSEL	2,867	6,587	2,556
	Morfessor Baseline	2,867	7,017	2,324
	Morfessor-CAT	2,867	8,124	2,366
	Lexicon	2,867	7,802	2,458
	+Segmentation	2,867	7,913	2,418
	Gold standard	2,867	20,227	1,760
Tamil	MORSEL	1,080	1,840	989
	Morfessor Baseline	1,080	2,182	1,043
	Morfessor-CAT	1,080	2,615	924
	Lexicon	1,080	1,707	969
	+Segmentation	1,080	1,735	971
	Gold standard	1,080	2,641	848
Telugu	MORSEL	997	1,108	1,033
	Morfessor Baseline	997	2,390	1,268
	Morfessor-CAT	997	3,086	1,186
	Lexicon	997	2,084	1,315
	+Segmentation	997	2,080	1,309
	Gold standard	997	1,732	1,266

4.3.1 Unrealistic setting

Table 4.5 shows the results of the expriments under unrealistic setting. MORSEL performs worst[1] when it is trained on small dataset since there are not many minimal word-pairs that could be found in the training data. Lexicon model and + Segmentation model give higher F1 scores for English and Tamil. Size of training data could affect the performance of the system. Training on large data, the system might induce spurious affixes.

MM technique helps improving F1 scores in general.

[1]This is because of MORSEL does not segment words in gold standard while every word in standard have approximaly 3 morphemes (Telugu) and each word can have more than one analysis (Turkish). This make the MC scheme is not usable.

Table 4.5: Results of evaluation with MC method in unrealistic setting. Precision, Recall and F1 are reported as the mean scores of 15 independent Gibbs samples. The sample standard deviations are shown in brackets. Lexicon MM and + Segmentation MM are the results after applying maximum marginal decoding technique. ∞ means that it is not possible to evaluate using MC scripts.

Language	Model	Precision	Recall	F1
English	MORSEL	100.00%	2.25%	4.40%
	Morfessor Baseline	65.81%	48.32%	55.73%
	Morfessor-CatMAP	71.93%	46.58%	56.55%
	Lexicon	61.46%	53.92%	57.40% (1.1)
	+Segmentation	60.51%	54.73%	57.43% (1.2)
	Lexicon MM	60.47%	55.40%	57.82%
	+Segmentation MM	62.15%	55.98%	**58.90%**
Turkish	MORSEL	∞	∞	∞
	Morfessor Baseline	77.29%	18.32%	29.61%
	Morfessor-CatMAP	82.63%	18.12%	**29.72%**
	Lexicon	80.83%	16.85%	27.88% (0.6)
	+Segmentation	81.03%	17.45%	28.71% (0.9)
	Lexicon MM	86.09%	16.16%	27.22%
	+Segmentation MM	86.48%	17.14%	28.61%
Tamil	MORSEL	81.82%	1.17%	2.31%
	Morfessor Baseline	52.54%	38.37%	**44.35%**
	Morfessor-CatMAP	53.55%	37.65%	44.21%
	Lexicon	53.43%	34.56%	41.95% (1.3)
	+Segmentation	52.76%	34.33%	41.57% (0.9)
	Lexicon MM	57.74%	33.63%	42.51%
	+Segmentation MM	57.98%	32.67%	41.80%
Telugu	MORSEL	∞	∞	∞
	Morfessor Baseline	38.72%	37.06%	37.87%
	Morfessor-CatMAP	42.29%	37.06%	**39.50%**
	Lexicon	17.59%	52.15%	26.23% (1.5)
	+Segmentation	18.01%	55.60%	27.15% (1.8)
	Lexicon MM	15.96%	57.58%	24.99%
	+Segmentation MM	17.06%	56.88%	26.24%

Table 4.6: Evaluation using EMMA method

Language	Model	Precision	Recall	F1
English	MORSEL	99.94%	46.07%	63.07%
	Morfessor Baseline	81.92%	70.69%	75.89%
	Morfessor-CatMAP	87.01%	71.26%	78.35%
	Lexicon	82.16%	76.38%	79.16% (0.38)
	+Segmentation	81.03%	76.83%	78.87% (0.28)
	Lexicon MM	84.35%	77.36%	**80.70%**
	+Segmentation MM	83.00%	78.00%	80.32%
Turkish	MORSEL	100%	16.67%	28.59%
	Morfessor Baseline	82.58%	31.54%	45.65%
	Morfessor-CatMAP	89.06%	32.07%	47.16%
	Lexicon	88.07%	31.73%	46.53% (0.43)
	+Segmentation	87.71%	32.20%	47.11% (0.43)
	Lexicon MM	90.99%	31.87%	47.20%
	+Segmentation MM	90.17%	32.92%	**48.23%**
Tamil	MORSEL	99.54%	47.27%	64.10%
	Morfessor Baseline	76.79%	74.10%	75.42%
	Morfessor-CatMAP	78.41%	73.84%	76.06%
	Lexicon	78.31%	72.84%	75.48% (0.37)
	+Segmentation	77.30%	72.86%	75.01% (0.38)
	Lexicon MM	79.93%	72.95%	**76.28%**
	+Segmentation MM	78.99%	72.73%	75.73%
Telugu	MORSEL	100%	78.86%	88.18%
	Morfessor Baseline	89.79%	88.90%	89.34%
	Morfessor-CatMAP	91.01%	88.76%	**89.91%**
	Lexicon	62.55%	91.95%	74.45% (0.74)
	+Segmentation	60.81%	92.28%	73.31% (0.49)
	Lexicon MM	64.37%	92.56%	75.93%
	+Segmentation MM	62.26%	92.87%	74.55%

Chapter 5

Word Representation improves Unsupervised Morphological Segmentation

Traditional NLP approaches have relied on set of human-designed features extracted from training data. The choice of features is often based on linguistic intuition and empirical experiment depending on a specific task. Recently, researchers have taken a new approach which attempts to automatically learn good features from input data. This approach is referred as *representation learning* or *feature learning*. It has been shown that these learned features greatly improve the performance of existing NLP systems Socher *et al.* (2011a,b, 2012); Turian *et al.* (2010) while reducing numerous effort for task-specific engineering features Collobert & Weston (2008); Collobert *et al.* (2011).

Inspired by previous successful approaches which yield substantial gains in performance across a wide range of NLP tasks by training existing supervised Turian *et al.* (2010) or semi-supervised Koo *et al.* (2008) NLP systems using unsupervised word representations as extra word features, I propose a simple generative model for *unsupervised* morphological segmentation that could make use of word representations. The research question here is: Do word representations help in unsupervised context?

5.1 Distributed representations

There are several approaches to represent words in a more useful and meaningful way. Word representations induced by those approaches, however, can be classified into three main categories: distributional representations Blei *et al.* (2003); Dumais *et al.* (1988); Hofmann (1999); Landauer *et al.* (1998), cluster-based representations, and distributed representations. Since previous research has successfully applied distributed representations for a variety of NLP tasks, I will focus on *distributed representations*.

Distributed word representations are typically induced by using neural language models. The language models learn to map words into real-valued feature vectors, which are dense and low dimensional. Words transformed into feature vectors are called word *embeddings*. Each dimension of the embedding represents a latent feature of the word. In the following, I briefly summarize the language model presented in Collobert & Weston (2008) using the notations in Turian *et al.* (2010).

Each word w_i in a finite dictionary $\mathcal{D}$ is embedded into a d dimensional space using a lookup table e:

The model reads input sentence $x = (x_1, ..., x_n)$ and transforms it into a series of vectors $e(w_1) \oplus ... \oplus e(w_n)$ by using the lookup table e, here $\oplus$ denotes concatenation operator. The next step is to generate a negative example by corrupting the last word w_n. This sprit is similar to contrastive estimation proposed by Smith & Eisner (2005). The language model should learn to assign high score for true example and low score to negative example. Let $\tilde{x} = (x_1, ..., \tilde{w}_n)$ denote the negative example, where $\tilde{w}_n$ is randomly selected from the dictionary $\mathcal{D}$. For convenience, denote $e(x) = e(w_1) \oplus ... \oplus e(w_n)$. Passing $e(x)$ through a single hidden layer neural network, the model returns a score $s(x)$. The loss function needed to be minimized is $L(x) = max(0, 1 - s(x) + s(\tilde{x}))$. The distributed representation is learnt as a result of doing gradient descent simultaneously over the neural network parameters and the embedding lookup table.

5.2 The Model

A distributed representation could capture many features for a word such as syntactic features (such as its distribution over POS tags), semantic features (is it the name of a job? etc), morphological features (which affix it could have?), and so forth Bengio (2009). For unsupervised morphological segmentation task, I employ morphological features captured in distributed word representation.

In the embedding space, words with similar affixes are closer together (Figure 5.1). Therefore, I group words into clusters and force words in the same cluster to select similar affixes.

The model contains three sub-models: Lexicon model, Segmentation model, and Cluster-Segmentation model. The Lexicon model and the Segmentation model are reused from chapter 4. The Cluster-Segmentation model is designed in a similar spirit to the Token-Seg model in the previous chapter.

Let $\mathcal{C} = C_1, ..., C_M$ denote the set of word clusters. Each word type W_i either

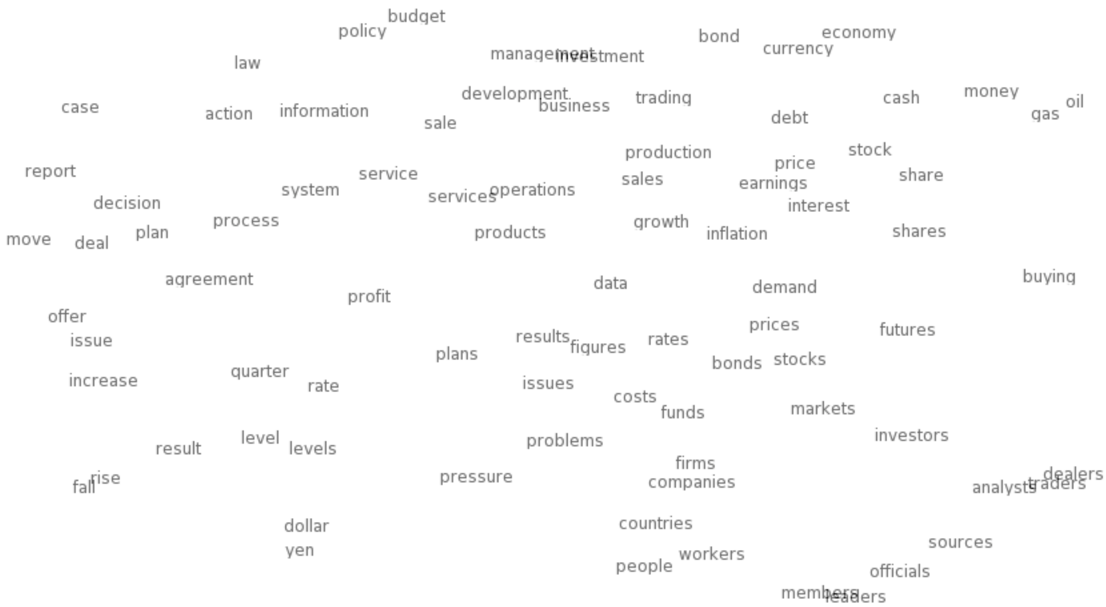

Figure 5.1: A visualization of word embeddings

belongs to a cluster $C_j \in \mathcal{C}$, or it belongs to none. I will explain where the clusters come from shortly.

Based on the linguistic intuition that the final suffix is often the strongest indicator for the syntactic category of the word, I place a Categorical distribution on the final suffixes of all the words in each cluster. Let L_{C-} denote the set of the final suffixes for cluster C. The final suffix σ_{-}^{i} (if a word does not have any suffix, its final suffix is `NONE`) of a word type $W_i \in C$ is generated from Categorical distribution:

$$\sigma_{-}^{i} \sim \mathrm{Cat}(\Theta_C) \tag{5.1}$$

where Θ_C is drawn from Dirichlet prior.

$$\Theta_C \sim \mathrm{Dirichlet}(\alpha_C, L_{C-}) \tag{5.2}$$

and the hyperparameter α_C of the Dirichlet prior is chosen to be less than 1 to encourage sparsity.

Table 5.1 gives an example of words and clusters. Words in the same cluster not only tend to have similar syntactic categories but also share similar semantic categories.

Table 5.1: Sample words and clusters extracted from data

Cluster sample				
654	716	984	273	1018
impressionistic	portraitist	interfering	slovak	melody
minimalistic	parliamentarian	questioning	slovakian	playback
improvised	polemicist	reconciling	slovenian	sounds
idiosyncratic	propagandist	sympathizing	slovene	stereo
innovative	revivalist	tinkering	valencian	sync
inventive	satanist	collaborating	macedonian	tempo
multifaceted	supporter	brainwashing	luxembourgish	voice
naturalistic	thinker	clashing	pomeranian	tone
ephemeral	woodcarver	adventuring	portuguese	reverb
distinctive	chronicler	deliberating	serbian	swing
anachronistic	centenarian	interfering	czechoslovak	drum
colourful	grammarian	conspiring	croatian	crescendo
idealised	theologian	assisting	corsican	instrumentation
idealized	bostonian	allying	bulgarian	acoustic
illustrative	landowner	eavesdropping	bosnian	distortion
imaginative	nobleman	enlisting	belarusian	ambient
incisive	frenchman	pleading	kyrgyz	arrangement

Where do the word clusters come from? Having word embeddings in N dimensional space of real numbers, one can use a clustering algorithm such as K-mean to obtain word clusters.

Because `NONE` is counted as the final suffix, it might be the case that there are many `NONE`s in a cluster (for example, cluster 1018 showed in Table 5.1.) In this case, the word "sounds" in cluster 1018 might not be segmented because the probability to generate `NONE` is much higher than the probability to generate `s` as the final suffix within cluster 1018.

As a treatment for this problem, I define a probability distribution $p(s_i|C)$ over the segmentation s_i given its cluster C as follows:

$$p(s_i|C) = \begin{cases} \beta_1, & \text{if the final suffix is } \texttt{NONE} \\ \beta_2, & \text{if the final suffix is unique in } C \\ \beta_3 & \text{otherwise} \end{cases}$$

where $\beta_1 + \beta_2 + \beta_3 = 1$ and $\beta_1 \leq \beta_2 < \beta_3$.

By setting the highest value to β_3, I encourage the words within the same cluster to exhibit the syntactic and semantic agreements.

5.2.1 Sampling equation

The sampling equation for Cluster-Segmentation model is

$$P(\sigma_{-}|C, \alpha_C, \boldsymbol{\beta}) = \frac{n^{-i}_{\sigma_{-}|C} + \alpha_C}{N^{-i}_C + \alpha_C |L^{-i}_{C-}|} \times {\beta_1}^{I_1(C)} {\beta_2}^{I_2(C)} {\beta_3}^{I_1(C)} \tag{5.3}$$

here N^{-i}_C is the size of cluster C, $n^{-i}_{\alpha-|C}$ is the number of the final suffix $\sigma-$ found in C, L^{-i}_{C-} is the set of final suffixes (excluding the counts contributed by word type Wi.). $I_j(C)$, $j \in \{1, 2, 3\}$ is the indicator functions (i.e if the final suffix = `NONE`) whose values $\in \{0, 1\}$.

5.3 Experimental Setup

5.3.1 Data

I use the same English word list as in chapter 4, I obtain word embeddings from Socher *et al.* (2011b). They pre-trained word embeddings using Collobert-Weston neural language model Collobert & Weston (2008).

To obtain word clusters, I use K-mean clustering algorithm with number of clusters $K = 1500$.

5.3.2 Parameters setting

I set hyperparameter $\alpha_C = 0.1$ for all clusters, $\beta_1 = 0.2$, $\beta_2 = 0.2$ and $\beta_3 = 0.6$. Rest of the parameters are set the same values as in chapter 4.

5.4 Result

Table 5.2 shows that adding Cluster model improved F1 score by 4.56%. Running the bootstrap for 10^5 iterations, the *confidence* (1-p-value) is equal to 1 in both paired tests for (Lexicon, +Segmentation) and (+Segmentation, +Cluster).

Table 5.2: Evaluation using MC method

Model	Precision	Recall	F1
Lexicon	60.36%	38.26%	46.83%
+Segmentation	59.54%	43.74%	50.43%
+Cluster	61.94%	49.44%	**54.99%**

Table 5.3: Evaluation using EMMA method

Model	Precision	Recall	F1
Lexicon	84.08%	72.11%	77.64%
+Segmentation	83.75%	73.26%	78.15%
+Cluster	84.18%	75.13%	**79.40%**

5.5 Discussion

I have shown that using word representations as extra features could improve the unsupervised system. However, there are some limitations in this work. Firstly, the experiment is only for English, we need to evaluate the model on more languages to see if the model behaves the same. Secondly, the quality of the clusters might affect the performance of the model. One drawback of K-mean is that number of clusters is required to specify beforehand. It would be better if we let the data decides the number of clusters by itself. For example, we can use Distance Dependent Chinese Restaurant Process Blei & Frazier (2009) for clustering instead of K-mean.

Chapter 6

Conclusions

In this thesis, I have evaluated various unsupervised morphological segmentation systems for 4 languages: English, Turkish, Tamil, and Telugu. I also have shown that maximum marginal decoding could help reducing variance and noise in the output of Gibbs samples.

In chapter 5, I have presented the generative model that uses word representation as extra features. The model improved dramatically F1 score for English.

6.1 Limitations

In chapter 5, I have not used maximum marginal decoding technique[1]. It would be interesting to see by how large the MM technique could improve F1 score. Also, the generative model in chapter 5 needs to be tested on other languages.

6.2 Future Work

The relationship between size of training data and the peformance of unsupervised systems is interesting as well. In which case the preformance of the system is better: training on a small selective dataset or training on a massive dataset? If it is the former case, how to select such a dataset?

[1]Due to the lack of computational resources.

Training data examples

English	Turkish	Tamil	Telugu
inital	elimizi	mwepiyaj	dhOraNilO
panics	trm	awTarangkaTTil	prOgraaMnibaTTi
namesakes	ulu	munmozivOm	bhootaM
familia	fermuarlI	kAraNaTTaikkURi	moduLLaku
unnaturally	filozof	wTETiyum	maarataaDaemOyidi
downfall	edilmelerini	variyai	naakishTaMlaeka
newsgroup	baktI	alangkarikkappattu	akkaraku
co-ordinated	klasOre	viLakkukaL	aadaarina
christabel	yapIlmamalIdIr	layancu	aeraati
goodwin	SUkran	ezuwTaTum	nirasanapatraM
paducah	pars	cattamanRaTTai	vidyudutpatti
upstream	gOrUSmelerinin	katciTTalaivarkaL	shel
castrated	CIkacaGInI	TIvira	aalOchiMchukOTaanikee
nisar	Cikmak	pArAkotu	aedaitae

Gold standard and model's output examples

Table 1: English

Word type	**Gold standard segmentations**	**Proposed segmentations**
stabilized	stable_A ize_s +PAST	stabiliz + ed
drumheads	drum_N head_N +PL	drumheads
resonant	resonate_V ant_s	resonant
punishment	punish_V ment_s	punish + ment
dragged	drag_V +PAST	dragg + ed
abounded	abound_V +PAST	abound + ed
commissioning	commit_V ion_s +PCP1	commission + ing
trying	try_V +PCP1, trying_V	trying
cabal	cabal_N	cabal
pensionable	pension_N off_B able_s	pension + able
the	the_B, the_D	the
corroborated	corroborate_V +PAST	corroborat + ed
suffuse	suffuse_V	suffuse
pottages	pot_N age_s +PL	pottages
townsman	town_N s_s man_N	townsm + an
sip	sip_V	sip
ford	ford_N	ford
golf-club	golf_N club_N	golf-club
ancestors	ancestor_N +PL	ancestor + s
tripartite	tri_p part_N ite_s	tripartite

Word type	Gold standard segmentations	Proposed segmentations
mankenlerden	manken +PL +ABL	manken + ler + den
sabotaj	sabotaj	sab + ot + aj
kazandIrIrlar	kazan +CAU_dir +TNS_ir +PER3P	kazan + dIr + Ir + la + r
gUClendirirken	gUC +la_DER_RFL +CAU_dir +TNS_ir iken_e	gUClendirir + ken
sOnme	sOn +NEG_ma, sOn +NOUN_ma	sOnme
parolalarI	parola +PL +ACC, parola +PL +POS3, parola +POS3S	parola + lar + I
yUrUyUSleri	yUrU yis +PL +ACC, yUrU yis +PL +POS3, yUrU yis +POS3S	yUrUyUS + ler + i
dinlemek	din +la_DER mak, dinle mak	dinle + mek
personelimiz	personel +POS1P, personel +POS1S +PER1P	personelimiz
biriktiGi	birik +ADJ_dig +ACC, birik +ADJ_dig +POS3	bir + ik + ti + Gi
literatUr	literatUr	literatUr
bilmeksizin	bil maksizin	bilmek + siz + in
atlayayIm	at +la_DER +OPT +PER1S, atla +OPT +PER1S	atla + ya + yI + m
savurganlIGI	savurgan +DER_lHg +ACC, savurgan +DER_lHg +POS3	savurg + an + lIGI
aptallIk	aptal +DER_lHk	aptal + lI + k
hayata	hayat +DAT	hayat + a
haClarIn	haC +PL +GEN, haC +PL +POS2S	haC + la + rI + n

Table 2: Turkish

Table 3: Tamil

Word type	**Gold standard segmentations**	**Proposed segmentations**
ewTa	ewTa	ewTa
ikkuzu	ik + kuzu	ikkuzu
mAwila	mAwila	mAwila
mUlam	mUlam	mUlam
iru	iru	iru
anniya	anniya	anniya
vazakkamAka	vazakkam + Aka	vazakkam + Aka
uriTTAkka	uriTTAkk + a	uriTTAkka
ceyalpatAmal	ceyalpat + Amal	ceyalpat + Amal
muzuvaTilum	muzuvaT + il + um	muzuva + Til + um
pOnapiRaku	pOna + piRaku	pOnapiRaku
puriwTukoLLa	puri + wT + u + koLL + a	puriwTu + koLLa
paTivu	paTivu	paTivu
kanavai	kanav + ai	kanav + ai
aRiyamutiyum	aRi + y + a + muti + y + um	aRiyamutiyum
irukka	iru + kk + a	irukka
pOStarkaL	pOStar + kaL	pOStar + kaL
kAlaTTin	kAla + TT + in	kAlaTT + in
waTikaLil	waTi + kaL + il	waTikaL + il

Table 4: Telugu

Word type	**Gold standard segmentations**	**Proposed segmentations**
cheema	cheema	cheema
yika	yika	yika
chaetinuMDi	chaeti + nuMDi	chaeti + nuMDi
udyOgi	udyOgi	udyOg + i
tiyyani	tiyyani	tiyya + ni
railumeeda	railu + meeda	railu + meed + a
maaTlaaDadalistae	maaTlaaDa + dalistae	maaTlaaD + adali + stae
vechchagaa	vechcha + gaa	vechcha + gaa
graama	graama	graama
nuMchee	nuMchee	nuMchee
paTTamu	paTTamu	paTT + amu
koorchuni	koorchuni	koorchu + ni
yennaaLlani	yennaaLl + ani	yennaaL + lani
koddinimushaallO	koddi + nimushaal + lO	koddini + mushaa + llO
saMghamunaku	saMghamu + na + ku	saMgha + mu + naku
bayaTivaaLlatO	bayaTi + vaaLla + tO	bayaTi + vaaL + latO
taedeela	taedee + la	taedeel + a
choosi	choosi	choosi
kOrika	kOrika	kOrika
dooramunuMchi	dooramu + nuMchi	dooramu + nuMchi

References

BENGIO, Y. (2009). Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, **2**, 1–127, also published as a book. Now Publishers, 2009. 38

BERG-KIRKPATRICK, T., BURKETT, D. & KLEIN, D. (2012). An empirical investigation of statistical significance in nlp. In *Proceedings of EMNLP*, Jeju, South Korea. 20, 21

BISANI, M. & NEY, H. (2004). Bootstrap estimates for confidence intervals in ASR performance evaluation. 20

BLEI, D.M. & FRAZIER, P.I. (2009). Distance Dependent Chinese Restaurant Processes. *Engineering*, **12**, 1–26. 42

BLEI, D.M., NG, A.Y. & JORDAN, M.I. (2003). Latent Dirichlet Allocation. *Journal of Machine Learning Research*, **3**, 993–1022. 37

COHEN, S.B., DAS, D. & SMITH, N.A. (2011). Unsupervised Structure Prediction with Non-Parallel Multilingual Guidance. *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, 50–61. 9

COLLOBERT, R. & WESTON, J. (2008). A Unified Architecture for Natural Language Processing : Deep Neural Networks with Multitask Learning. *Architecture*, 160–167. 37, 38, 41

COLLOBERT, R., WESTON, J., BOTTOU, L., KARLEN, M., KAVUKCUOGLU, K. & KUKSA, P. (2011). Natural Language Processing (almost) from Scratch. *Journal of Machine Learning Research*, **12**, 2493–2537. 37

COWAN, B. & COLLINS, M. (2005). Morphology and reranking for the statistical parsing of spanish. In *Proceedings of Human Language Technology Conference* *and Conference on Empirical Methods in Natural Language Processing*, 795–802, Association for Computational Linguistics, Vancouver, British Columbia, Canada. 3

Creutz, M. (2003). Unsupervised segmentation of words using prior distributions of morph length and frequency. *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics ACL 03*, 280–287. 5

Creutz, M. (2006). *Induction of the Morphology of Natural Language: Unsupervised Morpheme Segmentation with Application to Automatic Speech Recognition*. Ph.D. thesis, Helsinki University of Technology. 5

Creutz, M. & Lagus, K. (2002). Unsupervised Discovery of Morphemes. *Proceedings of the ACL02 workshop on Morphological and phonological learning*, **6**, 10. 5

Creutz, M. & Lagus, K. (2005a). Inducing the Morphological Lexicon of a Natural Language from Unannotated Text. *Baseline*, **1**, 106–113. 5

Creutz, M. & Lagus, K. (2005b). Unsupervised Models for Morpheme Segmentation and Morphology Learning. *ACM Transactions on Speech and Language Processing*, **4**, 1–34. 17

Das, D. & Petrov, S. (2011). Unsupervised part-of-speech tagging with bilingual graph-based projections. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 600–609, Association for Computational Linguistics, Portland, Oregon, USA. 9

Dreyer, M. & Eisner, J. (2011). Discovering morphological paradigms from plain text using a Dirichlet process mixture model. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 616–627, Edinburgh, supplementary material (9 pages) also available. 2

Dumais, S.T., Furnas, G.W., Landauer, T.K., Deerwester, S. & Harshman, R. (1988). Using latent semantic analysis to improve access to textual information. *Proceedings of the SIGCHI conference on Human factors in computing systems CHI 88*, 281–285. 37

Efron, B. & Tibshirani, R.J. (1993). *An Introduction to the Bootstrap*, vol. 57 of *Monographs on Statistics and Applied Probability*. Chapman & Hall. 20

Goldwater, S., Griffiths, T.L. & Johnson, M. (2006). Interpolating between types and tokens by estimating power-law generators. *Imagine*, **18**, 459. 8

GOLDWATER, S.J. (2007). *Nonparametric Bayesian Models of Lexical Acquisition*. Ph.D. thesis, Brown University. 12

HOFMANN, T. (1999). Probabilistic latent semantic indexing. *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval SIGIR 99*, **pages**, 50–57. 37

KIM, Y.B., GRAÇA, J.A. & SNYDER, B. (2011). Universal Morphological Analysis using Structured Nearest Neighbor Prediction. *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, 322–332. 9, 10

KOEHN, P. (2004). Statistical Significance Tests for Machine Translation Evaluation. In D. Lin & D. Wu, eds., *Proceedings of EMNLP*, vol. 4, 388–395, EMNLP, Association for Computational Linguistics. 20

KOEHN, P. & KNIGHT, K. (????). Empirical Methods for Compound Splitting. 7

KOO, T., CARRERAS, X. & COLLINS, M. (2008). Simple semi-supervised dependency parsing. In *Proceedings of ACL-08: HLT*, 595–603, Association for Computational Linguistics, Columbus, Ohio. 37

LANDAUER, T.K., FOLTZ, P.W. & LAHAM, D. (1998). An introduction to latent semantic analysis. *Discourse Processes*, **25**, 259–284. 37

LEE, Y.K., HAGHIGHI, A. & BARZILAY, R. (2011). Modeling Syntactic Context Improves Morphological Segmentation. *Computational Linguistics*, 1–9. v, vi, 4, 8, 16, 22, 25, 28, 30

LIANG, P., JORDAN, M.I. & KLEIN, D. (2010). Type-based MCMC. In *North American Association for Computational Linguistics (NAACL)*. 27

LIGNOS, C. (2010). Learning from Unseen Data. In M. Kurimo, S. Virpioja & V.T. Turunen, eds., *Proceedings of the Morpho Challenge 2010 Workshop*, 35–38, Aalto University School of Science and Technology, Helsinki, Finland. 6, 7

MCDONALD, R., PETROV, S. & HALL, K. (2011). Multi-source transfer of delexicalized dependency parsers. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, 62–72, Association for Computational Linguistics, Edinburgh, Scotland, UK. 9

Naradowsky, J. & Toutanova, K. (2011). Unsupervised Bilingual Morpheme Segmentation and Alignment with Context-rich Hidden Semi-Markov Models. *Computational Linguistics*, 895–904. 9

Och, F.J. (2003). Minimum error rate training in statistical machine translation. *Machine Translation*, **1001**, 160–167. 20

Pitman, J. & Yor, M. (1997). The two-parameter poisson-dirichlet distribution derived from a stable subordinator. *The Annals of Probability*, **25**, 855–900. 8

Poon, H., Cherry, C. & Toutanova, K. (2009). Unsupervised morphological segmentation with log-linear models. *Proceedings of Human Language Technologies The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics on NAACL 09*, 209. 3, 7, 8

Ramasamy, L., Žabokrtský, Z. & Vajjala, S. (2012). The study of effect of length in morphological segmentation of agglutinative languages. In *Proceedings of the First Workshop on Multilingual Modeling*, 18–24, Association for Computational Linguistics, Jeju, Republic of Korea. 29

Rissanen, J. (1989). *Stochastic Complexity in Statistical Inquiry*, vol. 15 of *Series in Computer Science*. World Scientific, 2nd edn. 5

Smith, N.A. & Eisner, J. (2005). Contrastive estimation: Training Log-Linear Models on Unlabeled Data. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics ACL 05*, 354–362, Association for Computational Linguistics. 3, 8, 38

Snyder, B. & Barzilay, R. (2008). Unsupervised multilingual learning for morphological segmentation. *Proceedings of the ACLHLT*, 737–745. 9

Snyder, B. & Barzilay, R. (2010). Climbing the Tower of Babel : Unsupervised Multilingual Learning. In J. Fürnkranz & T. Joachims, eds., *Proceedings of the 27th International Conference on Machine Learning ICML10*, 29–36, Omnipress. 9

Snyder, B., Naseem, T., Eisenstein, J. & Barzilay, R. (2008). Unsupervised multilingual learning for POS tagging. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 1041–1050, Association for Computational Linguistics. 9

Snyder, B., Naseem, T. & Barzilay, R. (2009). Unsupervised Multilingual Grammar Induction. *English*, **1**, 73–81. 9

Socher, R., Lin, C.C.Y., Ng, A.Y. & Manning, C.D. (2011a). Parsing Natural Scenes and Natural Language with Recursive Neural Networks. In *Proceedings of the 26th International Conference on Machine Learning (ICML)*. 37

Socher, R., Pennington, J., Huang, E.H., Ng, A.Y. & Manning, C.D. (2011b). Semi-Supervised Recursive Autoencoders for Predicting Sentiment Distributions. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 37, 41

Socher, R., Huval, B., Manning, C.D. & Ng, A.Y. (2012). Semantic Compositionality Through Recursive Matrix-Vector Spaces. In *Proceedings of the 2012 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 37

Spiegler, S. & Monson, C. (2010). EMMA : A Novel Evaluation Metric for Morphological Analysis. *Computational Linguistics*, 1029–1037. 18

Stallard, D., Devlin, J., Kayser, M., Lee, Y.K. & Barzilay, R. (2012). Unsupervised morphology rivals supervised morphology for arabic mt. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 322–327, Association for Computational Linguistics, Jeju Island, Korea. 16

Täckström, O., McDonald, R. & Uszkoreit, J. (2012). Cross-lingual word clusters for direct transfer of linguistic structure. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 477–487, Association for Computational Linguistics, Montréal, Canada. 9

Tenenbaum, J.B., Kemp, C., Griffiths, T.L. & Goodman, N.D. (2011). How to grow a mind: statistics, structure, and abstraction. *Science*, **331**, 1279–1285. 3

Toutanova, K., Suzuki, H. & Ruopp, A. (2008). Applying Morphology Generation Models to Machine Translation. In *Proceedings of ACL08 HLT*, June, 514–522, Association for Computational Linguistics. 3

Turian, J., Ratinov, L.A. & Bengio, Y. (2010). Word Representations: A Simple and General Method for Semi-Supervised Learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, July, 384–394, Association for Computational Linguistics, Association for Computational Linguistics. 37, 38

Xiao, Z., McEnery, A.M., Baker, P. & Hardie, A. (2004). Developing Asian language corpora: standards and practice. *Fourth Workshop on Asian Language Resources*, 1–8. 29

Yarowsky, D. & Ngai, G. (2001). Inducing Multilingual POS Taggers and NP Bracketers via Robust Projection Across Aligned Corpora. In *Proceedings of NAACL2001*, 200–207, Morgan Kaufmann. 9