Context-dependent Type-level Models for Unsupervised Morpho-syntactic Induction

by

Yoong Keok Lee

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2015

Author . . . .
Department of Electrical Engineering and Computer Science
October 31, 2014

Certified by . . . .
Regina Barzilay
Professor, Electrical Engineering and Computer Science
Thesis Supervisor

Accepted by . . . .
Leslie A. Kolodziejski
Professor, Electrical Engineering and Computer Science
Chairman, Department Committee on Graduate Students

Context-dependent Type-level Models for Unsupervised Morpho-syntactic Induction

by

Yoong Keok Lee

Submitted to the Department of Electrical Engineering and Computer Science
on October 31, 2014, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

This thesis improves unsupervised methods for part-of-speech (POS) induction and morphological word segmentation by modeling linguistic phenomena previously not used. For both tasks, we realize these linguistic intuitions with Bayesian generative models that first create a latent lexicon before generating unannotated tokens in the input corpus. Our POS induction model explicitly incorporates properties of POS tags at the type-level which is not parameterized by existing token-based approaches. This enables our model to outperform previous approaches on a range of languages that exhibit substantial syntactic variation. In our morphological segmentation model, we exploit the fact that affixes are correlated within a word and between adjacent words. We surpass previous unsupervised segmentation systems on the Modern Standard Arabic Treebank data set. Finally, we showcase the utility of our unsupervised segmentation model for machine translation of the Levantine dialectal Arabic for which there is no known segmenter. We demonstrate that our segmenter outperforms supervised and knowledge-based alternatives.

Thesis Supervisor: Regina Barzilay
Title: Professor, Electrical Engineering and Computer Science

Acknowledgments

I am a believer that the community shapes this thesis in one way or another. But if I just have to name one person, she has to be my advisor, Regina Barzilay. Her sharp instinct for impactful research directions and unyielding pursue for excellence never cease to inspire me. This is something you cannot learn from papers. Regina, thank you for honing me into the researcher I am today. Needless to say, this thesis is possible also because of my collaborators Aria Haghighi and David Stallard and his colleagues at BBN Technologies.

I am fortunate to be immersed in a stimulating environment. My thesis committee Tommi Jaakkola and Jim Glass are always able to bring fresh perspectives from their area of expertise. I also benefit greatly from seemingly useless [108] conversations with members and friends of my research group, past and present: Tahira Naseem, Christy Sauper, Harr Chen, S.R.K. Branavan, Amir Globerson, Roi Riechart, Ben Snyder, Jacob Einstein, Yevgeni Berzak, Karthik Narasim, Tao Lei, Yonatan Belinko, Nate Kushman, and many others, who of course includes our administrative assistant Marcia Davidson. She makes research possible without having to do research per se. These discussions help me to piece together a better understanding of the universe of which this thesis is a subset.

Lastly, this thesis is also due to the cumulative support of my other colleagues at the Computer Science and Artificial Intelligence Laboratory, the department of Electrical Engineering and Computer Science, members of the MIT community at large, friends, and family.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Statistical methods have become dominant in natural language processing (NLP) research today. Most of the advancements have been achieved in the area of supervised learning, where algorithms rely on annotated data to learn linguistic structure. Commonly these annotations are compiled manually. The difficulty in obtaining annotations for new tasks and new languages has motivated research in unsupervised methods. These methods aim to learn linguistic structure directly from raw data. Despite much progress, the performance of unsupervised methods still lags behind their supervised counterparts. With only a few notable exceptions (like word alignment modules in machine translation systems), supervised models are the only source of reliable components for applications.

In recent years, there is a growing body of work that incorporates linguistic knowledge to improve unsupervised models. One approach encodes complementary grammar formalisms in the same model to increase robustness. The dependency grammar and the constituency grammar represent syntactic parse trees as head-modifier edges and nested phrases respectively. Klein and Manning [67] propose a model that generates parse trees from the product of individual probabilistic models for each formalism. Another method employs knowledge about ungrammatical sentences can be generated from grammatical ones to improve estimation of unsupervised models. The constrastive estimation approach of Smith and Eisner [114] create negative examples from natural free text by minor word order perturbations and optimize model parameters to distinguish between them. Unsupervised models, like supervised ones, benefit from richer representations but have to balance the cost of data sparsity. A number of researchers enhance unsupervised parsers by using more complex grammar structures in the model and performing smoothing, for example with back-off strategies [57] and Bayesian priors [23]. An analysis of unsupervised parsers often reveals gross errors that can be easily rectified by rules, for example erroneous syntactic dependencies that says verbs to determiner. One recent development allows one to guide unsupervised learning by specifying linguistic knowledge declaratively. The posterior regularization [43] or generalized expectation Druck et al. [31] method guides learning of probabilistic models so they assign low probability to outputs that violate pre-specified constraints. Using this framework, Naseem et al. [91] encourage unsupervised parsers to produce outputs that are consistent with universal syntactic dependency rules.

1.1 This Thesis

This thesis takes the approach of modeling linguistic constraints to tame unsupervised models for part-of-speech (POS) induction and morphological word segmentation.

POS Induction Given an unannotated corpus comprising of sentences, the goal is to tag the syntactic category of each word, for example:

Input:	Unlabeled	unsegmented	words	are	abundant
Output:	T_1	T_1	T_2	T_3	T_1

The main feature of our approach is to perform POS induction at the level of word types level using a Bayesian generative model. Traditionally, POS categories are assigned to words at the token-level, based on its surroundings as Church [19] succinctly puts it in one of the earliest papers on supervised hidden Markov model (HMM) taggers:

> It is well-known that part of speech depends on context . . . A program has been written which tags each word in an input sentence with the most

likely part of speech.

However, allowing different token occurrences of the same word type to take different POS labels leaves too much freedom for unsupervised taggers [59]. We empirically validate properties about POS tags at type-level across 14 languages and encode this knowledge into a Bayesian (context-dependent) HMM.

Morphological Word Segmentation For this task, we develop a series of models that take as input a corpus of unsegmented and untagged words. The goal is to divide each word into a sequence of linguistically meaningful substrings called *morphemes*. The low-order models merely expect a set of word types, for example:

Input	**Output**
unlabeled	un–label–ed
unsegmented	un–segment–ed
segmented	segment–ed
segmenting	segment–ing

When sentences are provided, our high-order models exploit the contexts in which each word type appears to improve segmentation performance, for example:

Input:	Unlabeled	unsegmented	words	are	abundant
	The	training	corpus	is	unsegmented
Output:	Un–label–ed	un–segment–ed	word–s	are	abundant
	The	train–ing	corpus	is	un–segment–ed

Our morphological word segmentation model exploits the key linguistic knowledge that there are correlations between morphemes due to role of syntax in morphology and vice versa. The long-established approach for morphology, which is the study of the internal structure of words, are analyzed at the type-level. This is exemplified by the pioneering technique of Harris [53] decides if a division should be made based on its local-boundary statistics. The relationship between syntax and morphology has been studied deeper, and it has become more apparent in the linguistic literature that there is a close connection between syntax and morphology [52]. It is the dependencies between syntax and morphology that allow our higher-order models to also utilize the context in which the word appears.

Conceptual Framework A common theme in both our models is that they combine contextual (token-level) information with desirable properties of the lexicon (type-level). For both models, we postulate that an unobserved stochastic process first generates latent lexicons consisting of syntactic categories and morpheme that make up the word types. In the second step, the probabilistic process continues to generate sentences from these lexicons. Although the process is random, not all latent lexicons generate the same input corpus with equal probability; it is more likely to create the corpus from some lexicons than others. The task of morpho-syntactic induction is cast as a problem of inferring the likely latent structures from which the input corpus is generated. We take the Bayesian view that the underlying mechanism is parameterized not by one single set of fixed (but unknown) random variables. In other words, there is uncertainty in the parameters, and they are governed by some probability distributions. As such, when we infer the underlying lexicons, we recover not a single output but a distribution over a set of outputs. In our experiments, we evaluate each output quantitatively (see Section 2.5 and Section 3.5 for details) and report the mean scores.

Mathematically, both models have the following generative process:

$$P(\boldsymbol{r},\boldsymbol{t},\boldsymbol{w}) = \underbrace{P(\boldsymbol{L})}_{\text{lexicon}} \cdot \underbrace{P(\boldsymbol{w},\boldsymbol{r}|\boldsymbol{L})}_{\text{tokens}},$$

where $\boldsymbol{L}$ denote a sparse type-level lexicon. In fact, in our models each word type is assigned exactly one latent representation — a tag for POS induction, and a tag and a segmentation for morphology induction. Here, $\boldsymbol{w}$ and $\boldsymbol{r}$ denote word tokens and their corresponding latent representations of interest respectively. (For brevity, we omit hyperparameters which specify the distributions from which parameters drawn.)

The token-level component is parameterized by a HMM:

$$P(\boldsymbol{w}, \boldsymbol{r}|\boldsymbol{L}) = \prod_j \underbrace{\prod_k P(r_k^{(j)}|r_{k-1}^{(j)})P(w_k^{(j)}|r_k^{(j)}, \boldsymbol{L})}_{j\text{-th sentence}},$$

such that word-representation assignments specified in the lexicon are respected. In other words, a tag cannot generate a word if that pair is not an entry in the lexicon. And for morphological word segmentation, each (unsegmented) word token must be generated by a segmentation representation that is consistent with the lexicon.

The product of the two components couples type-level and token-level modeling signals in a joint model. This is exactly what enables our context-dependent model to learn type-level representations. Apart from the ability to model linguistic knowledge in a modular fashion, conditioning the token-level component on the lexicon also allows us to restrain the flexibility of standard unsupervised token-level models.

Recovering latent tags or segmentations amounts to sampling lexicon $\boldsymbol{L}$ from the posterior distribution[1]:

$$P(\boldsymbol{L}, \boldsymbol{r}|\boldsymbol{w}) \propto P(\boldsymbol{L}) \cdot P(\boldsymbol{w}, \boldsymbol{r}|\boldsymbol{L})$$

Section 1.2 and 1.3 introduce our main modeling ideas for POS induction and morphological word segmentation respectively. Section 1.4 presents our approach to applying our unsupervised segmentation model to machine translation of dialectal Arabic for which there is no known segmenter. Section 1.5 summarizes the contributions of our work. Finally, Section 1.6 provides a road map of this thesis.

[1] We omit some rigor here for clarity. For POS induction, word types $\boldsymbol{W}$ are observed so we only sample type tags $\boldsymbol{T}$. Section 2.3 flashes out the procedure in more details.

1.2 Part-of-Speech Induction

1.2.1 Background

Sequence-based models are not only the mainstay of supervised taggers, but also have become a cornerstone of unsupervised POS induction systems. (The latter do not require examples of how words should be tagged in context.) Although coined unsupervised, early work continues to rely on a tagging lexicon which lists POS candidates for each word type. For instance, Mérialdo [83] employs the lexicon to constrain Viterbi decoding in expectation maximization (EM) estimation of a unsupervised HMM tagger. However, as Banko and Moore [7] have shown, the accuracy of unsupervised taggers is sensitive to the quality of the lexicon. This observation spurs research that can accept an incomplete tagging lexicon. To put numbers in perspective, state-of-the-art supervised taggers of Collins [24] and many others [81] give an accuracy of about 97% on English. Assuming the availability of a tagging lexicon, sophisticated unsupervised taggers of Goldberg et al. [38], Ravi and Knight [103] achieve around 91% (also on the same corpus although there is some differences in experimental settings). A regular unsupervised HMM trained with EM achieves around 81–83% [40, 103]. But once rare word types are removed from the lexicon, the accuracy drops drastically to as low as 50%. The reliance on the tagging lexicon is further relaxed, leading to unsupervised Bayesian HMM tagger of Johnson [59] that operates in the absence of any lexicon. POS tags, however, are still continued to be assigned at the token-level, and accuracies remain in the 37–62% range, depending on tunable parameters and evaluation settings.

1.2.2 Linguistic Intuitions

Our POS induction model integrates the contextual dependent nature of POS tagging with two three modeling signals.

Words and tags show distinctive type-level distribution Words which share the same token-level distribution can show very different distribution at the type-level.

For instance in English, determiners (such as “the” and “a”) occur very frequently at the token-level (about 9%) but are rare at the type-level (about 0.05%). Proper nouns (i.e. names), which have about the same token-level frequency (around 10%), form a much larger part of the lexicon (at 29%). A standard HMM, which generates word and tag tokens, does not have any parameters to capture tag distribution in the lexicon explicitly.

POS tags exhibit sparsity Although natural language is inherently ambiguous, words are likely to take a single predominant tag in a corpus. In fact, assigning the most frequent tag to each word achieves about 95% accuracy on the Wall Street Journal portion of the Penn Treebank corpus. This phenomenon is not unique to English. In 13 other languages which have substantial syntactic variation (such as Arabic, Chinese, Czech, German, Japanese, Slovene, and Turkish), the upper bound of this “one tag per word” baseline is greater than 90%, which far exceeds the state-of-the-art unsupervised POS induction systems that are not given on any tagging lexicon.

Orthographic features correlate with syntactic categories Most languages employ morphological features to mark syntactic categories of its words. In English, verbs use suffix “ing” and “ed” to indicate present continuous and past tenses respectively. Morphological rich languages convey even more information in orthographic markers. For example, in Arabic, suffix “A” is an encoding for an imperfect, third person, dual, and masculine verb. In addition, other orthographic features also provide cues the POS of a word, for example capitalization hints that the word is a proper noun.

1.2.3 Technical Approach

Here, the lexicon comprises of word-tag pairs, $\boldsymbol{L} = (\boldsymbol{W}, \boldsymbol{T})$ and latent representations are POS tags, $\boldsymbol{r} = \boldsymbol{t}$. We decompose our lexicon model as follows

$$P(\boldsymbol{T}, \boldsymbol{W}) = P(\boldsymbol{T}) \cdot P(\boldsymbol{W}|\boldsymbol{T})$$

The tag component which factorizes as $P(\boldsymbol{T}) = \prod_i P(T_i)$ explicitly captures tag distribution in the lexicon we desire. To model tag sparsity, we impose the one-tag-per-word constraint in the distribution $P(\boldsymbol{W}|\boldsymbol{T})$ by placing zero probability mass over invalid configurations. To introduce dependencies between POS tags and orthographic features (which includes morphological suffixes that are obtained with an off-the-shelf unsupervised segmenter), we generate the feature-value pairs of each word type independently:

$$P(W_i|T_i) = \prod_{(f,v) \in W_i} P(v|\psi_{T_i f}),$$

where the probability of generating a feature value v for a word type W_i also depends on its tag T_i. This completes the lexicon model. Sentences are generated with a HMM that satisfy word-tag assignments in the lexicon as described earlier.

A distinctive feature of our Bayesian tagger is that when we employ Gibbs sampling to sample from the posterior, we tie token POS tags belonging to the same word type. This is in contrast to the standard Bayesian tagger which samples POS tag one token at a time, and thus requires more iterations for convergence. In the sampling equations, terms generating word occurrences at different sites become dependent (due to the unobserved parameter that generates them). Section 2.4 details the derivation which leads to an expression involving ascending factorials that can be computed efficiently. The model so far requires a number of tunable (hyperparmeters) settings. To reduce the amount of tunable hyperparameters, we extend the Bayesian hierarchy by assuming that these hyperparameters are drawn from the vague Gamma probability distribution which is specified by just two parameters. We use the same

Gamma distribution for all 14 languages used in the evaluation.

1.2.4 Findings

Main empirical results On a collection of data sets amounting to 14 languages, our POS induction system surpasses the best performing systems prior to our conference publication [74] in 2010 on 11 languages. Till date, our model is still the state-of-the-art for Bulgarian, German, Japanese, and Slovene in these data sets.

Class-based HMMs have been previously proposed by Brown et al. [13] and Clark [20]. In their word clustering HMMs, the same one-tag-per-word constraint is imposed. Clark's model is perhaps most similar to our model because morphological features of words are also incorporated. There are two primary differences: Firstly, our model includes a tag distribution at the type level. Secondly, our inference method is based on algorithms specifically designed for Bayesian graphical models, whereas they employ greedy local optimization heuristics. Our experiments suggest that these reasons account for our higher accuracy averaged over 14 languages.

In comparison to methods that encourage tag sparsity and employ features, our inference is simpler and more effective. Berg-Kirkpatrick et al. [8] modify the emission distribution of the standard HMM take a log-linear parameterization. This allows them to encode orthographic features of words without assuming feature independence. The m-step of expectation maximization (EM) estimation of their HMM, however, loses the closed form solution. In contrast we assume feature independence, and this allows use to continue to use Gibbs sampling for inference. On the other hand, Graça et al. [43] encourages tag sparsity by performing posterior regularization, i.e. the posterior distribution now has to respect linear constraints which can be specified to encourage word types to take a small number of tags. Again, EM estimation of HMM loses its closed form. Although optimization in each EM iteration can be formulate as a convex problem in the dual, a more elaborate gradient-based method has to be used. Our experiments also show that our model adopts a design trade-off is generally more effective.

1.3 Morphological Word Segmentation

1.3.1 Background

Morphology is rich field concerning the internal structure of words, and one of the most influential representation is developed by Hockett [58] and Harris [53]. Hockett's *item-and-arrangement* model of morphology posits a word is formed by first picking a set of minimal building blocks called *morphemes* and then arranging them in the desired sequence. Harris proposes a corpus-based procedure of morphological analysis by segmenting a word into its constituent morphemes. This is the representation we adopt in this thesis. Specifically, given a corpus consisting of just words, our goal is obtain morphological segmentations for each word.

We would also like to point out that there are two other commonly adopted morphological schemes. For instance, the *item-and-process* [58] model explains how related words can obtained by applying rules to transform the orthographic form of a base morpheme. On the other hand, the *word-and-paradigm* approach [58] explains word formation without resorting to morphemes. In this framework, a word and its morphological variants are grouped into a paradigm. Words that belong to the same paradigm are modified in the same way. Figure 1-1 contrasts these three schemes with some examples.

Contrary to token-based approach to POS tagging, morphological analysis is performed at the lexicon level. The central idea of Harris [53] is that at morpheme boundaries, the preceding or trailing substring can be easily composed with other substrings to form valid words. Given a word, he proposes several metrics for scoring each possible boundary using statistics of surrounding characters computed from a corpus of word types. The word is then segmented using a set of heuristic rules that operate on these boundary scores. His work inspired subsequent research not only in computational linguistic but also related fields such as psycholinguitics and speech processing, where the goal is to determine boundaries of word tokens in utterances or

Word	Morphemes
unsegmented	un – segment – ed
unsupervised	un – supervis – ed
segmenting	segment – ing
supervising	supervis – ing

(a) Item and arrangement

Root word		Past Tense
segment	$\xrightarrow{+ed}$	segmented
supervise	$\xrightarrow{-e,+ed}$	supervised
write	$\xrightarrow{-i,+o}$	wrote

(b) Item and process

Inflectional form	Paradigms		
	I	II	III
Infinitive	segment	supervise	write
Present tense	segments	supervises	writes
Present continuous	segmenting	supervising	writing
Past tense	segmented	supervised	wrote
Past participle	segmented	supervised	written

(c) Word and paradigm

Figure 1-1: Three major models of word morphology: (a) The item and arrangement model represents a word as a concatenation of morphemes. This is the scheme we adopt in this thesis. (b) The item and process model explains word formation as applying rewrite rules to a base morpheme. (c) The word and paradigm model posits that morphological variants of words belonging to the same paradigm are obtained analogously. For example, the word "decide" belongs to class II.

sentences. His approach is mainly extended by devising new boundary scoring functions and segmentation rules. This thread of algorithms, however, does not consider the optimality of the collective set of morphemes that are recovered. A second thread of work inspired by Olivier [99] complements the local approach by explicitly learning a lexicon of morphemes from corpus. Such approach is further expanded to include generative models that explain how a corpus of unsegmented words are formed with a morpheme lexicon or morpheme grammars. This also forms the basis of minimum description length (MDL) models that achieve a balance between recovering a compact lexicon (or grammar) and explaining the unsegmented corpus well. Nevertheless, computational models of segmentation remain independent of the syntactic category of the word and its context.

1.3.2 Linguistic Intuitions

Our morphological segmentation work is novel in two aspects:

- We incorporate the role of (token-level) context into a type-level segmentation model.
- Our model explicitly models how syntax influences the structure of words. (Note that the corpus does not need to be annotated with POS tags or syntactic dependencies.)

Morphological consistency within POS categories Words within the same syntactic category tend to have similar affixes. In other words, affixes in the same word are not independent or another. And some affix combinations are more likely than others. In English, prefix "un" is compatible with verb suffix "ed" but not noun suffix "s". In Arabic, the prefixing determiner "Al"[2] can be selected with noun suffix "At" but not verb suffix "A". This, for example, is effective for picking the correct segmentation between the two candidates: "Al–{ntxAb–At" (translated as "the–elections–s") and "Al–{ntxAb–A–t" (erroneous segmentation).

[2] Here, we use the Buckwalter transliteration is an one-to-one mapping between Arabic characters and English alphabets.

Morphological realization of grammatical agreement In morphologically rich languages, agreement is commonly realized using matching suffices. Word pairs that form a dependency, such as adjective and noun, often have the same suffix. A common example in the Arabic Treebank is the bigram "Al–Df–p Al–grby–p" (which is translated word-for-word as "the–bank the–west") where the last morpheme "p" is a feminine singular noun suffix.

1.3.3 Technical Approach

In contrast to our POS induction model, the lexicon component of the segmentation model has a number of dictionaries that specify (latent) morphemes, POS tags, and word segmentations. The latent representations consist of POS tags and word segmentations, i.e. $\boldsymbol{r} = (\boldsymbol{t}, \boldsymbol{s})$. The generative process first draws a master morpheme lexicon L^*. Given the master lexicon, we generate one sub-lexicon for each of the morpheme types: prefix, stem, and suffix, denoted L_-, L_0, and L_+ respectively. The generative process for morpheme lexicons are modeled with basic morphological intuitions, such as smaller lexicons and shorter morphemes are preferred, by drawing the length of each morpheme $\sigma \in L^*$ and number of entries in each of the sub-lexicons from geometric distributions:

$$\begin{aligned}
\text{morpheme length:} \quad & \sigma \quad\ \sim \text{Geometric} \\
\text{prefix:} \quad & |L_-| \sim \text{Geometric} \\
\text{stem:} \quad & |L_0| \sim \text{Geometric} \\
\text{suffix:} \quad & |L_+| \sim \text{Geometric}
\end{aligned}$$

Once we have generated these morpheme lexicons, we draw the POS tag of a word T as before. Conditioned on T, we then draw its constituent morphemes to compose the segmented word type.

Let σ_- and σ_+ denote a prefix and a suffix respectively. To introduce dependencies between affixes within a word, we draw its affixes conditioned on its POS tag T:

$$\begin{aligned} \text{prefix:} \quad & \sigma_{-}|T \sim \text{Multinomial} \\ \text{suffix:} \quad & \sigma_{+}|T \sim \text{Multinomial} \end{aligned}$$

This is how we realize our first modeling intuition that there is morphological consistency with POS categories. Because the POS tag is unobserved, it follows from properties of directed graphical models that the affixes (within a word) become dependent on another. This completes the generative process that creates the word lexicon which assigns one tag and one segmentation to each word type.

Next, we proceed to generate sentences comprising of segmented words $\boldsymbol{s}$ and their POS tags $\boldsymbol{t}$. The observed unsegmented words $\boldsymbol{w}$ are trivally created by removing the morpheme boundaries in $\boldsymbol{s}$. The process is specified with a HMM that respects the lexicons:

$$P(\boldsymbol{w}, \boldsymbol{s}, \boldsymbol{t}|\boldsymbol{L}) = \underbrace{\prod_i P(t_i|t_{i-1})P(s_i|t_i, \boldsymbol{L})}_{\text{HMM}} \cdot \underbrace{\prod_i P(w_i|s_i)}_{\text{deterministic}}$$

Now that there are dependencies between POS tags of adjacent words, their affixes become dependent when the tags are unobserved. The token emission has to adhere to lexicon assignments, i.e. a segmented word s_i can only be generated by tag t_i if the pair is specified by the lexicon. This modeling technique enables type-level learning to be also context-dependent.

To encode our second linguistic knowledge that grammatical agreement is commonly realized using matching suffixes, we extend the token-level HMM by generating segmented tokens $\boldsymbol{s}$ again:

$$P(\boldsymbol{s}) = \prod_i p(s_i|s_{i-1})$$

Another way of viewing this component is adding a correction factor which overgenerates segmentated tokens in a HMM:[3]

$$P(\boldsymbol{s}, \boldsymbol{t}) = \prod_i \underbrace{p(s_i|s_{i-1})}_{\text{correction factor}} \cdot P(t_i|t_{i-1})P(s_i|t_i).$$

We pre-specify and fix $p(s_i|s_{i-1})$ to encourage adjacent tokens s_{i-1} and s_i to have similar suffix if they end with the same substrings. (Same endings are an indication that the pair of words participate in grammatical agreement.) The type-level Gibbs sampling inference method for POS induction applies to this model as well and is detailed in Section 3.4.

1.3.4 Findings

Main empirical results As in our POS induction model, our segmentation model improves as more components are added. Our final model outperforms the best system on the (Modern Standard) Arabic Treebank before our conference publication [75] in 2011.

Our work is most closely related to the approach of Can and Manandhar [16]. Their algorithm also combines POS-based clustering and morphological segmentation. Their method learns POS clusters in a separate preprocessing stage using distributional cues. For each cluster, their model picks a set of affixes depending on the frequency of their occurrences in the cluster. Perhaps because of the suboptimality of having two separate steps, their system does not outperform the state-of-the-art language independent segmenter of Creutz and Lagus [26]. Our approach differs in two ways that allows our model to be more effective. Firstly, our model integrates morpho-syntactic components in a joint generative model. Secondly, we can incorporate contextual dependencies into type-level segmentation. We would also like to point out that Toutanova and Cherry [124] were the first to develop a model that reconcile part-of-speech tagging with lemmatization decisions, although the problem formulation is different. They consider a semi-supervised setting where initial mor-

[3] This final component makes the model deficient, just like the higher-order IBM word alignment models, although empirically it improves performance.

phological and tagging lexicons are provided together with access of unlabeled data.

Our main point of empirical comparison is the unsupervised segmenter of Poon et al. [101]. Their model generates word types along with their segmentations using an undirected graphical model which allows them to incorporate arbitrary features, such as the number of morpheme types and local n-gram patterns around word boundary. However, they do not model correlations of affixes within a word or across word tokens. Our experiments on the Arabic Treebank data set point to these signals as the source of gains achieved by our model.

1.4 Unsupervised Morphological Segmentation for Machine Translation of Dialectal Arabic

We revisit the motivation for emergence of unsupervised models in NLP research — annotations for complex language structures are laborious to obtain, and unsupervised models are most useful where unlabeled data are abundant. This section shows how to effectively employ unsupervised morphological segmenters to improve machine translation, particular for the Levantine dialect for which there is no known segmenter. This work also showcases the utility of morphology in a end-to-end application of NLP technology.

1.4.1 Background

Stemming, the process of removing suffixes from a word, is perhaps the most well-known form of morphological segmentation. This process combines frequency counts of morphological variants of the same word, and thus helps to reduce data sparsity. In fact, this preprocessing is so crucial that it has become a defacto step in information retrieval models [6, 111]. With the same objective, we apply our unsupervised morphological segmenter to preprocess training data for machine translation (MT).

There are several similar threads of work in MT of inflectional languages. A body of work [45, 79, 88, 123] incorporate morphological information directly in the

MT system by modifying the architecture. For instance, factored translation Models [4, 69, 134] operate at the level of unsegmented words, but they parameterize phrase translation probabilities as factors that encode morphological features. Other approaches translate at the word level, but correct translation outputs for morphological rich target languages [86, 127].

1.4.2 Technical Approach

Our approach belongs to the family of segmented MT model which divides the input into morphemes and uses the derived morphemes as a unit of translation [5, 21, 29, 41, 98, 102, 109]. Our work is most closely related to the class of MT systems that apply unsupervised segmenter to obtain these morphemes [21, 26, 129]. Virpioja et al. [129] employ the unsupervised morphological segmenter Morfessor [26], and apply an existing MT system at the level of morphemes. The system does not outperform the word baseline partially due to the insufficient accuracy of automatic morphological analyzer.The system of Clifton and Sarkar [21] also uses Morfessor output but in a different translation architecture that post-processes Morfessor's deficiencies.

The work of Mermer and Akın [84] and Mermer and Saraclar [85] attempts to integrate morphology and MT more closely than we do in a Turkish-to-English MT system that uses bilingual alignment probabilities. However, their strategy shows no gain over the monolingual version, and neither version is competitive for MT with a supervised Turkish morphological segmenter [97]. In contrast, the unsupervised analyzer we report on here yields MSA-to-English MT performance that equals or exceed the performance obtained with a leading supervised MSA segmenter, MADA [46].

To increase robustness of our segmenter on large corpus, we perform *maximum marginal decoding* [61]. For each word of interest, this decoding method marginalizes all other latent variables, i.e. POS tags and segmentations of other words. We obtain a Monte Carlo approximation as follows: we first draw independent samples from the posterior, then perform majority vote on each word segmentation.

After word morphemes are separated, we feed the training corpus to a the state-of-the-art string-to-dependency-tree MT system of Shen et al. [113]. The MT system performs decoding with a 3-gram target language model generates the N-best unique translation hypotheses, and then reranks them using a 5-gram language to select the best-scoring translation. The decoder model parameters are tuned using Minimum Error Rate Training (MERT) [94] to maximize the IBM BLEU score [100].

1.4.3 Findings

Main empirical results On the Arabic Treebank data set, maximum marginal decoding improves our previous results and surpasses the state-of-the-art before our conference publication [121] in 2012. On the 1.5M-word Levantine dialectal MT corpus of Zbib et al. [136], our segmenter yields an 18% relative BLEU gain over supervised or knowledge-based alternatives.

Specifically, we compare against other variants of our MT system by substituting our segmenter with other alternatives: (1) the straw baseline of not performing segmentation, (2) Sakhr: a commercial rule-based Modern Standard Arabic (MSA) morphological analyzer, (3) MADA: a top-performing supervised analyzer tailored for MSA, (4) Morfessor [26]: an unsupervised language-independent segmenter.

In addition to Levantine, we also evaluate our MT systems in two other settings: (1) The NIST MT-08 Constrained Data Track Arabic corpus which consists of 35M total words, with a vocabulary of 336K unique Arabic words, (2) A small 1.3M-word subset of the MT-08 corpus. On the MSA data sets, the heavily engineered Sakhr unsurprisingly outperforms all segmenters. However, our segmenter performs on par with MADA on the full MSA setting. On the small MSA setting, our segmenter outperforms MADA. We consistently outperform Morfessor, a unsupervised language-independent segmenter.

1.5 Contributions

The contributions of this thesis are three-fold:

(1) **Model type-level tag properties to improve unsupervised POS induction** We develop a Bayesian hidden Markov model that labels word types. The main source of gains come from the restricting each word type to take only one tag, introducing a distribution over tags at the type-level, and engineering orthographic features that correlate with syntactic categories. This in combination with Monte Carlo Markov Chain inference techniques enables our model to achieve the best-performing results for 11 out of 14 languages in the year of our conference publication [74]. Till date, our model is still the start-of-the-art for Bulgarian, German, Japanese, and Slovene.

(2) **Model connection between syntax and morphology to improve unsupervised morphological word segmentation** We exploit the fact that morphemes, within and across words, are correlated due to the mutual influence of syntax and morphology. This linguistic intuition is translated into a computational model by employing a generative process that first generates a lexicon of latent POS tags and morphemes. Using on the lexicon, sentences are generated so that adjacent words that participate in grammatical agreement are encouraged to have compatible suffixes. On the Arabic Treebank, our model surpasses the best system prior to our conference publication [75].

(3) **First to demonstrate the effectiveness of unsupervised morphological segmentation in dialectal Arabic Machine Translation (MT)** We show that unsupervised word segmentation model outperforms supervised ones for machine translation (MT). We apply our unsupervised word segmentation model to the Levantine Arabic dialect for which there is no tailored segmenter. Using our segmentation preprocessing, the MT system gives an 18% relative BLEU gain over supervised or knowledge-based alternatives, including a commercial segmenter developed for Modern Standard Arabic.

1.6 Outline

The remainder of the thesis is organized as follows:

- **Chapter 2** presents our unsupervised Bayesian POS induction HMM. In addition to using token-level context information, our model parameterizes tag and orthographic features distributions at the type-level.

- **Chapter 3** describes our context-aware morphological word segmentation model. Besides incorporating traditional type-level cues, the model also introduces dependencies between affixes both within a word and across adjacent words.

- **Chapter 4** discusses how our unsupervised word segmenter surpasses supervised alternatives for machine translation of dialectal Arabic. It also describes maximum marginal decoding that improves word segmentation performance on the Arabic Treebank.

- **Chapter 5** summarizes main points of this thesis.

Chapter 2

Unsupervised Bayesian Type-Level POS Tagging

2.1 Introduction

In this chapter, we consider the task of unsupervised part-of-speech (POS) induction, i.e. given only a corpus comprising of just words, the goal is to assign a syntactic category to each token. Since the early days of statistical NLP, POS induction systems have assumed the availability of a *type-level* tagging lexicon which lists the set of valid POS tags for each word type. This assumption is crucial to the success of traditional hidden Markov model (HMM) taggers which capture regularities of tagging behavior at the *token-level* [19, 83]. The availability of a tagging lexicon makes unsupervised POS learning feasible by dramatically constraining the search space. Being inherently type-level, such constraints are difficult to incorporate in a token-level HMM in the absence of a tagging dictionary.

As a result, recent work finds alternative ways to enforce these constraints, while staying within the framework of a token-driven approach [8, 43]. Most notably, researchers have observed that a POS tag distribution exhibits "one tag per discourse" *sparsity* — words are likely to select a single predominant tag in a corpus, even when several tags are possible. Simply assigning to each word its most frequent associated tag in a corpus achieves 94.6% accuracy on the WSJ portion of the Penn Treebank.

Language	Original case	No case
English	94.6	92.6
Arabic	95.1	95.1
Bulgarian	97.9	97.8
Chinese	92.9	92.9
Czech	99.2	99.1
Danish	96.3	96.1
Dutch	96.6	96.2
German	95.5	94.8
Japanese	94.0	94.0
Portuguese	95.5	95.3
Slovene	98.5	98.4
Spanish	95.4	95.1
Swedish	93.3	93.0
Turkish	91.9	91.7

Table 2.1: Upper bound on tagging accuracy assuming each word type is assigned to majority POS tag. Across all languages, high performance can be attained by selecting a single tag per word type. When the case is collapsed, words with distinct predominant POS tag (for example the proper name "Trading" and the verb "trading") are combined, and this results in a slightly lower upper bound that still exceeds 90% for all languages. The English data is obtained from the Penn Treebank Wall Street Journal corpus, whereas the rest comes from the ConLL-X shared task data set.

This distributional sparsity of syntactic tags is not unique to English — similar results have been observed across multiple languages. As can be seen in Table 2.1, for all 14 languages considered here, upper bound on performance exceeds 90%. Clearly, explicitly modeling such a powerful constraint on tagging assignment has a potential to significantly improve the accuracy of an unsupervised part-of-speech tagger learned without a tagging dictionary.

In practice, this sparsity constraint is difficult to incorporate in a traditional POS induction system [8, 36, 43, 59, 83]. These sequence models-based approaches commonly treat token-level tag assignment as the primary latent variable. By design, they readily capture regularities at the *token-level*. However, these approaches are ill-equipped to directly represent *type-based constraints* such as sparsity. Previous work has attempted to incorporate such constraints into token-level models via heavy-handed modifications to inference procedure and objective function, for example posterior regularization [43] and integer linear programming decoding [103]. In most cases, however, these expansions come with a steep increase in model complexity, with respect to training procedure and inference time.

In this work, we take a more direct approach and treat a word type and its allowed POS tags as a primary element of the model. The model starts by generating a tag assignment for each word type in a vocabulary, assuming one-tag-per-word. Then, token-level HMM emission parameters are drawn conditioned on these assignments such that each word is only allowed probability mass on a single assigned tag. In this way we restrict the parameterization of a token-level HMM to reflect lexicon sparsity. This model admits a simple Gibbs sampling algorithm where the number of latent variables is proportional to the number of word types, rather than the size of a corpus as for a standard HMM sampler [59].

There are two key benefits of this model architecture. First, it directly encodes linguistic intuitions about POS tag assignments: the model structure reflects the one-tag-per-word property, and a type-level tag prior captures the skew on tag assignments (e.g., there are fewer unique determiners than unique nouns). Second, the reduced number of hidden variables and parameters dramatically speeds up learning and inference.

We evaluate our model on 14 languages exhibiting substantial syntactic variation. On several languages, we report performance exceeding that of state-of-the art systems. Our analysis identifies three key factors driving our performance gain: 1) selecting a model structure which directly encodes tag sparsity, 2) a type-level prior on tag assignments, and 3) a straightforward naïve-Bayes approach to incorporate features. The observed performance gains, coupled with the simplicity of model implementation, makes it a compelling alternative to existing more complex counterparts.

In the next section we review related work. Section 2.3 presents our model, and Section 2.4 describes the inference algorithm. In the absence of a tagging lexicon, an unsupervised tagger outputs a label set that is not directly comparable to the POS tag set used for annotation (see Figure 2-1 for an example). Section 2.5 specifically

I	love	dogs
T_1	T_2	T_3
Dogs	love	bones
T_3	T_2	T_3

Figure 2-1: Unsupervised POS tagging without a tagging lexicon. Without knowledge of the tag set, each token is assigned an arbitrarily-named tag. Our type-level tagger further constrains all occurrences of a word type to have the same tag. Section 2.5 discusses how such outputs are evaluated quantitatively.

addresses the issue of experimental setup and evaluation. We present empirical results in Section 2.6 and summarize our findings in Section 2.7.

2.2 Related Work

2.2.1 Partially Supervised POS Tagging

Although all unsupervised part-of-speech (POS) induction research operates on the fundamental premise that no labeled word tokens are given, early research has relied on auxiliary knowledge. Specifically, the requisite for a tagging lexicon is apparent in the two main approaches for unsupervised POS tagging. One method which is exemplified by Mérialdo [83] formulates unsupervised POS induction as recovering the latent states of a hidden Markov model (HMM) that generates word tokens. In the other framework, Brill [12] iteratively transforms non-probabilistic rules that tag each word in context. Regardless of the difference in how tokens are disambiguated, the overarching assumption is that there is a tagging lexicon to provide the set of valid POS tags for each word at the type-level. Such a tagging lexicon bounds ambiguity and therefore dramatically reduces the search space in unsupervised learning. The performance, however, depends critically on the quality of the tagging dictionary [7]. For instance, on a recently used 24K-word Wall Street Journal (WSJ) evaluation setup [114], a HMM learned with the EM algorithm achieves 83% [40]. When lexicon entries of rare words are removed (hence forcing EM to consider all possible POS tags for them), the accuracy plunges to as low as 50%.

Since then, researchers have devised POS taggers that rely on weaker assumptions. Banko and Moore [7] allow noisy tagging lexicons to be used by altering the HMM training algorithm. Because some word types have only one possible POS tag, some trigrams are unambiguously tagged. They use such trigrams to initialize transition distributions. In addition, they also update transition probabilities first. They are fixed after they have converged, and then the emission parameters estimated. Toutanova and Johnson [126] assume the POS tag set is known although the tagging lexicon do not have to be complete. To model missing lexicon entries, they represent POS tag ambiguity class for each word type as a latent variable in a directed Bayesian graphical model. The ambiguity class variable generates orthographic morphological features of word types as well as token POS tags[1] Subsequently observed data – word types, word tokens, and their context – are generated. In a related approach, Hasan and Ng [56] also address the problem of incomplete tagging lexicon entries. Since tagging dictionaries are typically constructed using a labeled corpus, they directly use a small amount of POS tagged data to improve their Bayesian HMM tagger. They modify their sampling-based inference to use empirical distributions derived from the labeled corpus and back-off to standard Gibbs sampling for unseen words or contexts.

Rather than handling missing entries in partial lexicon, another body work focuses on integrating weak sources of supervision. Under the complete tagging lexicon setting, parallel multilingual corpora are exploited to improve tagging performance [90, 115, 118, 119]. Using a Bayesian generative framework, POS tags that generate aligned words (in different languages) are coupled to constrain the search space. On the other hand, Haghighi and Klein [49] do not require a tagging dictionary is available, but assume a few prototypical words for each POS tag are given. They first compute distributional similarity between each prototype and the rest of the words. Then for each word, a set of prototypes exceeding a threshold are selected and encoded as features in a log-linear model.

Contrary to all of above, just like some recent work [8, 43, 59], we do not use any

[1] Tags are generated in a Latent Dirichlet Allocation-like fashion. Dependencies between adjacent tags are introduced by generating the surrounding word context for each word token.

partial tagging lexicon or any external corpus.

2.2.2 Sparsity Constraints

Recent work has made significant progress on addressing challenges of unsupervised POS tagging. Our work is closely related to recent approaches that incorporate the sparsity constraint to tame parameter estimation in the absence of any tagging lexicon. This line of work has been motivated by empirical findings that the standard EM-learned unsupervised HMM does not exhibit sufficient word-tag sparsity [59]. The extent to which sparsity is enforced varies greatly across existing methods.

On one end of the spectrum are approaches that encode sparsity as a soft constraint. For instance, by employing a sparse prior for the emission distribution parameters, Johnson [59] encourages the HMM to put most of the probability mass on few words. The prior encodes the preference for few word types per POS tag but does not capture the intuition that each word type typically takes few POS tags (although corpus statistics from WSJ also reveal the former [59, Figure 3]). Moreover, a sparse prior merely biases towards sparsity, but does not guarantee that learned distributions are actually so.

In view of these observations, Graça et al. [43] develop a more forceful approach for encoding sparsity. They constrain the posterior probability of each word has mass over only a small number of POS tags, which more closely reflects a compact lexicon.. Concretely, they propose a posterior regularization method that constrains posteriors to have a small number of expected tag assignments. The optimization algorithm is similar to the standard EM algorithm with the difference that the posterior used in the maximization step satisfies one linear expectation constraint per word type. The posterior does not have a closed form, but it can be obtained by solving its dual with a gradient-based method. Thus, the learning procedure now becomes more prohibitive compared to the EM algorithm for the original HMM. Apart from seeking compact a tagging lexicon, Ravi and Knight [103] also minimize the model grammar which is defined to be the number of unique tag bigrams. Starting with a full or partial tagging lexicon, they iteratively alternate between using the dictionary and the minimal grammar to constrain the EM algorithm. To obtain the minimal tagging grammar, they use integer linear programming (ILP) to find the smallest set of tag bigrams that forms a valid tag sequence with respect to the dictionary constraints.

In contrast, our method imposes a strict one-tag-per-word constraint directly into the structure of the model. Although less general than posterior regularization, our specialization of the Bayesian HMM is empirically effective but yet inference is simple. Because we limit each word type to take exactly one tag, we can sample token tags belonging to the same word type simultaneously. When more model components are added, we can still conveniently retain the inference method (Gibbs sampling).

There are a number of work that also assign a single POS tag to each word type, which essentially are clustering approaches. Schutze [112] and Lamar et al. [72] represent word types as contextual feature vectors and perform dimension reduction of these vectors using singular value decomposition (SVD). Then they cluster the low-dimensional vectors using a variant of the k-means algorithm. After that, they repeat the process again, now using the cluster identity of each token (instead of the word themselves) in the contextual vectors. In contrast, our model captures contextual information with the transitional distribution of a HMM, where the POS tag of a token directly influences that of its neighbors.

Our basic model is similar to the HMM-based clustering approaches of Brown et al. [13] and Clark [20]. The primary difference between our work lies in the inference procedure. Brown et al. [13] develop an agglomerative clustering algorithm that organizes word types in the form of a tree. The algorithm initially assigns each word type to a singleton cluster then greedily merges two clusters based an information theoretic criteria. On the other hand, Clark [20] initializes cluster assignments based on frequency counts, then greedily searches for the cluster assignments that maximizes the data likelihood. This is performed by heuristically moving a word to a cluster that maximizes the likelihood. The main difference is that our one-tag HMM is Bayesian, and we employ Markov chain Monte Carlo (MCMC) to induce POS tags. Moreover, using such generic probabilistic search algorithms also allow us to introduce model enhancements in flexible ways without having to re-design the inference.

2.2.3 Unsupervised Feature-based Models

Another thread of relevant research has explored the use of features in unsupervised POS induction [8, 114]. These methods demonstrated the benefits of incorporating linguistic features using a log-linear parameterization, but requires elaborate machinery for training. Smith and Eisner [114] introduce a training for training unsupervised conditional random field model (CRF), which are structurally similar to HMM with the crucial distinction that former is globally normalized. Instead of maximizing the marginal likelihood that requires enumerating over all possible word sequences, they modify the training criterion so that training becomes feasible. They generate negative data from observed word sequences by performing minor word order perturbations and optimize model parameters to distinguish between them. The accuracy of the tagger, however, relies on prior knowledge to generate effective negative neighborhoods.

Berg-Kirkpatrick et al. [8] develop a less demanding method that only applies log-linear parameterization to the emission distribution of a HMM. Because the log-linear model needs only be locally normalized, training does not require all possible word sequences to be enumerated. Although the model can be learned using the EM algorithm, the maximization step still requires gradient-based optimization as for Smith and Eisner's model, instead of a single pass when the multinomial distribution is used[2]. In fact, now that the maximization does not have a closed form, they also directly optimize for the log-linear parameters using a gradient-based solver.

In our work, we use a simple naïve-Bayes approach which assumes features are generated independently. Hence, we can perform inference using the same Gibbs sampling procedure as for our basic type-level Bayesian HMM. Even when features are overlapping, our experiments demonstrate that it yields substantial performance gains, without the associated training complexity.

VARIABLES

$\boldsymbol{W}$: Word types $(W_1, \ldots, W_n)$ (obs)

$\boldsymbol{T}$: Tag assigns $(T_1, \ldots, T_n)$

$\boldsymbol{w}$: Token word seqs (obs)

$\boldsymbol{t}$: Token tag assigns (det by $\boldsymbol{T}$)

PARAMETERS

ψ: Lexicon parameters

θ : Token word emission parameters

ϕ : Token tag transition parameters

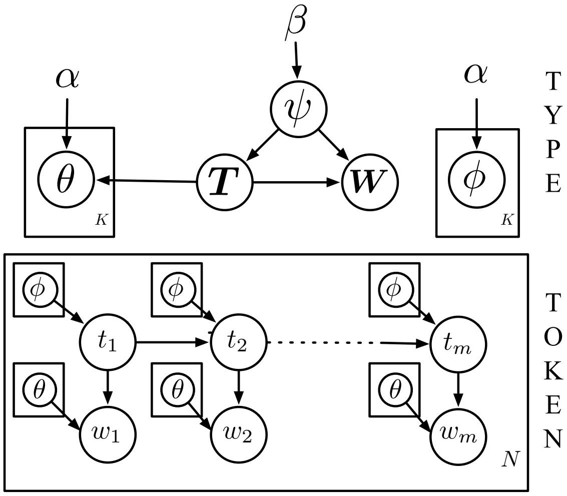


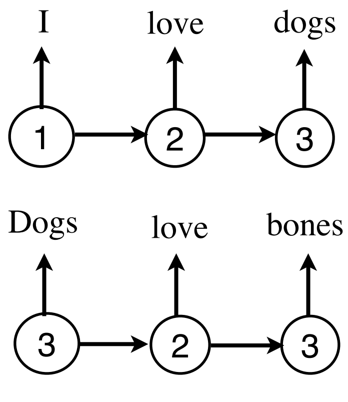
Figure 2-2: Graphical depiction of our model and summary of latent variables and parameters. The type-level tag assignments $\boldsymbol{T}$ generate features associated with word types $\boldsymbol{W}$. The tag assignments constrain the HMM emission parameters θ. The tokens $\boldsymbol{w}$ are generated by token-level tags $\boldsymbol{t}$ from an HMM parameterized by the lexicon structure. The hyperparameters α and β represent the concentration parameters of the token- and type-level components of the model respectively. They are set to fixed constants.

Tag	Word
1	I
2	love
3	dogs
3	Dogs
3	bones

(a)

P(w|t)	1	2	3
I	1.0		
love		1.0	
dogs			0.2
Dogs			0.4
bones			0.4

(b)



(c)

Figure 2-3: Example of structures generated by our type-level HMM: (a) shows a tagging lexicon that has the one-tag-per-word property, (b) shows an emission probability table for a HMM, and (c) shows a token-level corpus generated by a HMM that respects the lexicon. Note that a word cannot be generated by two different tags. POS induction amounts to inferring the latent lexicon that are likely to generate the observed sentences.

2.3 Model

We consider the unsupervised POS induction problem without the use of a tagging dictionary. A graphical depiction of our model can be found in Figure 2-2 and an example of structures generated is shown in Figure 2-3.

2.3.1 Generative Story

The model starts by generating a tagging lexicon, parameters of a hidden Markov Model (HMM), and then a token-level corpus:

1. **Tagging Lexicon**: Draw a sequence of n tags $\boldsymbol{T} = (T_1, \ldots, T_n)$. Conditioned on $\boldsymbol{T}$, features of word types $\boldsymbol{W} = (W_1, \ldots, W_n)$ are drawn. We refer to $(\boldsymbol{T}, \boldsymbol{W})$ as the lexicon of a language, and this creates a lexicon that has one tag per word type. Implicitly drawn prior to the lexicon are ψ, the parameters for their generation; ψ depends on a single hyperparameter β_t. The variant that generates feature-value pairs (f, v) first draws multinomial distributions from hyperparameter β_f. There are a total of three variants which we shall detail later.

2. **HMM Parameters**: Once the lexicon has been drawn, the model proceeds similarly to the standard token-level HMM: Emission parameters θ are generated conditioned on tag assignments $\boldsymbol{T}$. We also draw transition parameters ϕ. The hyperparameters for the transition and the emission distributions are α_t and α_e respectively.

3. **Token-level Corpus**: Once HMM parameters (θ, ϕ) are drawn, a token-level tag and word sequence, (t, w), is generated in the standard HMM fashion: a tag sequence t is generated from ϕ. The corresponding token words w are drawn conditioned on t and θ.[3]

[2] The transition distribution still has a closed form like the regular HMM.

[3] Note that t and w denote tag and word sequences respectively, rather than individual tokens or tags.

Our full generative model is given by:

$$
\begin{aligned}
&P(\boldsymbol{T}, \boldsymbol{W}, \theta, \psi, \phi, \boldsymbol{t}, \boldsymbol{w} | \alpha_t, \alpha_e, \beta_t, \beta_f) = \\
&\quad P(\boldsymbol{T}, \boldsymbol{W}, \psi | \beta_t, \beta_f) && \text{[Lexicon]} \\
&\quad P(\phi, \theta | \boldsymbol{T}, \alpha_t, \alpha_e, \beta_t) && \text{[Parameter]} \\
&\quad P(\boldsymbol{w}, \boldsymbol{t} | \phi, \theta) && \text{[Token]}
\end{aligned}
$$

We refer to the components on the right hand side as the lexicon, parameter, and token component respectively. Since the parameter and token components will remain fixed throughout experiments, we briefly describe each. Table 2.2 summarizes the notation used in this chapter.

2.3.2 Parameter Component

As in the standard Bayesian HMM [40], all distributions are independently drawn from symmetric Dirichlet distributions:

$$P(\phi, \theta | \boldsymbol{T}, \alpha_t, \alpha_e) = \prod_{t=1}^{K} \left(P(\phi_t | \alpha_t) P(\theta_t | \boldsymbol{T}, \alpha_t, \alpha_e) \right)$$

The transition distribution ϕ_t for each tag t is drawn according to $\textsc{Dirichlet}(\alpha_t, K)$, where α_t is the transition distribution hyperparameter. Similarly, the emission distribution θ_{t_j} for tag t_j is drawn from a dirichlet distribution with hyperparameter α_e. In total there are $O(K^2)$ parameters associated with the transition parameters.

In contrast to the Bayesian HMM, θ_t is not drawn from a distribution which has support for each of the n word types. Instead, we condition on the type-level tag assignments $\boldsymbol{T}$. Specifically, let $S_t = \{i | T_i = t\}$ denote the indices of the word types which have been assigned tag t according to the tag assignments $\boldsymbol{T}$. Then θ_t is drawn from $\textsc{Dirichlet}(\alpha_t, S_t)$, a symmetric Dirichlet which only places mass on word types indicated by S_t. This ensures that each word will only be assigned a single tag at inference time (see Section 2.4).

Notation used in the type-level component

K – The size of tag set, i.e. the number of latent states.

$\boldsymbol{W}$ – The sequence of word types in the lexicon

$\boldsymbol{T}$ – The sequence of tag assignments in the lexicon

W_i – The i^{th} word type

$T_i \in [1, K]$ – The i^{th} tag which takes an integral value from 1 to K.

(f, v) – A feature-value pair for a word type

ψ – The distribution for generating the lexicon. (There are different variants which we shall detail later).

β_t – The hyperparameter of the prior on the distribution that generates type tags

β_f – The hyperparameter of the prior on the distribution that generates features

β – All type-level hyperparameters, i.e. (β_t, β_f)

Notation used in the token-level component

$\boldsymbol{w}$ – The token-level corpus

$\boldsymbol{t}$ – The corresponding tags for each token in the corpus

w_j – The j^{th} token sequence in the corpus

t_j – The j^{th} tag sequence in the corpus

ϕ_j – The transition distribution (over tags) conditioned on the j^{th} tag

θ_{t_j} – The emission distribution (over words) conditioned on the j^{th} tag

α_t – The hyperparameter of the prior on the transition distribution

α_e – The hyperparameter of the prior on the emission distribution

α – All token-level hyperparameters, i.e. (α_e, α_t)

Table 2.2: Summary of notation used for our type-level tagging model. In general, capital random variables are types and lowercase are token-level.

Note that while the standard HMM, has $O(Kn)$ emission parameters, our model has $O(n)$ effective parameters.[4]

2.3.3 Token Component

Once HMM parameters (ϕ, θ) have been drawn, the HMM generates a token-level corpus $\boldsymbol{w}$ in the standard way:

$$P(\boldsymbol{w}, \boldsymbol{t}|\phi, \theta) = \prod_{(w,t)\in(\boldsymbol{w},\boldsymbol{t})} \left(\prod_j P(t_j|\phi_{t_{j-1}}) P(w_j|t_j, \theta_{t_j}) \right)$$

Note that in our model, conditioned on $\boldsymbol{T}$, there is precisely one $\boldsymbol{t}$ which has non-zero probability for the token component, since for each word, exactly one θ_t has support.

2.3.4 Lexicon Component

We present several variations for the lexical component $P(\boldsymbol{T}, \boldsymbol{W}|\psi)$, each injecting more linguistic knowledge and information into the generation of lexicon tag structure as well as word type information. Beginning with a lexicon model that only encodes one tag per word type, we shall describe a series of lexicon models of increasing sophistication that eventually leads to one that captures orthographic features of each word in the lexicon.

Uniform Tag Prior (1TW) Our initial lexicon component will be uniform over possible tag assignments as well as word types. Its only purpose is to explore how well we can induce POS tags using only the one-tag-per-word constraint. Specifically, the lexicon is generated as:

$$P(\boldsymbol{T}, \boldsymbol{W}|\psi) = P(\boldsymbol{T})P(\boldsymbol{W}|\boldsymbol{T}) = \prod_{i=1}^{n} P(T_i)P(\boldsymbol{W}|\boldsymbol{T})$$

where both $P(T_i)$ and $P(\boldsymbol{W}|\boldsymbol{T})$ gives uniform probability. This model is equivalent to the standard HMM except that it enforces the one-word-per-tag constraint.

[4] This follows since each θ_t has $S_t - 1$ parameters and $\sum_t S_t = n$.

Learned Tag Prior (PRIOR) We next assume there exists a single prior distribution ψ over tag assignments drawn from $\textsc{Dirichlet}(\beta_t, K)$. This alters generation of $\boldsymbol{T}$ as follows:

$$P(\boldsymbol{T}|\psi) = \prod_{i=1}^{n} P(T_i|\psi)$$

Note that this distribution captures the frequency of a tag across word types, as opposed to tokens. The $P(T|\psi)$ distribution, in English for instance, should have very low mass for the DT (determiner) tag, since determiners are a very small portion of the vocabulary. In contrast, NNP (proper nouns) form a large portion of vocabulary. Note that these observations are not modeled by the standard HMM, which instead can model token-level frequency.

Word Type Features (FEATS): Past unsupervised POS work have derived benefits from features on word types, such as suffix and capitalization features [8, 56]. Past work however, has typically associated these features with token occurrences, typically in an HMM. In our model, we associate these features at the type-level in the lexicon. Here, we consider suffix features, capitalization features, punctuation, and digit features. While possible to utilize the feature-based log-linear approach described in Berg-Kirkpatrick et al. [8], we adopt a simpler naïve Bayes strategy, where all features are emitted independently. Specifically, we assume each word type W consists of feature-value pairs (f, v). For each feature type f and tag t, a multinomial ψ_{tf} is drawn from a symmetric Dirichlet distribution with concentration parameter β_f. The $P(\boldsymbol{W}|\boldsymbol{T}, \psi)$ term in the lexicon component now decomposes as:

$$P(\boldsymbol{W}|\boldsymbol{T}, \psi) = \prod_{i=1}^{n} P(W_i|T_i, \psi) = \prod_{i=1}^{n} \left(\prod_{(f,v) \in W_i} P(v|\psi_{T_i f}) \right) \tag{2.1}$$

2.4 Inference

Given an unlabeled corpus consisting of tokens $\boldsymbol{w}$ and word types $\boldsymbol{W}$, our goal is to recover the latent variables of our model, i.e. tokens tags $\boldsymbol{t}$ and type tags $\boldsymbol{T}$. Specifically, we cast the POS induction task as a Bayesian inference problem where the objective is to draw a sample[5] for $(\boldsymbol{t}, \boldsymbol{T})$ from the collapsed[6] posterior distribution:

$$\begin{aligned} P(\boldsymbol{T}, \boldsymbol{t} | \boldsymbol{W}, \boldsymbol{w}, \alpha_t, \alpha_e, \beta_t, \beta_f) &\propto P(\boldsymbol{T}, \boldsymbol{t}, \boldsymbol{W}, \boldsymbol{w} | \alpha_t, \alpha_e, \beta_t, \beta_f) \\ &= \int P(\boldsymbol{T}, \boldsymbol{t}, \boldsymbol{W}, \boldsymbol{w}, \psi, \theta, \phi, \boldsymbol{w} | \alpha_t, \alpha_e, \beta_t, \beta_f) d\psi d\theta d\phi. \end{aligned} \tag{2.2}$$

Although the likelihood (equation 2.2) can be analytically calculated, we cannot sample the high-dimensional tag variables directly. For this, we employ Markov chain Monte Carlo (MCMC) inference. Starting with a random initial assignment for random variables of interest $\mathbf{h}_{(0)}$, MCMC works by repeatedly performing a random walk according to an appropriate probability distribution to a new assignment $\mathbf{h}_{(m)}$ conditioning only on the previous assignment $\mathbf{h}_{(m-1)}$. As the number of moves M gets large, $\mathbf{h}_{(M)}$ effectively comes from a distribution that approaches the desired posterior, independent from the initial value. To obtain another sample, one simply repeats the process. Here, we adopt Gibbs sampling which is a specialized form of MCMC technique that only samples a subset of random variables $\mathbf{h}_s$ from the conditional distribution[7] $P(\mathbf{h}_s | \mathbf{h}_{-s})$, where $\mathbf{h}_{-s}$ denotes the rest of the random variables.

In the context of our type-level HMM, $\mathbf{h}_s = (T_i, \boldsymbol{t}^{(i)})$. In other words, at each iteration, we consider the i^{th} word type and sample its type tag T_i and the set of associated token-level tags $\boldsymbol{t}^{(i)}$. Note that given type tag assignments $\boldsymbol{T}$, there is only one setting of token-level tags $\boldsymbol{t}$ which has mass in the above posterior. In other words, all tags in $\boldsymbol{t}^{(i)}$ must all take the value T_i. Thus in the context of Gibbs sampling, if we want to block sample T_i with $\boldsymbol{t}^{(i)}$, we only need sample values $\{1, \ldots, K\}$ for T_i and

[5] We evaluate the performance of the model by averaging the score of a number of samples independently drawn from the posterior. We discuss in Section 2.5 how each sample is scored.

[6] Multinomial parameters (ψ, θ, ϕ) associated with our components are unobserved but are conveniently analytically integrated out because conjugate Dirichlet priors are used

[7] Because all we need is to respect the relative odds of each hypothesis, the conditional distribution is computed up to proportion in practice to speed up computation.

consider this setting of $\boldsymbol{t}^{(i)}$. Thus, the state of our tagger is effectively governed by n distinct type-level random variables which we stochastically change one at a time[8]. Algorithm 1 shows an overview of our inference algorithm.

Algorithm 1: Gibbs sampling inference for our type-level POS model

```
Data: Corpus consisting of $n$ word types $\boldsymbol{W}$ and their tokens $\boldsymbol{w}$
Input: Hyperparameters $\alpha$,$\beta$. Number of passes $P$. Tagset of size $K$
Output: One-tag-per-word tagging lexicon $\boldsymbol{T}$
$\boldsymbol{T} \leftarrow$ initialize-lexicon
for $p \leftarrow 1$ to $P$ do
    for $i \in$ random-permute($1, \dots, n$) do
        for $t \in \{1, \dots, K\}$ do
            $p_i(t) \leftarrow P((T_i, \boldsymbol{t}^{(i)}) = t | \boldsymbol{T}_{-i}, \boldsymbol{W}, \boldsymbol{t}^{(-i)}, \boldsymbol{w}, \alpha, \beta)$ // Section 2.4.1
        end
        $T_i, \boldsymbol{t}^{(i)} \leftarrow$sample-tag($p_i$)
    end
end
```

2.4.1 Sampling Equations

The equation for sampling a single type-level assignment T_i is given by:

$$P(T_i, \boldsymbol{t}^{(i)} | \boldsymbol{T}_{-i}, \boldsymbol{W}, \boldsymbol{t}^{(-i)}, \boldsymbol{w}, \alpha, \beta) = P(T_i | \boldsymbol{W}, \boldsymbol{T}_{-i}, \beta) \cdot P(\boldsymbol{t}^{(i)} | \boldsymbol{t}^{(-i)}, \boldsymbol{w}, T_i, \alpha),$$

where $\boldsymbol{T}_{-i}$ denotes all type-level tag assignment except T_i and $\boldsymbol{t}^{(-i)}$ denotes all token-level tags except $\boldsymbol{t}^{(i)}$. The left-hand-side decomposes into two terms according to the chain rule. The terms on the right-hand-side denote the type-level and token-level probability terms respectively.

[8] This is different from the type-based MCMC of Liang et al. [78] which simultaneously samples an appropriately chosen subset of variables belonging to the same *type* but does not restrict all of them to take the same value. Also, their notion of type is different — unobserved variables belong to the same type if observing any one of them give the same data count statistics.

Type-level Component The type-level posterior term can be computed according to:

$$P(T_i|\boldsymbol{W}, \boldsymbol{T}_{-i}, \beta) = \frac{P(T_i, W_i|\boldsymbol{W}_{-i}, \boldsymbol{T}_{-i}, \beta)}{P(W_i|\boldsymbol{W}_{-i}, \boldsymbol{T}_{-i}, \beta)} \propto P(T_i|\boldsymbol{T}_{-i}, \beta_t) \cdot P(W_i|\boldsymbol{W}_{-i}, \boldsymbol{T}, \beta_f).$$

The equality follows from the definition of conditional probability and the numerator subsequently factorizes according to the chain rule. The denominator does not have to computed since it is a common factor for all values of T_i which does change the sampling outcome. The first term on the right-hand-side then simplifies due to Markov assumptions in our model.

All of the probabilities on the right-hand-side are posterior predictive distributions that can be computed analytically given data counts because of the Dirichlet-multinomial conjugacy. The posterior probability for a tag can be calculated as:

$$P(T_i|\boldsymbol{T}_{-i}, \beta_t) = \int_{\psi} P(T_i|\boldsymbol{T}_{-i}, \psi) p(\psi|\beta_t) d\psi = \frac{\beta_t + n(T_i)}{\beta_t \cdot K + |\boldsymbol{W}_{-i}|},$$

where $n(T_i)$ denotes the number of word types in $\boldsymbol{W}_{-i}$ that are assigned tag T_i. Hyperparameter β_t plays the role of pseudo-counts in what also commonly known as "add-alpha" smoothing.

The posterior probability for a word factorizes as:

$$P(W_i|\boldsymbol{W}_{-i}, \boldsymbol{T}, \beta_f) = \prod_{(f,v) \in W_i} P(v|f, \boldsymbol{W}_{-i}, \boldsymbol{T}, \beta_f) = \prod_{(f,v) \in W_i} \frac{\beta_f + n(f, v, T_i)}{\beta_f \cdot |V_f| + n(f, T_i)},$$

where β_f is the hyperparameter for the Dirichlet prior, $|V_f|$ is the number of distinct values for feature f, $n(v, f, T_i)$ denotes the number of word types that have tag T_i and feature-value pair (f, v), and $n(f, T_i)$ denotes the number of word types that have tag T_i and feature f. The first equality is due to feature independence assumptions of FEATS model (Section 2.3.4), and the second equality is due to Dirichlet-multinomial conjugacy as before. Similarly, all counts are obtained from word types in $\boldsymbol{W}_{-i}$.

Token Component The token-level term is similar to the standard sampling equations for the token-level Bayesian HMM [36, 40, 59], with the crucial exception that we are tying the values of state variables belong to the same word type and sampling them together. Note that because the transition and the emission parameters are unobserved, the the state variables under consideration becomes dependent[9].

Specifically, the posterior probability for token-level tags $\boldsymbol{t}^{(i)}$ for the i^{th} word type decomposes into emission and transition parts:

$$P(\boldsymbol{t}^{(i)}|T_i, \boldsymbol{t}^{(-i)}, \boldsymbol{w}, \alpha) = \frac{P(\boldsymbol{w}^{(i)}, \boldsymbol{t}^{(i)}|\boldsymbol{t}^{(-i)}, \boldsymbol{w}^{(-i)}, T_i, \alpha)}{P(\boldsymbol{w}^{(i)}|\boldsymbol{t}^{(-i)}, \boldsymbol{w}^{(-i)}, T_i, \alpha)} \tag{2.3}$$

$$\propto \prod_{j=1}^{|\boldsymbol{t}^{(i)}|} \Big\{ P(t_j^{(i)}|\{t_k^{(i)}\}_{k=1}^{j-1}, \{w_k^{(i)}\}_{k=1}^{j-1}, \boldsymbol{t}^{(-i)}, \boldsymbol{w}^{(-i)}, T_i, \alpha_t) \cdot P(w_j^{(i)}|\{t_k^{(i)}\}_{k=1}^{j}, \{w_k^{(i)}\}_{k=1}^{j-1}, \boldsymbol{t}^{(-i)}, \boldsymbol{w}^{(-i)}, T_i, \alpha_e) \Big\}. \tag{2.4}$$

To avoid enumerating over the whole corpus for each value of $\boldsymbol{t}^{(i)} = T_i$ when computing the sampling equation, we make the approximation that denominator on equation 2.3 is the same for all values of T_i, and thus cancels out. This is equivalent to assuming that the total pseudo-counts used for smoothing, i.e. $\alpha_e|W_{T_i}|$, is the same for all values of T_i.

Each of the $|\boldsymbol{t}^{(i)}|$ factors in equation 2.4 is a product of two expressions, arising from observing the j^{th} tag and word tokens respectively. Note that this also conditions on the $(j-1)$ word-tag pairs that are already generated. (For the j^{th} word token, this also conditions on the j^{th} tag.) Again, because of the Dirichlet-multinomial conjugacy, each predictive posterior probability has a closed form. Note that for the j^{th} emission posterior, we have observed $(j-1)$ word-tags pairs of (T_i, W_i), in addition to counts from the rest of the data belonging to other word types. And thus the j^{th} emission posterior probability is given by:

$$P(w_j^{(i)}|\{t_k^{(i)}\}_{k=1}^{j}, \{w_k^{(i)}\}_{k=1}^{j-1}, \boldsymbol{t}^{(-i)}, \boldsymbol{w}^{(-i)}, T_i, \alpha_e) = \frac{\alpha_e + (j-1)}{\alpha_e \cdot |W_{T_i}| + n_{-i}(T_i) + (j-1)},$$

[9] In our conference paper [74], we simplify computation of the sampling equation by assuming that these latent variables are independent.

where $n_{-i}(T_i)$ is the number of tokens belonging to other word types (i.e. $\boldsymbol{t}^{(-i)}$) that are assigned tag T_i, and $|W_{T_i}|$ is the number of word types in the lexicon that are tagged with T_i.[10] Note that as the chain rule is applied, the word tokens for which we previously compute the posterior probabilities now becomes part of the observed data. Accumulating all $|\boldsymbol{t}^{(i)}| \doteq n(i)$ token occurrences, the emission part of equation 2.4 becomes:

$$\frac{\alpha_e^{[n(i)]}}{(\alpha_e \cdot |W_{T_i}| + n_{-i}(T_i))^{[n(i)]}}, \tag{2.5}$$

where the expression $a^{[k]} \doteq a(a+1)\cdots(a+k-1)$ is the ascending factorial.

The transition (second) factor in equation 2.4 is computed similarly. Consider a single token tag $t_j^{(i)} = T_i$ which has neighboring tags t_l and t_r on the left and right respectively. Its posterior probability is given by [40, 59][11]:

$$\begin{aligned} P(t_j^{(i)}|\{t_k^{(i)}\}_{k=1}^{j-1},&\{w_k^{(i)}\}_{k=1}^{j-1}, \boldsymbol{t}^{(-i)}, \boldsymbol{w}^{(-i)}, T_i, \alpha_t) \\ &\propto P(t_j^{(i)}|t_l, \{t_k^{(i)}\}_{k=1}^{j-1}, \{w_k^{(i)}\}_{k=1}^{j-1}, \boldsymbol{t}^{(-i)} \backslash \{t_l, t_r\}, \boldsymbol{w}^{(-i)}, T_i, \alpha_t) \\ &\cdot P(t_r|t_j^{(i)}, t_l, \{t_k^{(i)}\}_{k=1}^{j-1}, \{w_k^{(i)}\}_{k=1}^{j-1}, \boldsymbol{t}^{(-i)} \backslash \{t_l, t_r\}, \boldsymbol{w}^{(-i)}, T_i, \alpha_t) \\ &= \left(\frac{\alpha_t + n(t_l, T_i)}{\alpha_t \cdot K(t_l) + n(t_l)}\right) \cdot \left(\frac{\alpha_t + n(T_i, t_r) + I(t_l = T_i = t_r)}{\alpha_t \cdot K(T_i) + n(t_r) + I(t_l = Ti)}\right), \end{aligned} \tag{2.6}$$

where $K(\cdot)$ denotes the number of possible transitions (K for the start state and $K+1$ for a regular POS tag since the next state can be a stop state). Expressions $n(t_l, T_i)$ and $n(T_i, t_r)$ denote the number of (t_l, T_i) and (T_i, t_r) tags transitions obtained in the data observed thus far, i.e. $\{t_k^{(i)}\}_{k=1}^{j-1} \cup \boldsymbol{t}^{(-i)}$. The notation $I(\cdot)$ is an indicator variable that handles the case that observing $t_j^{(i)}$ influences the posterior estimate of t_r (now that the former is part of the evidence that is conditioned on). As in the case for emission, accumulating the transition posterior probabilities for $\boldsymbol{t}^{(i)}$ give rise to an expression involving ascending factorials. Putting everything together, Figure 2-4 shows how the token-level sampling equation 2.4 is calculated analytically.

[10] Note that we have generated the lexicon by the time we are generating the tokens, and so $|W_{T_i}|$ includes the current word type W_i.

[11] Note that we have factored out the emission probability.

$$\left(\frac{\alpha_e^{[n(i)]}}{(\alpha_e \cdot |W_{T_i}| + n_{-i}(T_i))^{[n(i)]}}\right) \cdot \left(\prod_{t=0}^{K}\prod_{t'=0}^{K} \frac{\alpha_t^{[n(t,t')]}}{(\alpha_t \cdot [K + I(t \neq 0)] + n_{-i}(t))^{[n(t,t')]}}\right)$$

T_i	–	Tag for the i^{th} word type
$\boldsymbol{w}^{(i)}$	–	Word tokens of the i^{th} word type in the corpus
$\boldsymbol{t}^{(i)}$	–	Corresponding tags for word tokens $\boldsymbol{w}^{(i)}$
$\boldsymbol{t}^{(-i)}$	–	Tags in token-level corpus, excluding $\boldsymbol{t}^{(i)}$
α_e / α_t	–	Hyperparameter of prior on emission / transition
$\|W_{T_i}\|$	–	Number of word types in the lexicon that are tagged with tag T_i
$n_{-i}(t)$	–	Number of out-going tag transitions from tag t in $\boldsymbol{t}^{(-i)}$
$n(i)$	–	Number of word tokens of the i^{th} word type, i.e. $\|\boldsymbol{w}^{(i)}\|$
$n(t)$	–	Number of tags in $\boldsymbol{t}^{(i)}$ that are assigned t
$n(t,t')$	–	Number of t to t' tag transitions introduced by $\boldsymbol{t}^{(i)}$
$t = 0$	–	Tag t is the start state of the HMM
$t' = 0$	–	Tag t' is the stop state of the HMM
$I(t \neq 0)$	–	Indicator function. Needed since we assume no start-stop transition
$\alpha^{[k]}$	–	Ascending factorial, i.e. $\alpha(\alpha+1)\ldots(\alpha+k-1)$. Note $\alpha^{[0]} \doteq 1$

Figure 2-4: Closed-form for calculating the token-level sampling equation 2.4 up to proportion.

Note that each round of sampling T_i variables takes time proportional to the size of the corpus, as with the standard token-level HMM. A crucial difference is that the number of parameters is greatly reduced as is the number of variables that are sampled during each iteration. In contrast to results reported in Johnson [59], we found that the performance of our Gibbs sampler on the basic 1TW model stabilized very quickly after about 10 full iterations of sampling (see Figure 2-5 for a depiction).

2.4.2 Sampling Hyperparameters

To systematically infer the hyperparameter settings in our model, we treat the Dirichlet priors as random variables and sample their values. Because there is no known conjugate prior for the Dirichlet distribution, we adopt the commonly used vague Gamma(10,0.1) prior [80]. More concretely, the generative story now begins with drawing parameter values of Dirichlet distributions from the Gamma distribution independently, then the rest of the variables are generated as before. To perform posterior inference for the new model, we retain the Gibbs sampler as our MCMC algorithm of choice. Here, the only modification that is needed is to consider the Dirichlet parameters as block of random variables which we sample every n passes over the whole corpus. To sample the block, we randomly cycle through each of the Dirichlet parameter, for instance α_e, and sample its new value conditoned on the rest of the variables:

$$\begin{aligned} P(\alpha_e|\boldsymbol{W},\boldsymbol{T},\boldsymbol{w},\boldsymbol{t},\alpha_t,\beta_t,\beta_f) &= \frac{P(\alpha_e,\boldsymbol{W},\boldsymbol{T},\boldsymbol{w},\boldsymbol{t}|\alpha_t,\beta_t,\beta_f)}{P(\boldsymbol{W},\boldsymbol{T},\boldsymbol{w},\boldsymbol{t}|\alpha_t,\beta_t,\beta_f)} \\ &\propto P(\boldsymbol{W},\boldsymbol{T},\boldsymbol{w},\boldsymbol{t}|\alpha_e,\alpha_t,\beta_t,\beta_f)\cdot \mathrm{Gamma}(\alpha_e|10.0,0.1), \qquad (2.7) \end{aligned}$$

where the first term is the likelihood of the observed data and tag variables, and the second term is probability of the Dirichlet parameter according to the Gamma prior which is a density function that has support over positive reals.

Slice Sampler To perform one-dimensional sampling from probability density function (pdf) , we employ the slice sampler of Neal [92]. All the sampler needs is to be able evaluate a function $f(x)$ proportional to the pdf of interest $p(x)$. The idea is to introduce an auxiliary variable y and sample from a new function $f(x,y)$ such that the value of x in a (x,y) sample essentially is drawn from $p(x)$. To see this, let us introduce a new dimensional to pdf $p(x)$:

$$p(x,y) = \begin{cases} 1/Z & \text{if} \quad 0 < y < f(x) \\ 0 & \text{otherwise} \end{cases},$$

where $Z = \int f(x)dx$. Suppose we can sample from $p(x,y)$, then the marginal density of x is

$$\int_0^{f(x)} (1/Z)dy = f(x)/Z = p(x),$$

which is our desired pdf in the first place. To sample from $p(x,y)$, one can adopt Gibbs sampling where x and y are sampled in an alternate fashion. Sampling y from $p(y|x)$ simply reduces to drawing a value in the ("vertical") interval $[0, f(x)]$ which is uniformly distributed. To sample x, the algorithm draws a value from the conditional $p(x|y)$ which is uniformed over the set of valid values for $x \in S \doteq \{x : y < f(x)\}$ coined a ("horizontal") *slice*. In other words, drawing a horizontal line at y, the slice contains intervals such that y is in the area under the curve $f(x)$. The crux of the slice sampler is an algorithm for sampling from potentially disjoint intervals in S. Defining $f(x)$ to be equation 2.7, a new parameter value for the Dirichlet distribution is obtained via slice sampling according to Algorithm 2.

2.5 Experiments

We evaluate our approach on 14 languages: English, Arabic, Bulgarian, Chinese, Czech, Danish, Dutch, German, Japanese, Portuguese, Spanish, Swedish, and Turkish. On each language we investigate the contribution of each component of our model. For all languages we do not make use of a tagging dictionary, i.e. the input is a corpus of sentences consisting of only words. Our output is an clustering of word types from which we can tag unannotated corpus for evaluation.

2.5.1 Data Sets

Following the set-up of Johnson [59], we use the whole of the Penn Treebank Wall Street Journal corpus for training and evaluation on English. For other languages, we use the CoNLL-X multilingual dependency parsing shared task corpora [14] which include gold POS tags (used for evaluation). We train and test on the CoNLL-X training set. Statistics for all data sets are shown in Table 2.5.

2.5.2 Setup

Models To assess the marginal utility of each component of the model (see Section 2.3), we incrementally increase its sophistication. Specifically, we evaluate three variants: The first model (1TW) only encodes the one tag per word constraint and is uniform over type-level tag assignments. The second model (+PRIOR) utilizes the in-

Algorithm 2: Slice sampling algorithm for POS model hyperparameter inference

```
Input:
Function $f(x) \propto$ pdf $p(x)$.
Initial value $x_0$.
Number of iterations $T$.
Initial guess for slice width $w$
Output:
A random sample from $p(x)$
for $i \leftarrow 1$ to $T$ do
    // Step 1.  Draw auxiliary variable $y$
    $y \leftarrow$ uniform-draw($[0, f(x_{i-1})]$)
    // Step 2.  Bracket for interval $(x_l, x_r)$ enclosing some slice(s)
    $x_l \leftarrow x_{i-1} - w \cdot$ uniform-draw($[0, 1]$)
    $x_r \leftarrow x_l + w$
    while $f(x_l) > y$ do
        $x_l \leftarrow x_l - w$
    end
    while $f(x_r) > y$ do
        $x_r \leftarrow x_r + w$
    end
    $x_l \leftarrow \max(x_l, 0)$ // Dirichlet parameter cannot be negative
    // Step 3.  Shrink interval until we hit a slice
    repeat
        $x_i \leftarrow$ uniform-draw($[x_l, x_r]$)
        if $x_i < x_{i-1}$ then
            $x_l \leftarrow x_i$
        else
            $x_r \leftarrow x_i$
        end
    until $f(x_i) > y$
end
return $x_i$
```

Feature	Examples					
	AK-47	loving	U.S.-based	1950s	Dr.	Dogs
Contains digit	Yes	No	No	Yes	No	No
Contains punctuation	Yes	No	Yes	No	Yes	No
Initial capital	Yes	No	Yes	No	Yes	Yes
Suffix	∅	ing	-based	s	∅	∅

Table 2.3: Features used in our POS induction model. ∅ denotes a null suffix. The suffixes are obtained with an unsupervised language-independent morphological segmenter [26]. The segmenter can produce erroneous suffixes. For example, it does not give any suffix for the word "Dogs".

dependent prior over type-level tag assignments $P(\boldsymbol{T}|\psi)$. The final model (+FEATS) utilizes the tag prior as well as features (e.g., suffixes and orthographic features), discussed in Section 2.3, for the $P(\boldsymbol{W}|\boldsymbol{T}, \psi)$ component. Table 2.3 lists the complete set of features and gives a few examples. We use Morfessor Catmap [26], an unsupervised language-independent morphological segmenter, [12] to obtain the suffix feature. The segmenter gives zero or more suffixes for each word type. If there is more than one suffix, we concatenate them to form a single suffix feature [18].

Hyperparameters Our model has four Dirichlet concentration hyperparameters: α_e and α_t are the hyperparameter for the token-level HMM emission and transition distributions respectively. β_t and β_f are the hyperparameter for the tag assignment prior and word feature multinomials respectively. We initialize the hyperparameter for the transition distribution prior to 1.0 and the rest of the hyperparameters to 0.01. At every 10 passes of Gibbs sampling, we resample the Hyperparameters. We sample one hyperparameter at a time with 10 iterations of slice sampling.

Iterations In each run, we performed 50 iterations of Gibbs sampling for the type assignment variables $\boldsymbol{W}$ Typically, the performance stabilizes after only 10 iterations. We use the final sample as the output of the run.

[12] In our conference paper [74], we use the rule-based Snowball stemmers available at `http://snowball.tartarus.org/`. However, they do not support some of the additional languages which we use for evaluation.

Tag set As is standard, for all experiments, we set the number of latent model tag states to the size of the annotated fine-grained tag set. The original tag set for the CoNLL-X Dutch data set consists of compounded tags that are used to tag multi-word units (MWUs) resulting in a tag set of over 300 tags. We tokenize MWUs and their fine-grained POS tags; this reduces the tag set size to 12. For Chinese, we use the 15-tag coarse-grained tag size instead of the the 296-tag fine-grained tag set. See Table 2.5 for the tag set size of other languages. With the exception of the Dutch data set, no other processing is performed on the annotated tags.

Other preprocessing Apart from tokenizing MWUs in Dutch and some exceptions which we describe here, we retain the original form of all words in all data sets, i.e. digits, punctuations, and rare words are not collapsed into a smaller set of special tokens. For the Arabic data set, each word its English transliteration. We remove the transliteration so that the morphological segmenter can perform analysis (to give suffixes) as expected. For the Chinese data set, we remove sentences containing erroneously labeled coarse-grained POS tags. These erroneous tag annotations has the form "X|Y", where "X" is a Chinese word comprising of a few characters and "Y" is "Head", "property", or "epistemics".

2.5.3 Evaluation Metrics

We report three commonly used metrics to evaluate tagging performance. The first two are token-based metrics, whereas the last one is type-based. We introduce some notations in Table 2.4 that we used to describe each metric.

(a) *Many-to-one* (m-1): This metric finds a many-to-one mapping that assigns a latent tag to a gold tag such that the token-level accuracy is maximized. For each possible latent-gold tag pair (t, u), the number of tokens $c_{t,u}$ is obtained. Then, each latent tag t is mapped to highest scoring assignment:

$$u(t) = \arg\max_{u} n(t, u).$$

For a corpus with M tokens, this metric is calculated as $(1/M) \cdot \sum_s n(t, u(t))$.

(b) *V-measure* [105] (vm): We also report the v-measure which is a metric for assessing the quality of a clustering. Unsupervised POS induction without a tagging dictionary assigns an arbitrary label to each token and so essentially is equivalent to partitioning tokens into categories. The v-measure is an entropy-based metric that combines two desirable properties of clustering: *homogenity* (h) and *completeness* (c). Homogenity looks at how gold tags are distributed within tokens tagged with same latent tag t. It encourages tag assignments such that tokens tagged with latent tag t belong mostly to a gold tag u. This property is captured using the conditional entropy $H(u|t)$ which is calculated as:

$$H(u|t) = -\sum_{u=1}^{L}\sum_{t=1}^{K} \frac{n(t,u)}{M} \log \frac{n(t,u)}{n(\cdot,t)},$$

To handle tag sets of various sizes, the conditional entropy is normalized by entropy of gold tags $H(u)$:

$$H(u) = -\sum_{u=1}^{L} \frac{n(u,\cdot)}{M} \log \frac{n(u,\cdot)}{M},$$

and the homogenity is defined as:

$$h = 1 - \frac{H(u|t)}{H(u)}.$$

A perfect score of one is obtained when all tokens assigned tag t has the same gold tag. However, homogenity also evaluates to one when each token is assigned its own tag. To handle his degenerated case, completeness evaluates clustering from the other direction, i.e. by looking at how latent tags are distributed within each cluster of gold tags. Completeness is thus defined as:

$$c = 1 - \frac{H(t|u)}{H(t)},$$

$t \in [1, K]$	–	A learned latent tag among K possible ones
$u \in [1, L]$	–	A gold POS tag among L possibe ones
M	–	Total number of tokens in the corpus
$n(t, u)$	–	Number of tokens tagged with t and u
$n(t, \cdot)$	–	Number of tokens tagged with t, i.e. $\sum_u n(t, u)$
$n(\cdot, u)$	–	Number of tokens tagged with u, i.e. $\sum_t n(t, u)$
$H(u)$	–	Entropy of gold tags at the token-level
$H(t)$	–	Entropy of latent tags at the token-level
$H(u\|t)$	–	Conditional entropy of gold tags given latent tags (token-level)
$H(t\|u)$	–	Conditional entropy of latent tags given gold tags (token-level)

Table 2.4: Summary of notations used for calculating POS evaluation metrics

where

$$
\begin{aligned}
H(t|u) &= -\sum_{u=1}^{L} \sum_{t=1}^{K} \frac{n(t, u)}{M} \log \frac{n(t, u)}{c_{c\cdot}} \\
H(t) &= -\sum_{t=1}^{L} \frac{n(\cdot, t)}{M} \log \frac{n(\cdot, t)}{M} \\
c &= 1 - \frac{H(t|u)}{H(t)}
\end{aligned}
$$

Finally, the v-measure is obtained by taking the harmonic mean of homogenity (h) and completeness (c):

$$
\mathrm{vm} = \frac{2 \cdot h \cdot c}{h + c}.
$$

(c) *Type-level accuracy*: We also report word type level accuracy, the fraction of word types assigned their majority tag (where the mapping between model state and tag is determined by greedy one-to-one mapping discussed above).

For each language, we aggregate results by performing five runs with different random initialization of sampling state. We then report the mean value for each performance metrices.

Language	Tags	Types	Tokens	1TW	+PRIOR	+FEATS
English	45	49,206	1,173,766	72.6 / 64.6	73.6 / 65.5	74.6 / 67.1
Arabic	20	12,915	54,379	55.9 / 32.6	59.5 / 36.2	62.1 / 40.3
Bulgarian	54	32,439	190,217	67.0 / 53.3	67.6 / 55.3	73.1 / 60.8
Chinese	15	40,563	337,118	66.1 / 34.2	66.4 / 35.1	66.3 / 35.0
Czech	12	130,208	1,249,408	60.7 / 41.0	61.3 / 42.8	65.1 / 47.8
Danish	25	18,356	94,386	66.3 / 49.6	68.6 / 53.0	72.2 / 57.6
Dutch	12	28,393	203,568	64.7 / 47.7	66.2 / 50.8	69.0 / 54.2
German	54	72,326	699,610	71.6 / 59.5	73.2 / 61.9	74.9 / 64.8
Japanese	80	3231	151,461	79.8 / 79.0	79.8 / 79.1	79.9 / 79.3
Portuguese	22	28,931	206,678	68.6 / 53.6	70.2 / 56.5	75.3 / 62.6
Slovene	29	7,128	28,750	59.5 / 44.2	61.7 / 48.6	64.2 / 51.0
Spanish	47	16,458	89,334	66.7 / 55.1	70.3 / 58.4	74.2 / 62.8
Swedish	41	20,057	191,467	63.8 / 52.7	66.2 / 55.0	68.4 / 57.6
Turkish	30	17,564	57,510	53.7 / 31.7	55.9 / 34.1	59.9 / 39.6

Table 2.5: Multi-lingual POS Induction Results: We report token-level many-to-one accuracy and v-measure (in this order) on a variety of languages under several experimental settings (Section 2.5). Model components cascade, so the row corresponding to +FEATS also includes the PRIOR component (see Section 2.3).

2.6 Results and Analysis

We report token- and type-level accuracy in Table 2.5 and 2.7 for all languages and system settings. Our analysis and comparison focuses primarily on the many-to-one accuracy since it is the most commonly used form of evaluation used in literature. We also report the v-measure which is an entropy-based metric used to evaluate clustering outputs.

2.6.1 Comparison with other unsupervised taggers

For comparison we consider two unsupervised taggers: the HMM with log-linear features of Berg-Kirkpatrick et al. [8] and the posterior regularization HMM of Graça et al. [43]. The system of Berg-Kirkpatrick et al. [8] reports the best unsupervised results for English. We consider two variants of Berg-Kirkpatrick et al. [8]'s richest model: optimized via either EM or LBFGS, as their relative performance depends on the language. Our model outperforms the best results attained by of any their models on three out of five languages on yielding an average absolute difference across languages of 2.8%. On Portuguese, we perform on par with the best model.

Our second point of comparison is with Graça et al. [43], who also incorporate a sparsity constraint, but does via altering the model objective using posterior regularization. We compare with Graça et al. [43] on Portuguese (Graça et al. [43] also report results on English, but on the reduced 17 tag set, which is not comparable to ours). Their best model yields 69.2% many-to-one accuracy, compared to our accuracy of 75.3%. However, our full model takes advantage of word features not present in Graça et al. [43]. Even without features, but still using the tag prior, our many-to-one accuracy 70.2%, still significantly outperforming Graça et al. [43].

Lastly, we also compare against the word-clustering HMMs of Brown et al. [13][13] and Clark [20][14]. Both methods restrict each word type to one class and greedily search for the optimal clustering. Clark incorporates morphological information but Brown et al. do not. Although their models are estimated heuristically, the empirical results are competitive. Brown et al. [13] give the best accuracy on Chinese and Clark [20] is the best model on Czech and Swedish. Averaged over all 14 languages, they give an accuracy of 65.9 and 67.1 respectively. Our 1TW, +PRIOR, and full (+FEATS) models gives 65.5, 67.2, and 69.9 respectively. Our +PRIOR model improves over the their models indicating the effectiveness of the tag prior. We continue to observe gains when orthographic features are added.

Overall, our full model yields better results on 11 out of 14 languages than all systems evaluated above.

2.6.2 Ablation Analysis

We evaluate the impact of incorporating various linguistic features into our model in Table 2.5. A novel element of our model is the ability to capture type-level tag frequencies. For this experiment, we compare our model with the uniform tag assignment prior (1TW) with the learned prior (+PRIOR). Across all languages, +PRIOR

[13] We use the implementation of Liang [77].

[14] The results are obtained from Christodoulopoulos et al. [17].

Language	Brown92	Clark03	BK10 EM	BK10 LBFGS	G10	+FEATS
English	68.5	71.2	68.1	75.5	–	**74.6**
Arabic	54.9	59.8	–	–	–	**62.1**
Bulgarian	67.9	70.4	–	–	–	**73.1**
Chinese	**68.2**	56.7	–	–	–	66.3
Czech	60.5	**65.5**	–	–	–	65.1
Danish	68.3	65.3	66.7	58.0	–	**72.2**
Dutch	58.6	67.9	67.0	64.7	–	**69.0**
German	73.0	73.9	–	–	–	**74.9**
Japanese	79.4	77.4	–	–	–	**79.9**
Portuguese	67.6	69.2	**75.3**	74.8	69.2	**75.3**
Slovene	61.4	63.5	–	–	–	**64.2**
Spanish	71.9	71.9	–	73.2	–	**74.2**
Swedish	64.5	**68.7**	–	–	–	68.4
Turkish	58.2	58.1	–	–	–	**59.9**

Table 2.6: Comparison of our full model (+FEATS) to related methods using the many-to-one accuracy. Feature-based HMM Model [8]: The KM model uses a variety of orthographic features and employs the EM or LBFGS optimization algorithm; Posterior regularization model [43]: The G10 model uses the posterior regularization approach to ensure tag sparsity constraint. Word clustering HMMs: Clark03 [20] utilizes morphological information but Brown92 [13] does not.

consistently outperforms 1TW, reducing error on average by 4.68% on both the many-to-one (m-1) accuracy and v-measure (vm). Similar behavior is observed when adding features. The difference between the featureless model (+PRIOR) and our full model (+FEATS) is 8.26% and 7.19% average error reduction on the m-1 accuracy and vm respectively. Overall, the difference between our most basic model (1TW) and our full model (+FEATS) is 12.5% and 11.5% for m-1 and vm respectively. If we exclude Chinese which is morphologically poor, the error reductions improves to 13.4% and 12.3% respectively. One striking example is the error reduction for Spanish, which reduces error by 22.5% and 17.1% for m-1 and vm respectively.

We observe similar trends when using another measure – type-level accuracy (defined as the fraction of words correctly assigned their majority tag using the many-to-one mapping). Our full model yields 11.5% and 7.19% average error reduction over our basic configuration (1TW) and the prior tag model (PRIOR). Even without using any features, simply modeling type-level tag distribution still gives 4.68% error

Language	1TW	+PRIOR	+FEATS
English	64.6	65.5	67.1
Arabic	32.6	36.2	40.3
Bulgarian	53.3	55.3	60.8
Chinese	34.2	35.1	35.0
Czech	41.0	42.8	47.8
Danish	49.6	53.0	57.6
Dutch	47.7	50.8	54.2
German	59.5	61.9	64.8
Japanese	79.0	79.1	79.3
Portuguese	53.6	56.5	62.6
Slovene	44.2	48.6	51.0
Spanish	55.1	58.4	62.8
Swedish	52.7	55.0	57.6
Turkish	31.7	34.1	39.6

Table 2.7: Type-level results: Each cell report the type-level accuracy computed against the most frequent tag of each word type. The state-to-tag mapping is obtained many-to-one mapping shown in Table 2.5.

reduction over the basic one-tag HMM.

2.6.3 Convergence

In this section, we investigate convergence properties of our tagger. We vary one experimental condition at a time and plot the tagging metrics against the number of Gibbs sampling iterations. For all experiments here, we run 200 iterations of Gibbs sampling and resample hyperparameters every 10 iterations.

Our first experiment investigates the convergence across random restarts. Using our full model on English, we perform three random restarts (each from a different random intialization). We plot all tagging scores (many-to-one, v-measure, and type-level accuracy) and the log posterior likelihood (Equation 2.2) in Figure 2-5 (a) and (b) respectively. We observe that all metrics climb and stablize rapidly. Note that the jump in log posterior likelihood at the 10^{th} iteration is due to hyperparameter resampling. In the next experiment, we run three model variants on English — the basic 1TW model, the +PRIOR model, and the +FEATS model. All components cascade, i.e. the +PRIOR model includes the basic model and the +FEATS model is the full model. Again, we observe that our tagger climb and stablize rapidly. To avoid clutter we only plot the many-to-one accuracy, although we observe similar trends for the other tagging matrics. In the last experiment, we examine the behavior of our full model across all languages we use for evaluation. We observe the same trend that our tagger climbs rapidly and typically stablizes after that.

2.6.4 Robustness Across Random Restarts

Random Initialization We investigate the effect of random restarts on POS induction performance. For each language and model as described in Section 2.5.2, we show in Table 2.8 the standard deviation of each tagging metric across random restarts each with a different random initialization of the latent tagging dictionary. We observe that our models give fairly stable tagging scores. As more model components are added, there is only a modest increase in standard deviations of scores. Specifically, the average standard deviations of the many-to-one accuracy are 1.17, 1.2 and 1.27 for the basic (1TW), intermediate (+PRIOR), and the full (+FEATS) model respectively. The fluctuation of v-measure is less. On average, the difference between the standard deviations of m-1 and vm is 0.24, 0.22, and 0.29 for the basic, intermediate, and the full models respectively. The variation for the type-level is marginally higher — 1.7, 1.2, and 1.5 for the basic, intermediate, and the full model respectively.

Frequency-based initialization Instead of randomly initializing the tagging dictionary, we also experiment with the frequency-based initialization heuristic of Clark [20] — For a HMM with K states, we assign each of the top K most frequent word types to its own state. For each of the remaining word types, we randomly pick its initial state. Using the same experiment settings, we compare tagging scores against those obtained earlier. For each model and evaluation metric, we test if the pairs of scores are different using the paired t-test. We find no significant difference between this initialization and the random initialization, except for the intermediate model (+PRIOR) evaluated under the v-measure. Table 2.9 summarizes the results. This

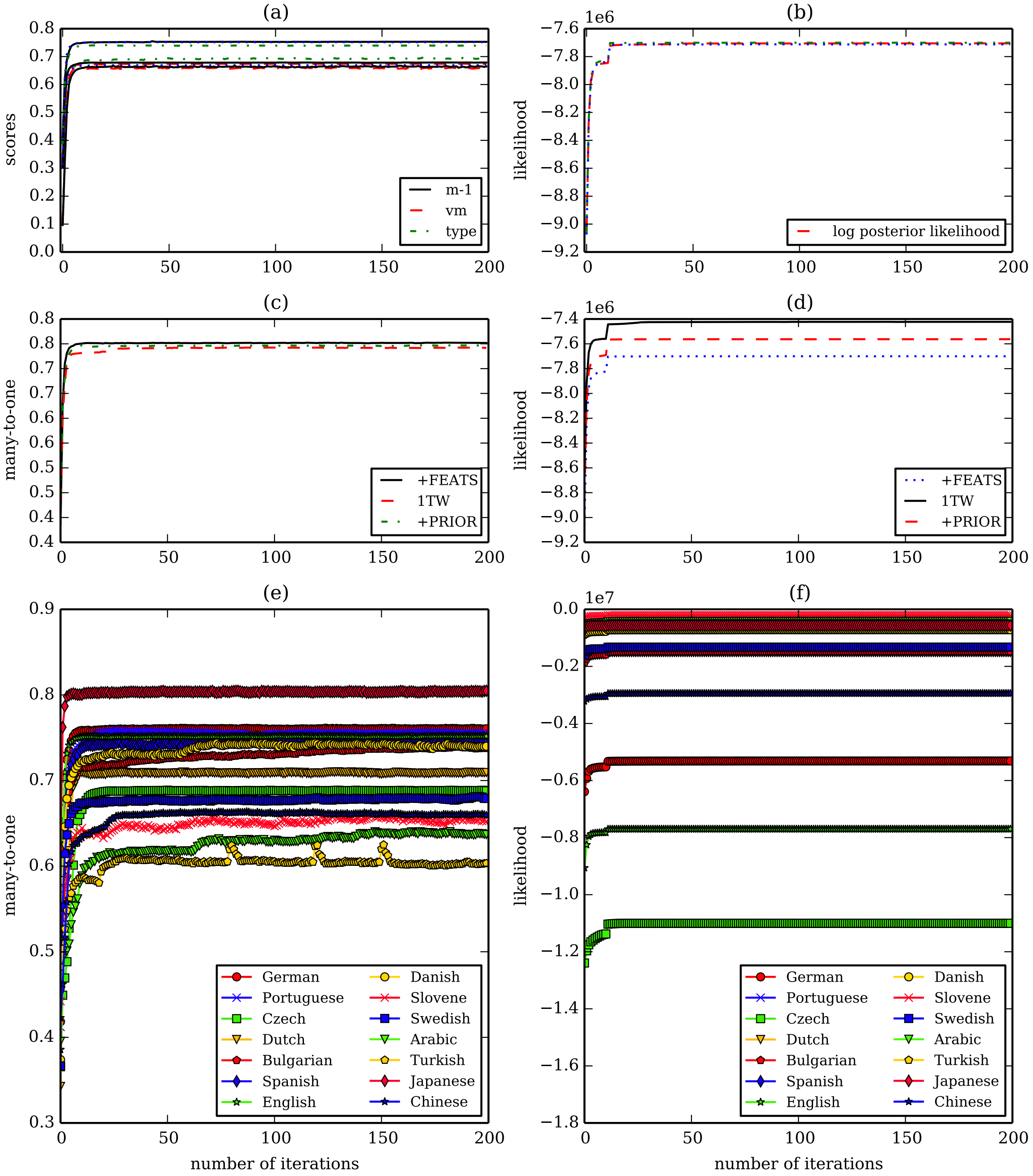


Figure 2-5: POS tagging metrics against number of rounds of Gibbs sampling. We show tagging scores in the left panels, and log posterior likelihood on the right panels. Figures (a) and (b) plots all evaluation metrics and the log posterior likelihood, respectively, across random restarts of our full model on English. Figures (c) and (d) show the many-to-one and log posterior likelihood over one run across model variants on English. Model components cascade, so +PRIOR includes the 1TW basic model and +FEATS is our full model. Figures (e) and (f) show one run of our full model for each of the languages we evaluated.

Language	1TW	+PRIOR	+FEATS
English	0.9 / 0.5 / 0.8	0.6 / 0.6 / 0.4	0.5 / 0.4 / 1.1
Arabic	1.0 / 0.7 / 2.8	0.9 / 0.8 / 1.9	1.8 / 1.4 / 1.9
Bulgarian	0.3 / 0.3 / 0.6	0.8 / 0.6 / 0.5	0.7 / 0.2 / 1.6
Chinese	0.7 / 0.8 / 0.8	1.0 / 0.8 / 1.3	1.7 / 1.3 / 1.5
Czech	4.3 / 3.3 / 4.9	4.1 / 2.6 / 1.8	2.7 / 2.2 / 3.0
Danish	1.3 / 1.1 / 1.2	1.5 / 1.4 / 0.7	1.5 / 1.4 / 0.7
Dutch	1.2 / 0.7 / 3.9	1.6 / 1.1 / 2.5	1.8 / 1.3 / 2.9
German	0.7 / 0.4 / 0.9	0.3 / 0.3 / 0.5	1.0 / 0.4 / 1.9
Japanese	0.7 / 0.3 / 1.6	0.6 / 0.2 / 1.2	1.1 / 0.3 / 1.3
Portuguese	0.8 / 1.2 / 0.7	0.7 / 1.4 / 1.3	1.2 / 2.0 / 1.7
Slovene	1.7 / 1.3 / 2.1	1.8 / 1.6 / 2.1	1.9 / 1.2 / 0.7
Spanish	0.9 / 0.8 / 1.0	1.7 / 1.2 / 0.7	0.5 / 0.6 / 1.0
Swedish	1.5 / 0.9 / 1.1	0.6 / 0.2 / 0.7	0.4 / 0.4 / 0.6
Turkish	0.4 / 0.8 / 1.8	0.6 / 0.9 / 1.5	1.0 / 0.6 / 1.4

Table 2.8: Standard deviations of POS tagging metrics for each language and model. In each entry, we report the standard deviation of the many-to-one accuracy, v-measure, and the type-level accuracy in this order.

suggests that the Markov chain has mixed and the drawn posterior samples have become independent of their initial states.

2.6.5 Error Analysis

Tables 2.10 and 2.11 provide insight into the behavior of different models (on English) in terms of the distribution of predicted tags at the type-level and token-level respectively. For each model (1TW, +PRIOR, and +FEATS), we pick the output corresponding to the random restart with the median many-to-one accuracy. The tables show that our full model produces tag distributions closest to the gold standard both at the token and the type level. For example, in English proper common nouns (NN) are most common at the token-level although proper nouns (NNP) are most frequent at the type-level. Our basic model (second row) fails to to make this distinction. But once the tag prior component is added (third and fourth rows), the model recovers the the relative ordering correctly. Appendix A compares outputs of our model variants quantitatively for all languagues.

Language	1TW	+PRIOR	+FEATS
English	$-0.2/-0.7/-0.1$	$+0.6/-0.3/-0.1$	$-1.3/-0.9/-1.4$
Arabic	$-0.5/-1.1/+0.3$	$-1.4/-2.0/+1.1$	$+1.0/+0.7/+1.2$
Bulgarian	$+0.3/+0.0/+0.5$	$-0.6/-0.6/-0.5$	$+0.1/+0.1/-0.4$
Chinese	$+0.5/+0.5/-1.2$	$+0.4/+0.3/+1.0$	$+0.5/+0.1/-0.9$
Czech	$+1.0/+0.8/-2.9$	$-0.4/-1.0/-1.1$	$-0.1/-0.4/-2.0$
Danish	$+0.4/-0.1/+0.5$	$-0.4/-0.7/-1.6$	$-0.4/-0.8/+0.0$
Dutch	$+3.0/+2.2/+3.0$	$-0.6/-0.6/+2.8$	$+0.4/+0.7/+2.3$
German	$+0.2/-0.1/-1.0$	$+0.3/-0.1/-0.3$	$-0.7/-0.5/-2.4$
Japanese	$+0.2/+0.0/-1.2$	$+0.0/-0.2/-1.2$	$+0.1/-0.1/-0.7$
Portuguese	$+0.4/+0.2/+0.9$	$+0.3/-0.1/-1.7$	$-0.7/-0.5/-1.2$
Slovene	$+0.4/-0.2/+0.6$	$+0.4/+0.4/+1.7$	$+0.4/+0.4/+0.9$
Spanish	$-1.1/+0.0/+0.2$	$-0.8/-0.4/-0.4$	$-0.9/-0.8/+0.5$
Swedish	$-1.2/-0.2/-0.2$	$+0.1/+0.2/+0.1$	$+0.1/-0.2/+0.2$
Turkish	$-0.1/+0.5/+0.3$	$+0.0/-0.3/+1.6$	$+0.1/+0.4/+0.1$
pair t-test	$\sim / \sim / \sim$	$\sim / * / \sim$	$\sim / \sim / \sim$

Table 2.9: Impact of frequency-based initialization on POS induction. Using the same experiment settings (Section 2.5.2), we initialize our HMM model with a frequency-based initialization heuristic (Section 2.6.4). We report the difference in tagging scores between random initialization and this heuristic, for each language, model, and evaluation metric (many-to-one token accuracy, v-measure, and type-level accuracy in this order), i.e. a positive number indicates random initialization is better. $*$ and $\sim$ denote a two-tail p-value of less than 0.05 and no significance respectively.

	Top 5 (type-level)	Bottom 5 (type-level)
Gold	NNP NN JJ NNS CD	-LRB- EX -RRB- # WP$
Basic	NN NNP JJ CD RB	, POS TO $.
+PRIOR	NNP JJ NN NNS CD	, POS TO . ”
+FEATS (Full Model)	NNP NN JJ CD NNS	$ MD , TO .

Table 2.10: Type-level POS tag ranking for English. The first row shows the ranking from gold annotations and the next three rows show outputs from our model variants.

	Top 5 (token-level)	Bottom 5 (token-level)
Gold	NN IN NNP DT JJ	WP$ # UH SYM LS
Basic	NN DT IN NNP JJ	POS VBN “ $ ”
+PRIOR	NN IN NNP DT NNS	VBD POS MD WDT ”
+FEATS (Full Model)	NN DT NNP IN NNS	$ MD “ ” -RRB-

Table 2.11: Token-level POS tag ranking for English. The first row shows the ranking from gold annotations and the next three rows show outputs from our model variants.

2.7 Conclusion, Current Results, and Future Work

We have presented a method for unsupervised part-of-speech tagging that considers a word type and its allowed POS tags as a primary element of the model. This departure from the traditional token-based tagging approach allows us to explicitly capture type-level distributional properties of valid POS tag assignments as part of the model. The resulting model is compact, efficiently learnable and linguistically expressive. Our empirical results demonstrate that the type-based tagger rivals state-of-the-art tag-level taggers which employ more sophisticated learning mechanisms to exploit similar constraints.

Since the word class models have been formulated by Brown et al. [13] and Clark [20], the effectiveness of using one-type-per-word representation for unsupervised POS induction is also simultaneously demonstrated in the same year of our conference publication [74] by Christodoulopoulos et al. [17] and Lamar et al. [71]. Subsequently, this fundamental approach has extended to encode shallow morphological knowledge. For instance, Blunsom and Cohn [9] extended the Bayesian one-tag-per-word HMM to incorporate the Pitman-Yor process prior. They also model morphological affixes with a character language model leading to improve induction results. Recently, Dubbin and Blunsom [32] extend the previous model to employ ambiguity classes over tags which gives improvement for some languages. Christodoulopoulos et al. [18], on the other hand, model context without introducing direct dependencies between neighboring latent variables. They instead use a mixture model to incorporate context as features. Using this feature engineering framework, they also encode lexical suffixes as features, leading to improvements in several languages. Our model, however, still gives the state-of-the-art results for Bulgarian, German, Japanese, and Slovene.

A promising direction for this thread of work is to explicitly model the rich morphological interactions between POS categories. Toutanova and Cherry [125] have exploited morphological lexicons for POS prediction and lemmatization. We believe the approach can be taken a step further in an unsupervised setting because modeling morphological relationships can further constrain the set of possible tags a word type

Language	+FEATS	Best recent	
English	74.6 / 67.1	77.5 / 69.8	BC11
Arabic	62.1 / 40.3	67.5 / -	BC11
		60.7 / 43.3	CGS11
Bulgarian	73.1 / **60.8**	76.2 / -	DB14
		66.5 / 55.6	CGS11
Chinese	66.3 / 35.0	69.4 / 42.6	CGS11
Czech	65.1 / 47.8	70.1 / -	BC11
		65.7 / 48.4	CGS11
Danish	72.2 / 57.6	76.1 / -	BC11
		71.1 / 59.0	CGS11
Dutch	69.0 / 54.2	71.1 / 54.7	CGS11
German	**74.9** / **64.8**	74.4 / 61.9	CGS11
		73.9 / 63.0	C03
Japanese	**79.9** / **79.3**	78.5 / 77.4	CGS11
		77.4 / 78.6	C03
Portuguese	75.3 / 62.6	78.5 / -	BC11
		76.8 / 63.9	CGS11
Slovene	**64.2** / 51.0	**64.2** / 52.6	CGS11
		63.5 / 53.9	C03
Spanish	74.2 / 62.8	80.0 / -	DB14
		71.7 / 63.2	CGS11
Swedish	68.4 / 57.6	70.4 / -	DB14
		68.7 / 58.9	C03
Turkish	59.9 / 39.6	62.8 / 40.8	CGS11

Table 2.12: Survey of recent results in POS induction. We show the best recent results under the many-to-one accuracy (m-1) and (first row for each language) the v-measure (second row). C03 [20] is a word clustering HMM that employs morphological information. BC11 [9] is the Pitman-Yor letter n-gram extension of our Bayesian HMM. CGS11 [18] is a feature-based the Bayesian mixture model. DB14 [32] is an extension of BC11 that uses ambiguity classes instead of a single tag for each word type. Using additional unlabeled data (Wall Street Journal for English and Wikipedia for other languages), Yuret et al. [135] obtain better results for English (80.2 m-1), Bulgarian (75.1 m-1), Dutch (71.2 m-1), and Turkish (63.7 m-1).

can take. For instance, inflectional and derivational morphology maps the POS of a word into distinct sets of candidates. Because the class of morphological transformation are typically realized with a number of recurring lexical cues, we hypothesize a tree-based or graph-based prior can help recover morphological structure in the tagging lexicon.

Chapter 3

Improving Unsupervised Word Segmentation with Morpho-syntactic Connections

3.1 Introduction

In the previous chapter, we are concerned with inducing syntactic categories (parts-of-speech) from unlabeled sentences. Here, we consider the task of unsupervised *morphological segmentation* which is the problem of analyzing the internal structure of a word by dividing it into substrings. Given, for example, an English word:

unlabeled

we break the word up into smaller units called *morphemes*:

un	–	label	–	ed
(prefix)		(stem)		(suffix)

The segmentation of the above word reveals how it is composed from its parts. The *stem* gives core meaning, and *affixes* are concatenated to create a related word. In the above example, *prefix* "un" indicates negation and *suffix* "ed" marks the word as a past tense verb. Not all words can be cleanly delineated into complete morphemes, for instance the word "unsupervised", which is commonly segmented as "un–supervis–ed". Nevertheless, this concatenative assumption remains widely adopted by many applications. For instance, removing suffixes heuristically with a stemmer to reduce data sparsity has become a defacto preprocessing step in information retrieval models [6, 111].

As seen above, once a word is decomposed, its syntactic role becomes apparent. In fact, a tight connection between morphology and syntax is well-documented in linguistic literature. In many languages, morphology plays a central role in marking syntactic structure, while syntactic relations help to reduce morphological ambiguity [52]. Therefore, in an unsupervised linguistic setting which is rife with ambiguity, modeling this connection can be particularly beneficial.

However, existing unsupervised morphological analyzers take little advantage of this linguistic property. In fact, most of them operate at the vocabulary level, completely ignoring sentence context. This design is not surprising: a typical morphological analyzer does not have access to syntactic information, because morphological segmentation precedes other forms of sentence analysis.

In this chapter, we demonstrate that morphological analysis can utilize this connection without assuming access to full-fledged syntactic information. In particular, we focus on two aspects of the morpho-syntactic connection:

- **Morphological consistency within POS categories.** Words within the same syntactic category tend to select similar affixes. This linguistic property significantly reduces the space of possible morphological analyses, ruling out assignments that are incompatible with a syntactic category.

- **Morphological realization of grammatical agreement.** In many morphologically rich languages, agreement between syntactic dependents is expressed via correlated morphological markers. For instance, in Semitic languages, gender and number agreement between nouns and adjectives is expressed using matching suffixes. Enforcing mutually consistent segmentations can greatly reduce ambiguity of word-level analysis.

Arabic word	سيعود				
Position	5	4	3	2	1
Individual Arabic glyph	د	و	ع	ي	س
ASCII character mapping	d	w	E	y	s
Transliteration	syEwd				

Table 3.1: Example of Buckwalter transliteration for Modern Standard Arabic. In the first row, we show an Arabic word (which is written right to left). The subsequent rows show the transliteration for individual Arabic orthographic symbols. The last row shows the final transliteration (which is written left to right, like we do in English.)

In both cases, we do not assume that the relevant syntactic information is provided, but instead jointly induce it as part of morphological analysis.

We capture morpho-syntactic relations in a Bayesian model that grounds intra-word decisions in sentence-level context. Like traditional unsupervised models, we generate morphological structure from a latent lexicon of prefixes, stems, and suffixes. In addition, morphological analysis is guided by a latent variable that clusters together words with similar affixes, acting as a proxy for POS tags. Moreover, a sequence-level component further refines the analysis by correlating segmentation decisions between adjacent words that exhibit morphological agreement. We encourage this behavior by encoding a transition distribution over adjacent words, using string match cues as a proxy for grammatical agreement.

Here, we perform unsupervised morphological segmentation in Modern Standard Arabic (MSA). Given a corpus of just words, our goal is segment each word type into morphemes. We do not require parts-of-speech annotations, lexicons, or any knowledge bases. Section 3.3 describes a series of four models of increasing complexity. The first two models just require a list of word types (frequency counts are not needed). The last two models exploit contexts of words to improve segmentation performance and hence require complete sentences. Note that throughout this chapter, we typeset MSA words with the Buckwalter transliteration system [15], which is a one-to-one mapping of Arabic scripts to ASCII letters. Table 3.1 shows an example.

Specifically, given an Arabic word, we perform the following:

Input:			syEwd		
Output:	s	–	y	–	Ewd
English gloss (not expected)	will		he/it		return

The above word comprises of a verb stem (Ewd) and two prefixes. Prefix "s" marks the tense of the verb (future tense), whereas prefix "y" indicates that the verb is third person masculine or neuter singular. Again, we see that the morphological structure of the word and its syntactic role are interdependent.

We evaluate our model on the standard Arabic treebank. Our full model yields 86.2% accuracy, outperforming the best published results [101] by 8.5%. We also found that modeling morphological agreement between adjacent words yields greater improvement than modeling syntactic categories. Overall, our results demonstrate that modeling syntactic information is a promising direction for improving morphological analysis.

3.2 Related Work

3.2.1 Local Boundary-based Segmentation

A family of boundary-based segmentation research draws its inspiration from Harris [54, 55]. He introduces a number of heuristics for scoring each letter position for a word under consideration. The score is then used to judge whether it is appropriate to insert a morpheme boundary at each possible position. For example, given a corpus of word types $\boldsymbol{W}$, the *letter successor variety* (LSV) [48, 54, 55] of a prefix x is the number of distinct single letters y that can possibly follow it, i.e. the concatenated string xy is also a prefix of some word or a word itself in the corpus:

$$\text{LSV}(x) = |\{y|w \in \boldsymbol{W}\text{and prefix}(w) = xy\}|$$

For instance, suppose our corpus consists of word types

$$\boldsymbol{W} = \{\mathsf{unsupervised}, \mathsf{unsegmented}, \mathsf{unlabeled}\},$$

then the LSV for some prefixes are computed as follows:

Prefix	Matching words	Letter Successor Variety
u	unsupervised, unsegmented, unlabeled	1 {n}
un	unsupervised, unsegmented, unlabeled	2 {s,l}
uns	unsupervised, unsegmented	2 {u,e}
unsupervis	unsupervised	1 {e}
super	∅	0 {}

At morpheme boundaries, such a score aims to give a high value, which suggests that the prefix can be easily composed with other substrings to form valid words. Inside boundaries, the intuition is that there are regularities that limit what letter might follow, and thus LSV gives a low value.

To increase the robustness of boundary detection, *letter predecessor variety* (LPV) is defined analogously by applying the same technique in the other direction. For instance, using the corpus in the example above, we obtain:

Suffix	Matching words	Letter Predecessor Variety
d	unsupervised, unsegmented, unlabeled	1 {e}
ed	unsupervised, unsegmented, unlabeled	3 {e,s,t}
sed	unsupervised	1 {i}
supervised	unsupervised	1 {n}
super	∅	0 {}

Building upon Harris's concept of letter varieties, a number of researchers [35, 48, 63] propose using entropy as an alternative. The primary difference is that computation of entropy involves the number of matching word types, whereas the letter varieties metrics above just counts the number of distinct letters. Thus, *letter successor entropy* for a prefix x is defined as:

$$\mathrm{LSE}(x) = -\sum \frac{n(xy)}{n(x)} \log \frac{n(xy)}{n(x)},$$

where $n(xy)$ and $n(x)$ are the number of word types beginning prefixes xy and x respectively. Using the example above, the LSE for some segments are:

Prefix	Matching words	Letter Successor Entropy
un	unsupervised, unsegmented, unlabeled	$\text{entropy}(\{\frac{2}{3}, \frac{1}{3}\})$
uns	unsupervised, unsegmented	$\text{entropy}(\{\frac{1}{2}, \frac{1}{2}\})$

And the *Letter Predecessor Variety* is defined similarly. This give rise to a total of four scores for any potential boundary in a word.

Harris [54] and Hafer and Weiss [48] propose a number of ways to use these metrics for segmentation. At each position in a word, three main criteria are used to determine if a boundary should be introduced:

(a) Cutoff threshold: If the score exceeds a pre-determined threshold, a boundary is introduced

(b) Word match: If the remaining substring that comes after (or before) a prefix (suffix) appears in the corpus, a cut is made.

(c) Peak and plateau: If the score is a local maximal, the word is broken at this position. The maximal can be a plateau, i.e. it has the same value as its neighbors.

In fact, Hafer and Weiss [48] combine the four metrics and the three criteria in 15 ways. For example, one condition segments a word where both the LSE and the LPE exceeds the cutoff and the remaining segment matches a word in the corpus. This approach is further extended by introducing new heuristics for boundary scores [37, 51, 122] and segmentation decision-rules [2, 22, 82].

Following this intuition, a body of work extend the unit from which boundary statistics are computed from single letters to sequences of characters. In language acquisition research, it is found that syllables around a segmentation boundary show regularities [3, 64, 110]. For instance, the conditional probability of the syllable after a boundary given the syllable before it (obtained from a training corpus) is an effective (although not only) explanation of how infants segment words. This approach found itself as a means for incorporating dependencies in many sequence segmentation models such as generative probabilistic models [11, 42, 87, 128]. Our model does not model local n-gram character transitions at boundaries but captures dependencies between morphemes. Moreover, we capture them with a more elaborate model involving latent variables so that dependencies exist not only between adjacent morphemes but also between affixes and suffixes far apart.

3.2.2 Lexicon or Grammar-based Models

Another influential approach is based on lexicon or grammar-based sequence segmentation research. In contrast to local boundary-based methods, these models tie segmentation decisions of the whole corpus via the lexicon or the grammar. The early work of Olivier [99] (as described in Kit [65, Chapter 4.3]) presents an iterative algorithm for segmenting words and learning a lexicon simultaneously. It proceeds by initializing the lexicon with a uniform distribution over the character set of the language. In each iteration, the algorithm first obtains the maximum likelihood segmentation of the corpus using the lexicon. In the second step, the lexicon is updated using a set of heuristics and the probability distribution over the entries are re-estimated. For example, bigram segments are added to the lexicon and low frequency segments in the lexicon are pruned.

The MK10 algorithm [132] is a similar but non-probabilistic approach. Given a sequence of characters, the algorithm operates in an iteration manner to recover pairs of segments that appear at least 10 times. Similarly, the initial lexicon consists of all single characters that are expected from the language. As the algorithm scans the input string, it segments and updates the lexicon. At any position in the input data, the algorithm decides how many characters ahead constitute a segment by matching an entry in the lexicon. Then, it updates the frequency count of the entry. If any pair of adjacent segments seen thus far exceeds 10, it is added to the lexicon and the frequency counts are reset.

In later years, Nevill-Manning and Witten [93] introduce a single-pass algorithm Sequitur[1] for learning a context-free grammar (CFG) that generates the input sequence, and as a by-product the grammar also segments the data in a hierarchical

[1] `http://www.sequitur.info`

fashion. To reduce redundancy, they impose two constraints on the CFG:

(1) No (ordered) pair of adjacent symbols in any CFG rule appears more. (For example, an invalid rule is S→abcab.)

(2) Every CFG rule is used more than once. (This helps to combine rules such that the new rule generates more symbols on the right hand side, and hence prevents over-segmentation.)

For example, representing an input corpus consisting of three words as a sequence, Sequitur produces the following output:

Input unsupervised • unsegmented • unlabeled

Output: (Hierarchical segmentation/bracketing)

[[un] s] upervis [[ed] •][[un] s] egment [[ed] •][un] label [ed]

(CFG rules)

S → R_1 u p e r v i s R_2 R_1 e g m e n t R_2 R_3 l a b e l R_4

R_1 → R_3 s

R_2 → R_4 •

R_3 → u n

R_4 → e d

The approach of incorporating a lexicon or a grammar forms the basis of many other segmentation approaches, such as probabilistic grammar models [61, 133],finite state machine models[2] [70, 120], and heuristic rule-learning [28]. In a similar vein, our model learns a morpheme lexicon as it recovers word segmentations. Moreover, our model also explicitly aims to learn a compact lexicon much like the minimum description length models that we shall describe in the next section.

[2] These require handcrafted knowledge or annotated data for learning or a hybrid of both.

3.2.3 Minimum Description Length Models

There is also considerable effort in using the principle of minimum description length (MDL) [44, 104, 130] to improve lexicon and grammar-based models. In the context of word segmentation, the MDL principle, like traditional lexicon and grammar models, seeks to succinctly represent words using morphemes. However, in contrast to the models described in the previous section, MDL also aims to achieve a compact lexicon or grammar.

The early work of Brent [10] illustrates this principle by explaining word formation in English with a generative process. In this work, English words are generated by composing a stem and a suffix from their respective lexicons. Because each word has two segments, the size of the overall encoding is determined by the number of unique stems and suffixes. Using the same example corpus of word types as before,

$$\boldsymbol{W} = \{\mathsf{unsupervised}, \mathsf{unsegmented}, \mathsf{unlabeled}\},$$

the first segmentation has smaller lexicons and also turns out to be more natural than the second one:

	Stems	Suffixes	Lexicon Size
(a)	unsupervi unsegment unlabel	ed	4
(b)	unsuperv unsegmen unlabe	sed, ted led	6

Brent [11] extends the MDL-based segmentation to handle multiple segments, specifically for the problem of word boundary segmentation (where the input is an utterance without word delimiters). The model generates an input corpus of utterances probabilistically as follows:

(1) Generate the size of the lexicon of word types (including the utterance/sentence boundary)

(2) Generate the word types in the lexicon

(3) Generate the number of occurrences of each word type

(4) Generate an ordering of word tokens

(5) Generate the input un-delimited string by removing all the boundaries

In contrast to his earlier work, a different measurement of the compactness of the lexicon is used in this model. Here, the MDL principle is used in the first generation step by specifying a probability distribution that favors fewer lexical items (unique segments). Specifically, a lexicon of size k is generated with a probability:

$$Pr(i) \propto \frac{1}{k^2}$$

In fact, the MDL principle offers a flexible framework for controlling model complexity for a variety of segmentation tasks. Creutz and Lagus [25] propose a model for morphological word segmentation where the total encoding cost of a segmented corpus decomposes into two terms. The first term is the cost of encoding the corpus by representing it as morphemes. Each morpheme token m is generated independently and has cost given by the negative log-likelihood $p(m)$ (estimated with maximum likelihood):

$$\sum_{m} -\log p(m).$$

The second term is the complexity term for encoding morpheme types. Each morpheme type is encoded at the character level. Using a k-bit encoding for each character, the total cost for encoding the morpheme lexicon becomes:

$$\sum_{m'} k \cdot l(m'),$$

where $l(m')$ is the length of morpheme type m'. The desired segmentation balances between employing a concise lexicon with a small number of short morphemes, and achieving a high data likelihood by generating the corpus with as few highly recurring morphemes as possible.

MDL is incorporated into a number of other segmentation research. For example, de Marcken [30] devises a probabilistic grammar-based model that performs hierarchical text segmentation using lexicon and grammars that have compact encodings. Kit and Wilks [66] present an algorithm for utterance segmentation that achieves compression by optimizing for the *description length gain*, which is defined as a relative reduction in entropy. Goldsmith [39] proposes a suffix segmentation model that are based on minimal *signatures*, where a signature is a group of suffixes that are compatible with the same stem. (For example, stems{segment, train } share the set of verb inflectional suffixes contained in the signature {s, ing, ed}). Similarly, our segmentation model, like a number of recent work [42, 60, 61, 87] encourages a compact lexicon representation within a Bayesian generative framework.

3.2.4 Segmentation Recovery via Graphical Model Inference

Research in unsupervised morphological segmentation has gained momentum in recent years bringing about significant developments to the area due to flexibility of graphical model inference techniques. These advances include novel Bayesian formulations [42, 60]. Goldwater et al. [42] present a Bayesian generative model that combines the preference for a small lexicon with the goal of incorporating dependencies between adjacent segments. In their unigram model, segments are draw from a Dirichlet process prior that encourages probability mass to be placed on few morpheme types. (In their bigram model, a hierarchical Dirichlet process prior is used for the conditional probability of drawing the current segment based on the previous one.) The distinguishing feature is that recovering segmentation is now cast as graphical model inference. This is in contrast to prior MDL models which mostly perform heuristic greedy optimization. Moreover, the use of generic graphical model inference techniques allow models to be modified in elaborate ways. In fact, this line of work also motivated Bayesian segmentation models for speech data. Lee and Glass [73] present a Dirichlet process mixture model where each mixture component now is a HMM whose emission probability is governed by a Gaussian mixture model. On English, the model segments acoustic signals into sub-word units which are found to be highly correlate with actual English phones. For instance, adaptor grammars [62] which are Bayesian counterparts of grammar-based utterance segmentation models have been proposed. Such models employ a non-parametric prior that encourages production rules to be re-used and explain how words can be generated [60, 61]. O'Donnell et al. [96] develop fragment grammar, a generalization of the adaptor grammar, which is used to explain derivational morphology of English words by modeling productivity of suffixes.

Poon et al. [101] perform unsupervised word segmentation with an undirected graphical model that incorporates rich features using a log-linear parameterization. In their model, the joint probability of a corpus of words W and its segmentation S is given by

$$P(W, S) \propto \exp(\theta \cdot f(W, S)),$$

where θ is the parameter weights and $f(W, S)$ is a feature function that decomposes into a sum of local and global features. Global features track the number of morpheme types and their weights are set to negative to encourage compact lexicons similar to traditional MDL approaches. Local features examine each word and return n-gram characters around the segmentation boundaries. The parameters of the model are learned by maximizing the marginal likelihood which is approximated with the contrastive estimation technique of Smith and Eisner [114].

Another recent approach is the development of multilingual morphological segmenters using the graphical modeling framework. Snyder and Barzilay [117] propose a Bayesian generative model for segmenting parallel bilingual phrases. At the core, the model generates a distribution over bilingual morpheme pairs (*abstract morphemes*), a distribution over morphemes in one language that do not have a counterpart in the other language (*stray morphemes*), and a distribution over stray morphemes in the other language. To generate an aligned bilingual phrase, the process generates a number of abstract and stray morphemes before ordering them. These unobserved morphemes are recovered using standard Markov chain Monte Carlo techniques for undirected graphical model inference.[3]

Bayesian generative segmentation model. We also adopt a graphical modeling ap-

[3] In their earlier model [116], some segmented words in one or both languages are required.

proach to morphological segmentation, although our model incorporates distinctive linguistic features. The main departure of our Bayesian segmenter is that we incorporate the connection between morphology and syntax into the generative process. This is similar to a number of recent work which we shall describe next.

3.2.5 Combining Morphology and Syntactic Analysis

Our work most closely relates to approaches that aim to incorporate syntactic information into morphological analysis. Surprisingly, the research in this area is relatively sparse, despite multiple results that demonstrate the connection between morphology and syntax in the context of part-of-speech (POS) tagging [1, 27, 46, 126]. For instance, Toutanova and Johnson [126] use a latent variable to represent an ambiguity class of tags in a generative POS tagger. The ambiguity class then generates orthographic morphological features of word types. Hence, observed morphological features of words help to induce better POS tags in this model. Dasgupta and Ng [27] also employ orthographic morphological features of words to improve unsupervised POS tagging. The primary difference is that they first cluster words based on their morphological suffixes (obtained using an unsupervised morphological analyzer), then use these clusters as better seeds to improve POS induction. Adler and Elhadad [1] employ a morphological analyzer for POS induction in a different way. The analyzer proposes several possible segmentations for each word, and the authors formulate a lattice-based modification of the expectation maximization (EM) algorithm to learn parameters for the HMM tagger.

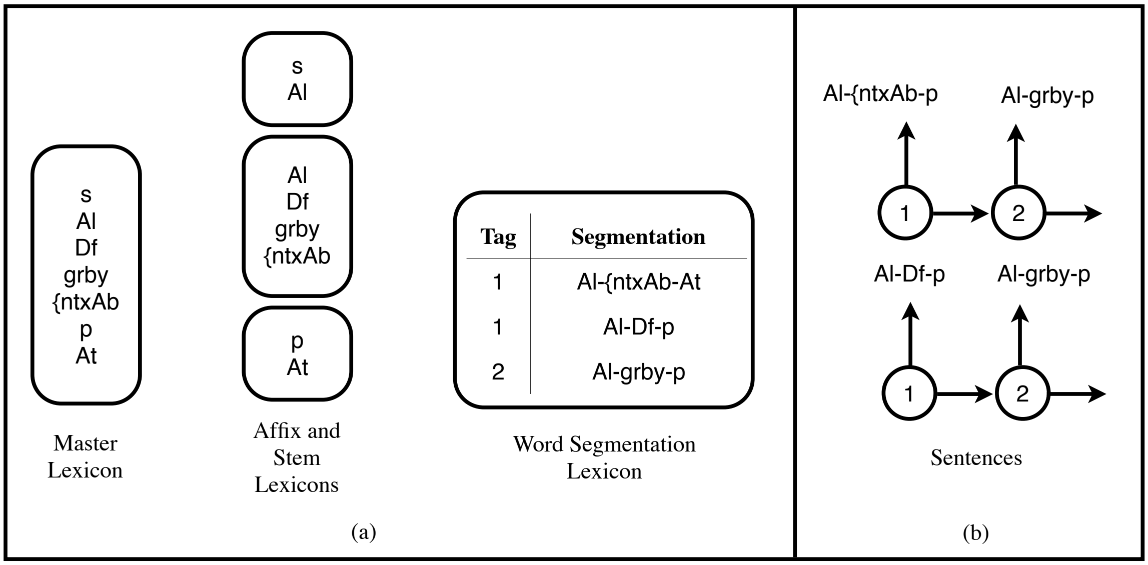
On the other hand, Habash and Rambow [46] are concerned with improving supervised morphological tagging of Arabic with using morphological analyzers. Specifically, the goal is to tag each word with a rich set of POS tags that describes its syntactic and morphological properties, such as syntactic category, gender, and number. Their approach uses a lexicon-based morphological analyzer to propose multiple morphological tag sets for a word, then feed its output into a supervised pipeline that ultimately predicts the POS tags. In contrast, we model the connection between morphology and syntax to improve unsupervised morphological segmentation. Moreover, our approach does not require labeled POS data.

Toutanova and Cherry [124] were the first to systematically study how to incorporate part-of-speech information into lemmatization and empirically demonstrate the benefits of this combination. While our high-level goal is similar, our respective problem formulations are distinct. Toutanova and Cherry [124] have considered a semi-supervised setting where an initial morphological dictionary and tagging lexicon are provided but the model also has access to unlabeled data. Since a lemmatizer and tagger trained in isolation may produce mutually inconsistent assignments, and their method employs a log-linear reranker to reconcile these decisions. This reranking method is not suitable for the unsupervised scenario considered in our paper.

Our work is most closely related to the approach of Can and Manandhar [16]. Their method also incorporates POS-based clustering into morphological analysis. These clusters, however, are learned as a separate preprocessing step using distributional similarity. For each of the clusters, the model selects a set of affixes, driven by the frequency of their occurrences in the cluster. In contrast, we model morpho-syntactic decisions jointly, thereby enabling tighter integration between the two. This design also enables us to capture additional linguistic phenomena such as agreement. While this technique yields performance improvement in the context of their system, the final results does not exceed state-of-the-art systems that do not exploit this information, for example the language-independent segmenter of Creutz and Lagus [26].

3.3 Model

Given a corpus of unannotated and unsegmented sentences, our goal is to infer the segmentation boundaries of all words. We represent segmentations and syntactic categories as latent variables with a directed graphical model. The model starts by generating a number of morpheme lexicons that leads to the creation of a word-tag lexicon. The lexicon then generates sentences using a HMM that respects constraints specified in the lexicon, specifically all token occurrences of a word type have the



Tag	Segmentation
1	Al-{ntxAb-At
1	Al-Df-p
2	Al-grby-p

Figure 3-1: Examples of structures generated by our morphological word segmentation model: (a) shows a master lexicon from which the prefix, the suffix, and the stem lexicons are generated. Using the affix and the stem lexicons, the word segmentation lexicon is created. Note that each word type has one corresponding tag and segmentation. (b) shows sentences that are generated from a HMM that respects constraints specified by the word segmentation lexicon. For example, state 2 cannot generated word "Al–Df–p". A deterministic process creates unsegmented words (omitted to avoid clutter) from the segmented HMM tokens. Morphological word segmentation amounts to inferring what unobserved structures (such as the lexicons) are given the observed unsegmented sentences.

same tag and segmentation. Examples of such structures are shown in Figure 3-1. We perform Bayesian inference to recover the latent variables of interest.

Apart from learning a compact morpheme lexicon that explains the corpus well, we also model morpho-syntactic relations both within each word and between adjacent words to improve segmentation performance. In the remaining section, we first provide the key linguistic intuitions on which our model is based before describing the complete generative process.

3.3.1 Model Overview

We develop a series of cascading models of increasing sophistication. Model 1 (BASIC) incorporates basic intuitions of word morphology in a Bayesian generative model.

For instance, we draw inspiration from prior lexicon-based segmentation research by first generating morpheme lexicons from which words are later composed. Model 2 (+POS) introduces a latent variable to couple affixes dependencies that arise due to morphological consistency within syntactic category. Model 3 (+Token-POS) starts to exploit token-level dependencies between latent syntactic categories of adjacent words. This makes affixes of adjacent tokens dependent on another when the POS tags are unobserved. Finally, Model 4 (+Token-Seg) fuses the second morpho-syntactic connection that morphological markers realize grammar agreement at the token-level.

3.3.2 Linguistic Intuition

While morpho-syntactic interface spans a range of linguistic phenomena, we focus on two facets of this connection. Both of them provide powerful constraints on morphological analysis and can be modeled without explicit access to syntactic annotations.

Morphological consistency within syntactic category. Words that belong to the same syntactic category tend to select similar affixes. In fact, the power of affix-related features has been empirically shown in the task of POS tag prediction [46]. We hypothesize that this regularity can also benefit morphological analyzers by eliminating assignments with incompatible prefixes and suffixes. For instance, a state-of-the-art segmenter erroneously divides the word "Al{ntxAbAt" into four morphemes "Al–{ntxAb–A–t" instead of three "Al–{ntxAb–At" (translated as "the-election-s".) The affix assignment here is clearly incompatible — determiner "Al" is commonly associated with nouns, while suffix "A" mostly occurs with verbs.

Since POS information is not available to the model, we introduce a latent variable that encodes affix-based clustering. In addition, we consider a variant of the model that captures dependencies between latent variables of adjacent words (analogous to POS transitions).

Morphological realization of grammatical agreement. In morphologically rich languages, agreement is commonly realized using matching suffices. In many cases, members of a dependent pair such as adjective and noun have the exact same suffix. A common example in the Arabic Treebank is the bigram "Al–Df–p Al–grby–p" (which is translated word-for-word as "the-bank the-west") where the last morpheme "p" is a feminine singular noun suffix.

Fully incorporating agreement constraints in the model is difficult, since we do not have access to syntactic dependencies. Therefore, we limit our attention to adjacent words which end with similar strings – for e.g., "p" in the example above. The model encourages consistent segmentation of such pairs. While our string-based cue is a simple proxy for agreement relation, it turns to be highly effective in practice. On the Penn Arabic treebank corpus, our cue has a precision of around 94% at the token-level.

3.3.3 Generative Process

The high-level generative process proceeds in five main phases:

(a) **Lexicon Component**: We begin by generating morpheme lexicons $\boldsymbol{L}$ using parameters $\boldsymbol{\gamma}$. This set of lexicons consists of separate lexicons for prefixes, stems, and suffixes generated in a hierarchical fashion.

(b) **Segmentation Component**: Conditioned on $\boldsymbol{L}$, we draw word segmentations and their syntactic categories $(\boldsymbol{S}, \boldsymbol{T})$.

(c) **Token-POS Component**: Next, we generate the segmented tokens and their syntactic classes $(\boldsymbol{s}, \boldsymbol{t})$ from a standard first-order HMM which has dependencies between adjacent syntactic categories.

(d) **Token-Seg Component**: This component augments the previous HMM with a first-order Markov chain that has dependencies between adjacent segmentations.[4]

[4] This component overgenerates and makes the probability model deficient although it improves segmentation performance.

Notation used in the type-level component

$\boldsymbol{L}$	–	The set of all lexicons
L^*	–	The master morpheme lexicon, i.e. the set of all morphemes
$L_-, L_0, L_+ \subseteq L^*$	–	The prefix, the stem, and the suffix lexicons respectively
$\sigma \in L^*$	–	A morpheme in the master morpheme lexicon
$\sigma_-, \sigma_0, \sigma_+$	–	A prefix, a stem, and a suffix morpheme
K	–	The size of tag set, i.e. the number of latent states.
α_T	–	The hyperparameter of the prior on the distribution that generates tags
Θ_T	–	The parameter for the distribution that generates tags
$\alpha_-, \alpha_0, \alpha_+$	–	The hyperparameter of the prior on the distribution that generates prefixes, stems, and suffixes respectively
$\Theta_{-\|T}, \Theta_0, \Theta_{+\|T}$	–	The parameter for the distribution that generates prefixes, stems, and suffixes respectively
$\boldsymbol{W}, \boldsymbol{S}, o\boldsymbol{T}$	–	The sequence of word types, tag assignments, and segmentations in the lexicon
W, T, S	–	A word type, its tag, and its segmentations (which is a sequence of morphemes)
$\| \cdot \|$	–	The length of a morpheme, the size of a lexicon, or the number of segments (depending on usage context)
$\boldsymbol{\gamma}$	–	All type-level hyperparameters
γ_l	–	The parameter of the geometric distribution for morpheme length
$\gamma_-, \gamma_0, \gamma_+$	–	The parameter of the geometric distribution for the lexicon size for prefixes, stems, and suffixes respectively
$\gamma_{\|S\|}$	–	The parameter of the distribution for the number of segments

Notation used in the token-level component

$\boldsymbol{w}, \boldsymbol{t}, \boldsymbol{s}$	–	The word tokens and their corresponding tags and segmentations
w_i, t_i, s_i	–	The i^{th} word token, its tag, and its segmentation in the corpus
$\theta_{t\|t}, \theta_{w\|t}$	–	The transition and emission distributions for the HMM
$\alpha_{t\|t}, \alpha_{w\|t}$	–	The hyperparameter of the prior on the transition and the emission distributions respectively
$\boldsymbol{\beta}$	–	The parameters of the segmentation transition distribution

Table 3.2: Summary of notation used for our type-level segmentation model. In general, capital random variables are types and lowercase are token-level.

(e) **Fusing morphemes**: Finally, a deterministic process removes morpheme boundaries to form an unsegmented corpus of word types $\boldsymbol{W}$ and tokens $\boldsymbol{w}$.

The complete generative story can be summarized by the following equation:

$$
\begin{aligned}
P(\boldsymbol{w},&\boldsymbol{s},\boldsymbol{t},\boldsymbol{W},\boldsymbol{S},\boldsymbol{T},\boldsymbol{L},\Theta,\boldsymbol{\theta}|\gamma,\boldsymbol{\alpha},\boldsymbol{\beta}) = \\
&P(\boldsymbol{L}|\boldsymbol{\gamma}) && \text{(a)} \\
&P(\boldsymbol{S},\boldsymbol{T},\Theta|\boldsymbol{L},\gamma,\boldsymbol{\alpha}) && \text{(b)} \\
&P(\boldsymbol{s},\boldsymbol{t},\boldsymbol{\theta}|\boldsymbol{S},\boldsymbol{T},\boldsymbol{L},\boldsymbol{\alpha}) && \text{(c)} \\
&P(\boldsymbol{s}|\boldsymbol{S},\boldsymbol{\beta}) && \text{(d)} \\
&P(\boldsymbol{w},\boldsymbol{W}|\boldsymbol{s},\boldsymbol{S}) && \text{(e)}
\end{aligned}
$$

where $\boldsymbol{\gamma},\boldsymbol{\alpha},\boldsymbol{\Theta},\boldsymbol{\theta},\boldsymbol{\beta}$ are hyperparameters and parameters whose roles we shall detail shortly.

Our lexicon component captures the desirability of compact lexicon representation proposed by prior work by using parameters $\boldsymbol{\gamma}$ that favors small lexicons. Furthermore, if we set the number of syntactic categories in the segmentation component to one and exclude the token-based models, we recover a segmenter that is very similar to the unigram Dirichlet Process model [42, 116, 117]. We shall elaborate on this point in Section 3.4.

The segmentation component captures morphological consistency within syntactic categories (POS tag), whereas the Token-POS component captures POS tag dependencies between adjacent tokens. Lastly, the Token-Seg component encourages consistent segmentations between adjacent tokens that exhibit morphological agreement.

Lexicon Component This component captures a number of desirable properties of morphemes that are previously explored in prior work. Specifically, we model the following:

- **Morpheme length**: Our model prefers short morphemes to long ones.
- **Morpheme types**: We distinguish the type of morphemes. Starting from a

pool of morphemes in a master lexicon, we draw separate lexicons for prefixes, stems, and suffixes. We allow morphemes to be shared across lexicons. This is motivated by the morpheme “Al” in Arabic can be both a prefix and an isolated stem.

- **Lexicon size**: We also encode preference for compact lexicons, specifically affix lexicons contain fewer morpheme types than the stem lexicon.

Concretely, we begin the generative process by first drawing morphemes in the master lexicon. Each morpheme σ in the master lexicon L^* is drawn according to a geometric distribution which assigns monotonically smaller probability to longer morpheme lengths:

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

The parameter γ_l for the geometric distribution is fixed and specified beforehand. We then draw the prefix, the stem, and suffix lexicons (denoted by L_-, L_0, L_+ respectively) from morphemes in L^*. Generating the lexicons in such a hierarchical fashion allows morphemes to be shared among the lower-level lexicons. For instance, once determiner “Al” is generated in the master lexicon, it can be used to generate prefixes or stems later on. To favor compact lexicons, we again make use of a geometric distribution that assigns smaller probability to lexicons that contain more morphemes:

$$\begin{aligned} \text{prefix:} \quad & |L_-| \sim \text{Geometric}(\gamma_-) \\ \text{stem:} \quad & |L_0| \sim \text{Geometric}(\gamma_0) \\ \text{suffix:} \quad & |L_+| \sim \text{Geometric}(\gamma_+) \end{aligned}$$

By separating morphemes into affixes and stems, we can control the relative sizes of their lexicons with different parameters.

Segmentation Component This component models the morphological segmentation at the type-level, specifically:

- **Number of Morphemes**: We encourage a word to be segmented into few number of morphemes. Note that our lexicon component encourages short morphemes to prevent under-segmentation.

- **Correlation between affixes**: We model dependencies between affixes within a word by employing a parent latent POS variable in the directed graphical modeling framework. When the number of possible states for this POS tag is set to one, we recover the degenerated model that assumes affixes are independent.

The generative process continues from where the lexicon component left off by independently drawing word types. Each word type is composed of morphemes in the affix and stem lexicons with the constraint that each word has exactly one stem. The word is also encouraged to have few morphemes. We fix the number of syntactic categories (tags) to K and begin the process by generating multinomial distribution parameters for the POS tag prior from a Dirichlet prior:

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

Next, for each possible value of the tag $T \in \{1, \ldots, K\}$, we generate parameters for a multinomial distribution (again from a Dirichlet prior) for each of the prefix and the suffix lexicons:

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

By generating parameters in this manner, we allow the multinomial distributions to generate only morphemes that are present in the lexicon. Also, at inference time, only morphemes in the lexicons receive pseudo-counts. Note that the affixes are generated conditioned on the tag; But the stem are not.[5]

Now, we are ready to generate each word type W, its segmentation S, and its syntactic category T. First, we draw the number of morpheme segments $|S|$ from a geometric distribution truncated to generate at most five morphemes:

$$|S| \sim \text{Truncated-Geometric}(\gamma_{|S|})$$

Next, we pick one of the morphemes to be the stem uniformly at random, and thus determine the number of prefixes and suffixes. Then, we draw the syntactic category T for the word. (Note that T is a latent variable which we recover during inference.)

$$T \sim \text{Multinomial}(\Theta_T)$$

After that, we generate each stem σ_0, prefix σ_-, and suffix σ_+ independently:

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

Token-POS Component This component captures the dependencies between the syntactic categories of adjacent tokens with a first-order HMM. Conditioned on the type-level assignments, we generate segmented tokens $\boldsymbol{s}$ and their POS tags $\boldsymbol{t}$:

$$\begin{aligned} &P(\boldsymbol{s}, \boldsymbol{t}|\boldsymbol{W}, \boldsymbol{T}, \boldsymbol{\theta}) \\ =&\prod_{s_i, t_i} P(t_{i-1}|t_i, \theta_{t|t}) P(s_i|t_i, \theta_{w|t}) \end{aligned}$$

[5] We design the model as such since the dependencies between affixes and the POS tag are much stronger than those between the stems and tags. In our preliminary experiments, when stems are also generated conditioned on the tag, spurious stems are easily created and associated with garbage-collecting tags.

where the parameters of the multinomial distributions are generated by Dirichlet priors:

$$\theta_{t|t} \sim \text{Dirichlet}(\alpha_{t|t}, \{1, \ldots, K\})$$
$$\theta_{w|t} \sim \text{Dirichlet}(\alpha_{w|t}, \boldsymbol{S_t})$$

Here, $\boldsymbol{S_t}$ refers to the set of segmented word types that are generated by tag t. In other words, each segmented word s can only be generated by its tag t specified in the lexicon. This is similar to the one-tag-per-word assumption in our POS induction model in Chapter 2.

Token-Seg Component The component captures the morphological agreement between adjacent segmentations using a first-order Markov chain. The probability of drawing a sequence of segmentations $\boldsymbol{s}$ is given by

$$P(\boldsymbol{s}|\boldsymbol{S}, \boldsymbol{\beta}) = \prod_{(s_{i-1}, s_i)} p(s_i|s_{i-1})$$

For each pair of segmentations s_{i-1} and s_i, we determine: (1) if they should exhibit morpho-syntactic agreement, and (2) if their morphological segmentations are consistent. To answer the first question, we first obtain the final suffix for each of them. Next, we obtain n, the length of the longer suffix. For each segmentation, we define the *ending* to be the last n characters of the word. We then use matching endings as a proxy for morpho-syntactic agreement between the two words. To answer the second question, we use matching final suffixes as a cue for consistent morphological segmentations. To encode the linguistic intuition that words that exhibit morpho-syntactic agreement are likely to be morphological consistent, we define the above probability distribution to be:

$$p(s_i|s_{i-1}) \\ = \begin{cases} \beta_1 & \text{if same endings and same final suffix} \\ \beta_2 & \text{if same endings but different final suffixes} \\ \beta_3 & \text{otherwise (e.g. no suffix)} \end{cases}$$

where $\beta_1 + \beta_2 + \beta_3 = 1$, with $\beta_1 > \beta_3 > \beta_2$. By setting β_1 to a high value, we encourage adjacent tokens that are likely to exhibit morpho-syntactic agreement to have the same final suffix. And by setting $\beta_3 > \beta_2$, we also discourage adjacent tokens with the same endings to be segmented differently. This component can also be view as a correction factor in a HMM that over-generates tokens: [6]

$$P(\boldsymbol{s}|\boldsymbol{t}) = \prod p(s_i|s_{i-1}) \cdot p(t_i|t_{i-1})p(s_i|t_i).$$

3.3.4 Model Variants

Starting with the basic model, we augment more components to form a series of models with increasing sophistication:

(1) **Model 1** (BASIC): This basic model includes the lexicon and the segmentation component. The segmentation model does not generate latent POS tags (or equivalently sets the number of states to one). Affixes are therefore independently generated.

(2) **Model 2** (+POS): This model is the same as Model 1 with the exception that the number of latent states is greater than one. We capture compatible affixes within syntactic categories with this model. Note that we do not assume POS tags are annotated to perform word segmentation. At this point, our models only require a list of word types.

[6] Although p sums to one, it makes the model deficient since, conditioned everything already generated, it places some probability mass on invalid segmentation sequences.

(3) **Model 3** (+Token-POS): This model includes the Token-POS HMM component, and so the model is able to exploit context in a corpus of sentences.

(4) **Model 4** (+Token-Seg): This models includes the factor that captures morphological markers for adjacent words that participate in grammatical agreement.

3.4 Inference

Given a corpus of unsegmented and unannotated word tokens $\boldsymbol{w}$, the objective is to recover values of all latent variables, including the segmentations $\boldsymbol{s}$.

$$P(\boldsymbol{s}, \boldsymbol{t}, \boldsymbol{S}, \boldsymbol{T}, \boldsymbol{L} | \boldsymbol{w}, \boldsymbol{W}, \boldsymbol{\gamma}, \boldsymbol{\alpha}, \boldsymbol{\beta}) \propto \int P(\boldsymbol{w}, \boldsymbol{s}, \boldsymbol{t}, \boldsymbol{W}, \boldsymbol{S}, \boldsymbol{T}, \boldsymbol{L}, \boldsymbol{\Theta}, \boldsymbol{\theta} | \boldsymbol{\gamma}, \boldsymbol{\alpha}, \boldsymbol{\beta}) d\boldsymbol{\Theta} d\boldsymbol{\theta}$$

We want to sample from the above distribution using collapsed Gibbs sampling ($\boldsymbol{\Theta}$ and $\boldsymbol{\theta}$ integrated out.) In each iteration, we loop over each word type W_i and sample the following latent variables: its tag T_i, its segmentation S_i, the segmentations and tags for all of its token occurrences $(\boldsymbol{s}_i, \boldsymbol{t}_i)$, and also the morpheme lexicons $\boldsymbol{L}$:

$$P(\boldsymbol{L}, T_i, S_i, \boldsymbol{s}_i, \boldsymbol{t}_i | \boldsymbol{s}_{-i}, \boldsymbol{t}_{-i}, \boldsymbol{S}_{-i}, \boldsymbol{T}_{-i}, \boldsymbol{w}_{-i}, \boldsymbol{W}_{-i}, \boldsymbol{\gamma}, \boldsymbol{\alpha}, \boldsymbol{\beta})$$

such that the type and token-level assignments are consistent, i.e. for all $t \in \boldsymbol{t}_i$ we have $t = T_i$, and for all $s \in \boldsymbol{s}_i$ we have $s = S_i$.

3.4.1 Approximate Inference

Naively sampling the lexicons $\boldsymbol{L}$ is computationally infeasible since their sizes are unbounded. Therefore, we employ an approximation which turns is similar to performing inference with a Dirichlet Process segmentation model. In our approximation scheme, for each possible segmentation and tag hypothesis $(T_i, S_i, \boldsymbol{s}_i, \boldsymbol{t}_i)$, we only consider one possible value for $\boldsymbol{L}$, which we denote the *minimal lexicons*. Hence, the total number of hypothesis that we have to consider is only as large as the number of possibilities for $(T_i, S_i, \boldsymbol{s}_i, \boldsymbol{t}_i)$.

Specifically, we recover the minimal lexicons as follows: for each segmentation and tag hypothesis, we determine the set of distinct affix and stem types in the whole corpus, including the morphemes introduced by segmentation hypothesis under consideration. This set of lexicons, which we call the minimal lexicons, is the most compact ones that are needed to generate all morphemes proposed by the current hypothesis.

Furthermore, we set the number of possible POS tags $K = 5$. [7] For each possible value of the tag, we consider all possible segmentations with at most five segments. We further restrict the stem to have no more than two prefixes or suffixes and also that the stem cannot be shorter than the affixes. This further restricts the space of segmentation and tag hypotheses, and hence makes the inference tractable.

3.4.2 Sampling equations

Suppose we are considering the hypothesis with segmentation S and POS tag T for word type W_i. Let $\boldsymbol{L} = (L^*, L_-, L_0, L_+)$ be the minimal lexicons for this hypothesis (S, T). We sample the hypothesis $(S, T, s = S, t = T, \boldsymbol{L})$ proportional to the product of the following four equations.

Lexicon Component

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

This is a product of geometric distributions involving the length of each morpheme σ and the size of each of the prefix, the stem, and the suffix lexicons (denoted as $|L_-|, |L_0|, |L_+|$ respectively.) Suppose, a new morpheme type σ_0 is introduced as a stem. Relative to a hypothesis that introduces none, this one incurs an additional cost of $(1-\gamma_0)$ and $\gamma_l(1-\gamma_l)^{|\sigma_0|}$. In other words, the hypothesis is penalized for increasing the stem lexicon size and generating a new morpheme of length $|\sigma_0|$. In

[7] We find that increasing K to 10 does not yield improvement.

this way, the first and second terms play a role similar to the concentration parameter and base distribution in a DP-based model.

Segmentation Component

$$\begin{multline}\frac{\gamma_{|S|}(1-\gamma_{|S|})^{|S|}}{\sum_{j=0}^{5}\gamma_{|S|}(1-\gamma_{|S|})^{j}}\cdot\frac{n_T^{-i}+\alpha}{N^{-i}+\alpha K}\cdot\frac{n_{\sigma_0}^{-i}+\alpha_0}{N_0^{-i}+\alpha_0|L_0|}\cdot\frac{n_{\sigma_-|T}^{-i}+\alpha_-}{N_{-|T}^{-i}+\alpha_-|L_-|}\\ \cdot\frac{n_{\sigma_+|T}^{-i}+\alpha_+}{N_{+|T}^{-i}+\alpha_+|L_+|}. \tag{3.1}\end{multline}$$

The first factor is the truncated geometric distribution of the number of segmentations $|S|$, and the second factor is the probability of generate the tag T. The rest are the probabilities of generating the stem σ_0, the prefix σ_-, and the suffix σ_+ (where the parameters of the multinomial distribution collapsed out). n_T^{-1} is the number of word types with tag T and N^{-i} is the total number of word types. $n_{\sigma_-|T}^{-i}$ refers to the number of times prefix σ_- is seen in all word types that are tagged with T, and $N_{-|T}^{-i}$ is the total number of prefixes in all word types that has tag T. All counts exclude the word type W_i whose segmentation we are sampling. If there is another prefix, $N_{-|T}^{-i}$ is incremented (and also $n_{\sigma_-|T}^{-i}$ if the second prefix is the same as the first one.) Integrating out the parameters introduces dependencies between prefixes. The rest of the notations read analogously.

Token-POS Component

$$\frac{{\alpha_{w|t}}^{(m^i)}}{(M_t^{-i}+\alpha_{w|t}|\boldsymbol{S_t}|)^{(m^i)}}\cdot\prod_{t=1}^{K}\prod_{t'=1}^{K}\frac{(m_{t'|t}^{-i}+\alpha_{t|t})^{(m_{t'|t}^{i})}}{(M_t^{-i}+\alpha_{t|t})^{(m_{t'|t}^{i})}}. \tag{3.2}$$

The two terms are the token-level emission and transition probabilities with parameters integrated out. The integration induces dependences between all token occurrences of word type W which results in ascending factorials defined as $\alpha^{(m)} = \alpha(\alpha+1)\cdots(\alpha+m-1)$ [78]. M_t^{-i} is the number of tokens that have POS tag t, m^i is the number of tokens w_i, and $m_{t'|t}^{-i}$ is the number of tokens t-to-t' transitions. (Both exclude counts contributed by tokens belong to word type W_i.) $|\boldsymbol{S_t}|$ is the number of segmented word types with tag t.

Token-Seg Component

$$\beta_1^{m^i_{\beta_1}}\beta_2^{m^i_{\beta_2}}\beta_3^{m^i_{\beta_3}}. \tag{3.3}$$

Here, $m^i_{\beta_1}$ refers to the number of transitions involving token occurrences of word type W_i that exhibit morphological agreement. This does not result in ascending factorials since the parameters of transition probabilities are fixed and not generated from Dirichlet priors, and so are not integrated out.

3.4.3 Staged Training

Although the Gibbs sampler mixes regardless of the initial state in theory, good initialization heuristics often speed up convergence in practice. We therefore train a series of models of increasing complexity (see section 3.6 for more details), each with 50 iterations of Gibbs sampling, and use the output of the preceding model to initialize the subsequent model. The initial model is initialized such that all words are not segmented. When POS tags are first introduced, they are initialized uniformly at random.

3.5 Experimental Setup

Performance metrics To enable comparison with previous approaches, we adopt the evaluation set-up of Poon et al. [101]. They evaluate segmentation accuracy on a per token basis, using recall, precision and F1-score computed on segmentation points. We also follow a transductive testing scenario where the same (unlabeled) data is used for both training and testing the model.

Data set We evaluate segmentation performance on the Penn Arabic Treebank (ATB).[8] It consists of about 4,500 sentences of modern Arabic obtained from newswire articles. Following the preprocessing procedures of Poon et al. [101] that exclude certain word types (such as abbreviations and digits), we obtain a corpus of 120,000 tokens and 20,000 word types. Since our full model operates over sentences, we train the model on the entire ATB, but evaluate on the exact portion used by Poon et al. [101].

Pre-defined tunable parameters and testing regime In all our experiments, we set $\gamma_l = \frac{1}{2}$ (for length of morpheme types) and $\gamma_{|S|} = \frac{1}{2}$ (for number of morpheme segments of each word.) To encourage a small set of affix types relative to stem types, we set $\gamma_- = \gamma_+ = \frac{1}{1.1}$ (for sizes of the affix lexicons) and $\gamma_0 = \frac{1}{10{,}000}$ (for size of the stem lexicon.) We employ a sparse Dirichlet prior for the type-level models (for morphemes and POS tag) by setting $\alpha = 0.1$. For the token-level models, we set hyperparameters for Dirichlet priors $\alpha_{w|t} = 10^{-5}$ (for unsegmented tokens) and $\alpha_{t|t} = 1.0$ (for POS tags transition.) To encourage adjacent words that exhibit morphological agreement to have the same final suffix, we set $\beta_1 = 0.6, \beta_2 = 0.1, \beta_1 = 0.3$.

In all the experiments, we perform five runs using different random seeds and report the mean score and the standard deviation.

Baselines Our primary comparison is against the morphological segmenter of Poon et al. [101] which yields the best published results on the ATB corpus. In addition, we compare against the Morfessor Categories-MAP system [26]. Similar to our model, their system uses latent variables to induce clustering over morphemes. The difference is in the nature of the clustering: the Morfessor algorithm associates a latent variable for each morpheme, grouping morphemes into four broad categories (prefix, stem, suffix, and non-morpheme) but not introducing dependencies between affixes directly. For both systems, we quote their performance reported by Poon et al. [101].

[8] Our evaluation does not include the Hebrew and Arabic Bible datasets [101, 116] since these corpora consists of short phrases that omit sentence context.

Model	R	P	F1	t-test
PCT 09	69.2	88.5	77.7	-
Morfessor	72.6	77.4	74.9	-
Model 1 (Basic)	71.4	86.7	78.3 (2.9)	-
Model 2 (+POS)	75.4	87.4	81.0 (1.5)	+
Model 3 (+Token-POS)	75.7	88.5	81.6 (0.7)	∼
Model 4 (**+Token-Seg**)	**82.1**	**90.8**	**86.2 (0.4)**	++

Table 3.3: Results on the Arabic Treebank (ATB) data set: We compare our models against Poon et al. [101] (PCT09) and the Morfessor system (Morfessor-CAT). For our full model (+Token-Seg) and its simplifications (Basic, +POS, +Token-POS), we perform five random restarts and show the mean scores. The sample standard deviations are shown in brackets. The last column shows results of a paired t-test against the preceding model: ++ (significant at 1%), + (significant at 5%), ∼ (not significant), - (test not applicable).

3.6 Results

Comparison with the baselines Table 3.3 shows that our full model (denoted +Token-Seg) yields a mean F1-score of 86.2, compared to 77.7 and 74.9 obtained by the baselines. This performance gap corresponds to an error reduction of 38.1% over the best published results.

Ablation Analysis To assess relative impact of various components, we consider several simplified variants of the model:

- Model 1 (Basic) is the type-based segmentation model that is solely driven by the lexicon.[9]
- Model 2 (+POS) adds latent variables but does not capture transitions and agreement constraints.
- Model 3 (+Token-POS) is equivalent to the full model except that it does not incorporate agreement constraints.

Our results in Table 3.3 clearly demonstrate that modeling morpho-syntactic constraints greatly improves the accuracy of morphological segmentation.

[9] The resulting model is similar in spirit to the unigram DP-based segmenter [42, 116, 117].

Model	Token		Type	
	F1	Acc.	F1	Acc.
1 (Basic)	68.3	13.9	73.8	24.3
2 (+POS)	75.4	26.4	78.5	38.0
3 (+Token-POS)	76.5	34.9	82.0	49.6
4 (+Token-Seg)	84.0	49.5	85.4	57.7

Table 3.4: Segmentation performance on a subset of words that have compatible affixes. These words begin with prefix "Al" (determiner) and end with suffix "At" (plural noun suffix). The mean F1 scores are computed using all boundaries of words in this set. For each word, we also determine if both affixes are recovered while ignoring any other boundaries between them. The other two columns report this accuracy at both the type-level and the token-level.

We further examine the performance gains arising from improvements due to (1) encouraging morphological consistency within syntactic categories, and (2) morphological realization of grammatical agreement.

We evaluate our models on a subset of words that exhibit morphological consistency. Table 3.4 shows the accuracies for words that begin with the prefix "Al" (determiner) and end with a suffix "At" (plural noun suffix.) An example is the word "Al–{ntxAb–At" which is translated as "the-election-s". Such words make up about 1% of tokens used for evaluation, and the two affix boundaries constitute about 3% of the all gold segmentation points. By introducing a latent variable to capture dependencies between affixes, +POS is able to improve segmentation performance over Basic. When dependencies between latent variables are introduced, +Token-POS yields additional improvements.

We also examine the performance gains due to morphological realization of grammatical agreement. We select the set of tokens that share the same final suffix as the preceding token, such as the bigram "Al–Df–p Al–grby–p" (which is translated word-for-word as "the-bank the-west") where the last morpheme "p" is a feminine singular noun suffix. This subset makes up about 4% of the evaluation set, and the boundaries of the final suffixes take up about 5% of the total gold segmentation boundaries. Table 3.5 reveals this category of errors persisted until the final component (+Token-Seg) was introduced.

Model	Token		Type	
	F1	Acc.	F1	Acc.
1 (Basic)	85.6	70.6	79.5	58.6
2 (+POS)	87.6	76.4	82.3	66.3
3 (+Token-POS)	87.5	75.2	82.2	65.3
4 (+Token-Seg)	92.8	91.1	88.9	84.4

Table 3.5: Segmentation performance on a subset of words that have morphological markers to indicate grammatical agreement. These words have the same final suffix as their preceding words. The F1 scores are computed based on all boundaries within the words, but the accuracies are obtained using only the final suffixes.

3.7 Conclusion

Although the connection between syntactic (POS) categories and morphological structure is well-known, this relation is rarely exploited to improve morphological segmentation performance. The performance gains motivate further investigation into morpho-syntactic models for unsupervised language analysis.

Chapter 4

Unsupervised Morphological Segmentation for Dialectal Arabic Machine Translation

4.1 Introduction

Morphological analyzers are increasingly used in modern machine translation systems developed for inflectional languages. In such languages, the presence of affixes greatly expands the basic vocabulary, giving rise to severe sparsity problems. When manually segmented data is available, a solution of choice is to employ a supervised morphological analyzer. However, when annotations are lacking the only option is to rely on an unsupervised segmenter.

In this chapter, we explore how to effectively utilize unsupervised morphological segmenters to improve machine translation. Today, the development of these analyzers is driven by metrics that compare system output to gold-standard segmentation. To effectively utilize their output in MT, we need to determine whether these metrics are predictive of MT performance. Our analysis shows that the two measures do not necessarily correlate – not all mistakes of morphological analyzers have equal importance in the context of machine translation. In fact, MT systems can robustly tolerate certain classes of errors that are systematically produced by unsupervised models. This ultimately enables unsupervised systems to compare favorably with their supervised counterparts.

Another consideration when applying morphological analyzers to MT is the issue of scalability, which is required to handle the large datasets that are typically used to train MT systems. One would expect that an unsupervised segmenter would increasingly improve in performance as more data is provided to it. In practice, however, an increase in dataset size leads to an explosion in the number of unique affixes induced by the model. This phenomenon is particularly acute for non-parametric models, where the size of the induced vocabulary increases with the size of the dataset. Another scaling-related issue is the high variability in segmentation output that results from commonly-used stochastic sampling. This variability negatively impacts the stability of MT systems. In this chapter, we demonstrate that using maximum marginal decoding reduces these phenomena, thereby improving segmenter performance on large datasets.

We explore the usefulness of unsupervised morphology in the context of a state-of-the-art Arabic-to-English MT system. A crucial advantage of Arabic for our study is the availability of high-quality supervised morphological analyzers, such as MADA [46] and Sakhr,[1] as valuable benchmarks for performance comparison. The experiments are conducted on both MSA and dialectal Arabic data. We demonstrate that the unsupervised system rivals and sometimes even outperforms MADA on the NIST MT-08 evaluation corpus. Moreover, on a Levantine dialectal Arabic corpus, our system outperforms both MADA and Sakhr, which were developed for MSA. Finally, the morphological analyzer presented in this chapter achieves the best reported segmentation results on the Arabic Treebank benchmark.

[1] http://www.sakhr.com/Default.aspx

4.2 Related Work

Machine translation systems that process highly inflectional languages often incorporate morphological analysis as part of their system architecture. Some of these approaches rely on morphological analysis for pre- and post-processing, while others modify the core of a translation system to incorporate morphological information [45, 79, 88, 123]. For instance, factored translation models [4, 69, 134] operate at the level of unsegmented words, but they parameterize phrase translation probabilities as factors that encode morphological features. Other approaches translate at the word level, but correct translation outputs for morphological rich target languages [86, 127].

The approach we have taken in this chapter is an instance of a segmented MT model, which divides the input into morphemes and uses the derived morphemes as a unit of translation [5, 21, 29, 41, 98, 102, 109]. This is a mainstream architecture that shown to be effective when translating from a morphologically rich language.

Most of the existing systems assume access to gold morphological analysis [41, 76], or employ a supervised or knowledge-based morphological analyzer [29, 33, 88, 109]. A number of recent approaches explored the use of unsupervised morphological analyzers [21, 26, 129]. Virpioja et al. [129] apply unsupervised morphological segmenter Morfessor [26], and apply an existing MT system at the level of morphemes. The system does not outperform the word baseline partially due to the insufficient accuracy of automatic morphological analyzer. The system of Clifton and Sarkar [21] also uses Morfessor output but in a different translation architecture – it pre-processes target-side training data by segmenting words into morphemes then stitching the decoded output morphemes to form coherent words. This system accounts for noise in segmentation by adding a filtering component that eliminates extraneous affixes. In addition, the use of a probabilistic model for combining morphemes further alleviates Morfessor deficiencies. Under these conditions, the unsupervised morphological segmenter gives better MT performance than a knowledge-based one[2]. However, the

[2] http://gna.org/projects/omorfi

segmentation performance of the knowledge-based segmenter is not evaluated, so it remains inconclusive how segmentation quality correlates with MT quality. In this chapter, we provide a broader analysis on how segmentation quality impacts MT performance.

The work of Mermer and Akın [84] and Mermer and Saraclar [85] attempts to integrate morphology and MT more closely than we do, by incorporating bilingual alignment probabilities into a Gibbs-sampled version of Morfessor for Turkish-to-English MT. However, the bilingual strategy shows no gain over the monolingual version, and neither version is competitive for MT with a supervised Turkish morphological segmenter [97]. In contrast, the unsupervised analyzer we report on here yields MSA-to-English MT performance that equals or exceed the performance obtained with a leading supervised MSA segmenter, MADA [46].

4.3 Scaling An Unsupervised Segmenter to Large Datasets

It has been widely documented in the MT community that increase in the size of training data inevitably leads to performance gains. Ideally, we would see a similar trend for unsupervised morphological analyzers which could now utilize large amounts of raw data available for training MT systems. Surprisingly, an opposite trend is observed in practice. For instance, the developers of the unsupervised Morfessor segmenter [26] observe that performance on English decreases once the data set exceeds a limit. More recently in MT research, to eliminate this noise, Clifton and Sarkar [21] train the system on 5000 most frequent words, and segmented the rest of their Finnish corpus using the vocabulary derived from this subset. Similarly as we show in Section 4.5, the increase in dataset size results in six-fold increase in the number of affixes. Most of these newly derived morphemes do not constitute valid linguistic units.

Another undesirable side effect of scaling to large datasets is high variability in system outputs. Section 4.5 documents the severity of this phenomenon. This variability is inherent to sampling inference procedures commonly employed by unsupervised probabilistic segmenters. This variance is particularly harmful in the MT context as it would result in unstable MT performance across multiple runs. In section 4.3.2, we describe Maximum Marginal Decoding that addresses both the issue of variability and affix overgeneration.

We discuss the issues of scalability in the framework of our unsupervised segmenter [75]. Not only this algorithm achieves state-of-the-art results on Arabic, but it also belongs to a broad class of non-parametric Bayesian segmenters which are likely to exhibit similar trends [42, 117]. Moreover, the code of the system is publicly available which made it easy to reproduce the results and expand the model. We start this section by briefly summarizing our segmenter [75].

4.3.1 Review of Morphological Segmenter

Our segmenter [75] is implemented in a probabilistic model that operates at the type level. The model posits that each word consists of few morpheme segments drawn from latent prefix, suffix, and stem lexicons. Below we describe the four variants of the model.

The basic variant, **Model 1** (Lexicon), prefers small affix lexicons and assumes that morphemes are drawn independently. **Model 2** (POS) enriches the above model by generating a latent syntactic category for each word type on which its affixes are conditioned. This latent variable thus introduces dependencies between affixes and encourages compatible affixes to be generated together, for instance prefix “Al” (determiner) and suffix “At” (feminine plural noun marker).

Model 3 (Token-POS) extends the previous model by incorporating token-level contextual information. After generating word types, word tokens are generated with a type-level hidden Markov model (HMM). Each hidden variable in the HMM represents the syntactic category of a token which are now dependent on its surrounding context. **Model 4** (Token-Seg) further enriches the model by modeling morphosyntactic agreement by employing a transition probability distribution that favors adjacent tokens with the same endings to also have the same final suffix. For example, the two words in the bigram "AlDfp Algrbyp" (the-bank the-west) are encouraged to have "p" (feminine singular noun marker) as their final suffixes.are summarized

4.3.2 Maximum Marginal Decoding

To resolve the scalability issues described above, we employ a voting technique that is based on Maximum Marginal Decoding [61]. This method marginalizes out the other latent variables (POS tags and other word segmentations), to obtain a marginal distribution over the segmentations of the current word, from which the mode is selected. This procedure can be approximated by marginalizing over a set of independent samples. In our experiments, we draw each sample from a separate run of the segmenter, and allow them to vote on the word segmentations. [3]

An interesting perspective on the decoding method is that it is analogous to system combination, which has been shown to produce significant benefits for MT and ASR in the past [34, 106]. While the "systems" that are being combined here come from the same generative source, they are nevertheless substantially different from one another in terms of the outputs they produce. By employing voting, this diversity of output, which would otherwise be problematic, is turned into a benefit.

The decoding method also offers several important benefits. First, It dramatically increases the stability of the segmentation output, yielding rates of segmentation agreement that are over 95% at both the word-type and word-token levels. Second, it improves segmentation accuracy over even an oracle that selects the best-performing run of the ones being combined. Third, it cuts down on the affix over-generation described earlier. And lastly, it gives better MT performance on the segmented text. All of these advantages will be described in more detail in Section 4.5.

[3] We also experimented with decoding over 25 samples that were drawn from the same run rather than different ones (take 10 iterations apart for independence) but they gave reduced gains.

4.4 Research Questions

Our investigation is driven by the following questions:

- What is the correlation between gold-standard segmentation evaluation and MT performance?
- What is the effect of scaling to large datasets on the performance of unsupervised morphological systems and the resulting impact on MT?
- How useful is an unsupervised segmenter for MT in the low-resource settings?

Below we describe experimental set-up that we use to answer these questions.

4.4.1 Experimental Design

MT System Our experiments were performed using a state-of-the-artnn hierarchical string-to-dependency-tree MT system of Shen et al. [113]. The hierarchical MT system performs decoding with a 3-gram target LM, generates the N best unique translation hypotheses, and then rescores them, using a large, unpruned 5-gram LM to select the best-scoring translation. Forward-backward and context-free lexical smoothing are used as decoder features. We use GIZA++ [95] for word alignments. The decoder model parameters are tuned using Minimum Error Rate Training (MERT) [94] to maximize the IBM BLEU score [100]. In all conditions, we train our target language model on 3.8 billion words of English data, corresponding to the NIST MT-08 constrained track.

Details on Scaling the Segmenter To handle large data sets at the scale typically for MT, we reimplemented our unsupervised Bayesian morphological segmenter [75] in Java. For computational tractability on the very large (336K-word) lexicon we study, we additionally restrict the system to consider at most one suffix per word. To implement maximum marginal decoding, voting is applied individually on each model level by combining 25 independent runs for that level. Following our previous approach [75], we use a transductive approach that segments the test and training sets together.

Morphological Analyzers As a point of comparison with automatic segmentation systems, we employ Morfessor [26] which has been used in prior MT work [21, 129]. We also compare against MADA [46], a state-of-the-art supervised system that uses SVMs to select from the analyses provided by the rule-based Buckwalter morphological analyzer [15]. In addition, we compare against Sakhr, a context-dependent rule-based Arabic analyzer. It relies on substantial rule-encoded linguistic knowledge about Arabic vocabulary. In addition, it uses a statistical POS tagger, a parser, a named-entity recognizer, and an Arabic language model. Because the Sakhr analyzer has proven highly effective as a component of our own MT system, we use it as our strongest point of comparison.

Integration of Segmentation and MT We integrate morphology into MT during preprocessing, segmenting words into morphemes, and subsequently treating these morphemes as words for alignment and decoding. This framework is commonly employed in MT and has shown to be effective [5]. For each combination of segmentation system, condition, and training corpus, we estimate separate translation models, by performing Giza++ alignment and MERT training on the appropriately segmented training corpus. The resulting translation model is then used to decode the test set. Our experiments are thus exhaustive.

Training and Test Corpora For MSA, our training corpus is the NIST MT-08 Constrained Data Track Arabic corpus which consists of 35M total words, with a vocabulary of 336K unique Arabic words. To test with reduced training data, we use a 1.3M-word subset of this corpus. Our MSA test set is the MT-08 evaluation set. For dialectal Arabic, we use a Levantine corpus collected from the web, and translated using Mechanical Turk [136]. We use 1.5M words of this corpus for training, and and 18K words for test. The training set has a vocabulary of 160K unique words.

		inflect	*no-inflect*	*neutral*
Sakhr		65.9	86.4	86.8
MADA		70.1	92.2	92.3
Morfessor		74.9	64.5	77.0
Lee GS	M1	80.1	72.8	85.0
	M2	81.4	72.2	85.4
	M3	81.4	71.7	84.8
	M4	86.2	68.7	85.4
Lee MM	M1	81.8	74.6	87.3
	M2	82.0	73.3	86.6
	M3	83.2	73.7	87.4
	M4	88.2	70.6	87.7

Table 4.1: Segmentation accuracy on Arabic Treebank for various segmenters and evaluation metrics.

Performance Metrics For segmentation, we compute recall, precision, and F-measure. For MT, our primary metric is the BLEU score, which is also the metric our system is tuned for. To calculate statistical significance, we use the boot-strap resampling method of Koehn [68], and report confidence levels for the key scoring differentials.

4.5 Results

We first compare the performance of morphological segmenters based on human-annotated segmentations then report their impact in the context of MT. Our analysis of results is driven by research questions we posed in the previous section.

4.5.1 Segmentation Results

In this section, we present our results for segmentation accuracy on the Arabic Treebank (ATB) for all segmenters described in the evaluation setup.

We present three F1-scores due to differences in gold standards adopted in unsupervised vs. supervised evaluation. For instance, the gold standard used in evaluation of unsupervised systems considers “p” (feminine singular noun suffix) as a separate

Input: AlDfp (“the bank”)
Output: Al–Df–p

Metric	Gold	# mistakes
inflect	Al–Df–p	0
no-inflect	Al–Dfp	1
neutral	Al–Dfp	0

Figure 4-1: An example illustrating the three segmentation standards. Inflectional suffix “p” is a feminine singular noun marker which does not have a corresponding a lexical unit in English.

morpheme [101] while the regular ATB annotation[4] which is employed by supervised systems do not [107]. Our first measure *inflect* uses the former gold standard and the second one *no-inflect* uses the latter. The third metric *neutral* provides a more neutral evaluation by using the supervised gold standard but does not penalize systems for prediction contentious affixes. A comparison of the three standards is shown in Figure 4-1.

As expected, unsupervised systems out-perform supervised ones under *inflect*, whereas MADA naturally performs better under *no-inflect*. Under *neutral*, the gap between supervised and unsupervised systems narrows but MADA still out-performs the rest of the segmenters. It is worthwhile to point out that Sakhr underperforms MADA on all three measures because it uses different segmentation conventions than ATB. However, as we shall see in section 4.5.3, it consistently gives better MT scores than MADA.

Table 4.1 also demonstrates that our Bayesian segmenter out-performs the best published results on the ATB [89] — using maximum marginal decoding increases segmentation F-measure from 86.2% to 88.2%.

4.5.2 Analysis of Segmenter Output in Large Data Setting

Now, we apply all the segmenters to the much larger NIST MT-08 Constrained Data Track Arabic corpus, comprising of 35M tokens (compared to 150K tokens in the ATB.) Although we do not have gold segmentations for this data set, we compile several statistics that easily reveal striking differences between supervised and un-

[4] With the exception that determiners “Al” are separated.

supervised systems. An immediate observation is that applying our unsupervised segmenter [75] and Morfessor to large data sets leads to an explosion in the number of unique affixes. This phenomenon reflects the character of non-parametric models, which allows them to adapt to the data without having to specify the number of parameters beforehand [131]. As shown in Table 4.3, our Bayesian segmenter induces merely 41 suffixes in the ATB but the number increases dramatically to 261 suffixes for the NIST MT-08 data set — a linguistically implausible analysis since we expect inflectional suffixes to come from a closed lexicon. We also observe that Morfessor suffers from the same limitation.

Surprisingly, the sheer number of extraneous affixes does not necessarily lead to a negative outcome for MT performance (see section 4.5.3.) To understand why, we compute two statistics that measure the skewness of affix distributions at the token-level. The first statistic *Top-95* simply counts the number of unique morphemes that account for 95% of all prefixes or suffixes. The second statistic *ppl* calculates the perplexity of the distribution. (It assigns a low number to a skewed distribution and a higher one to a more uniform distribution.) The statistics for ATB and the MT-08 data set are presented in Table 4.3. While the huge number of unique affixes may seem daunting, note that a small number of prefixes — merely 7 of them — dominate the token-level distributions. This suggest that the low-frequency extraneous affixes have hardly any impact on the subsequent MT pipeline.

We also investigate the utility of maximum marginal decoding in large data setting. Firstly, we observe that it helps to alleviate the issue of affix explosion — it yields a 28% reduction of prefixes and 21% reduction of suffixes.

Another benefit of this decoding method is that it drastically reduces the variability of the segmentation output of the stochastic segmenter. Table 4.2 compares the output agreement statistics between standard Gibbs sampling and maximum marginal decoding. In 25 separate runs of Gibbs sampling, segmentation boundaries are different for about 25% of the time at the type-level and 15% at the token-level. In contrast, in two separate runs of maximum marginal decoding (each combining 25 independent sub-runs), there is only disagreement of at most 5% at the type level

Decoding	Level	Rec	Prec	F1	Acc
Gibbs sampling	Type	82.9	83.2	83.1	74.5
	Token	87.5	89.1	88.3	86.7
Max marginal	Type	95.9	95.8	95.9	93.9
	Token	97.3	94.0	95.6	95.1

Table 4.2: Comparison of agreement in outputs between random restarts for two decoding methods on the full MT-08 data set: We compute the average segmentation recall, precision, F1-measure, and exact-match accuracy of separate decoding runs with each other.

	ATB	MT-08		
		Gibbs	MM	Morf
Unique prefixes	17	130	93	287
Unique suffixes	41	261	216	241
Top-95 prefixes	7	7	6	6
Top-95 suffixes	14	26	19	19
Prefix ppl	4.2	4.4	4.0	3.75
Suffix ppl	8.1	8.3	7.6	13.15

Table 4.3: Affix statistics of unsupervised segmenters. On the ATB (21K word types), we show statistics for the Bayesian unsupervised segmenter that employs regular Gibbs sampling (Gibbs). On the large MT-08 data set (336K word types), we also use the output of the Bayesian segmenter that employs maximum marginal decoding (MM). In addition, we show statistics for Morfessor.

and at the token level.

4.5.3 Impact of Morphology Analyzers on MT

Table 4.4 summarizes scores of segmenters on various data sets. We observe that Sakhr performs the best for the MSA MT-08 data sets, while the Bayesian unsupervised segmenter performs the best for Levantine dialectal data. To put the performance of our MT system in perspective, at the time of evaluation in 2008, the best performing system achieves a BLEU score of 45.26. [5]

Correlation between segmentation scores and MT performance As these results and those of Table 4.1 illustrate, there is some correlation between segmenta-

[5] http://www.itl.nist.gov/iad/mig/tests/mt/doc/

tion accuracy and MT performance – for example, Morfessor significantly lags behind in both F-measure and BLEU scores. However, the relative rankings of these two measures are not consistent with respect to other systems. For instance, Sakhr has the lowest F-score on MSA Treebank, yet it achieves the best performance on MSA datasets. For all three datasets, our system rivals or out-performs MADA, and on the dialect set, achieves higher performance than Sakhr as well. While MADA and Sakhr were not developed for Levantine, and thus cannot be expected to perform well on it, the fact that our system gives better results is further validation of its usefulness.

Impact of maximum marginal decoding of segmenter on MT Not only does this decoding method alleviates affix explosion and output variability in large data settings, we also see that in most cases, it improves MT performance as well. For both MSA datasets, this method yields superior results, and is what allows our segmenter to out-perform MADA. On dialectal data, it preserves almost the same performance.

Impact on MT in low-resource settings We see that the contribution of segmentation to MT performance varies across datasets, with relative gains of 7%, 14%, and 18%, for MSA full training, MSA partial training, and dialect, respectively. As we would expect, the impact increases in low-data settings, consistent with previous work [47]. Specifically, for the larger MSA dataset, our system recoups 79% of this gain and for the smaller one, 93%.

Impact of inflectional morphology There is an interesting connection between the relative performance of the four variants of the unsupervised models in terms of F-measure and BLEU-scores. While Model 4 out-performs other models in terms of F-score, it is Model 3 that gives a higher BLEU score. This might be explained from the observation that Model 4 induces more inflectional suffixes that do not correspond to any lexical unit on the English side, such as the feminine singular noun suffix marker “p” (as opposed to lexical morphemes like determiner “Al” which aligns to “the” in English) Inspection of the differences between the output of Model 3 and Model 4 showed that this was indeed the case.

System		MSA Small	MSA Full	Lev Dial
Unsegmented		38.69	43.45	17.10
Sakhr		**43.99**	**46.51**	19.60
MADA		43.23	45.64	19.29
Morfessor		42.07	44.71	18.38
Lee GS	M1	43.12	44.80	19.70
	M2	43.16	45.45	**20.15+**
	M3	43.07	44.82	19.97
	M4	42.93	45.06	19.55
Lee MM	M1	43.53	45.14	19.75
	M2	43.45	45.29	19.75
	M3	**43.64+**	**45.84**	**20.09**
	M4	43.56	45.16	19.93

Table 4.4: BLEU scores for all the systems: MSA Small gives results on a 1.3M-word subset of the MT-08 training corpus whereas MSA Full shows results on the full training corpus. Lev Dial gives the results for our Levantine dialectal Arabic data. "Lee MM" is that version of our segmenter that uses maximum marginal decoding. M1-M4 are the different model levels of our segmenter. For each data sets, the overall best score and the highest score for our segmenter are shown in bold. A "+" indicates a statistically significant difference between our segmenter and MADA.

To further confirm this hypothesis, we deterministically restricted our segmenter to only consider affixes in MADA's list of affixes. This did not give an overall improvement, perhaps because the knowledge was imposed as hard constraint rather than probabilistically. Unlike most of the experiments, Model 3 scored was out-scored by Model 4, giving only 45.07 to Model 4's 45.52. Unlike most of the experiments, however, Model 4 outscored Model 3, 45.52 to 45.07. It would appear that once the problem of the over-segmentation was alleviated, Model 4 actually was able to improve the MT performance.

Analysis of performance on Levantine Arabic dialect To understand the performance gains of using our unsupervised segmenter on the Levantine data, the differences between the segmenter and MADA outputs were inspected by a native Levantine speaker. We find that the unsupervised segmenter is able to discover Levantine affixes not present in MSA, and Table 4.5 shows the ones that occurs frequently.

Our analysis reveals two major differences between Levantine and MSA. First, some affixes are written differently in Levantine. Affixes are sometimes abbreviated – for instance, the MSA prefix "ElY" (which means "on") is abbreviated as "E" in Levantine – or even written in a different way. For example, the MSA masculine plural marker "m" is written as "n" in Levantine. The second difference is that a sequence of affixes in MSA is sometimes represented as one affix in Levantine. For example, the prefix sequence in MSA "h*A-Al" (which translates to "this-the") is written as a single prefix "hAl" in Levantine.

4.6 Conclusion

In this chapter we investigate the role of unsupervised morphological segmentation in the context of Arabic-to-English machine translation. We evaluate several morphological analyzers based on segmentation performance as well as their impact on translation quality. Our analysis show that there is some correlation between the two metrics but relative rankings according to segmentation performance is not predictive

Affix	**Freq**	**MSA**	**Gloss**
hAl+	1076	h*A-Al+	“this-the”
E+	927	ElY	“on”
H+	611	s+	future tense
by+	611	y+	present habitual
Em+	179	y+	present continuous
EAl+	541	ElY-Al+	“on-the”
+hn	185	+hm	“them”/“their”
+kn	541	+km	plural “you”/“your”

Table 4.5: Levantine affixes induced by unsupervised segmenter that are not captured by supervised MSA segmenters. Sequences of morphemes are separated by “-”. To distinguish prefixes from suffixes, we append “+” to the former.

of relative MT quality. To apply unsupervised segmenter to the data scale on which MT systems operate, we identify challenges and propose effective solutions. Specifically, we demonstrate that employing maximal marginal decoding (which can be easily implemented by means of voting) reduces variability in segmentation output, helps to contain affix lexicon explosion, surpasses the state-of-the-art segmentation performance on standard Arabic Treebank data set, and ultimately gives MT scores competitive with supervised segmenters. Finally, we demonstrate the utility of unsupervised segmenter in the context of low-resource machine translation. In particular, we show that without any specific adaptation our unsupervised segmenter yields significant improvement gains when applied to the Levantine Arabic dialect.

Chapter 5

Conclusion

In this thesis, we present develop models for unsupervised part-of-speech (POS) induction and morphological word segmentation. A common theme is that both models combines both type and token level cues to improve performance. Both models explain how the observed unsegmented untagged corpus is generated from a latent lexicon with which we encode linguistic intuitions.

We model three type-level properties in our generative POS induction model. First, although words need to be disambiguation in context, words tend to take one predominant tag in any given corpus. Second, POS tags at the type-level has a different distribution from the token-level. Third, words show orthographic features that correlated with their syntactic category. We realize these intuitions by enforcing each word type to take only one POS in the latent lexicon. Moreover, there is a type-level tag distribution that is not present in a standard hidden Markov model (HMM). Each word in the lexicon also generates a bag of features conditioned on the already generated POS tag. These properties of lexicon tames unsupervised inference of POS tags and bias them towards linguistically plausible assignments. In combination with Monte Carlo Markov Chain inference algorithms developed for Bayesian graphical models, our POS model improves performance for a wide variety of languages.

In our morphological word segmentation model, we exploit the connection between morphology and POS to improve performance. Specifically, we model the fact that morphemes are correlated within and across words that participate in grammar agreement. First, morphological affixes are consistent within the same syntactic class. Second, adjacent words in grammatical agreement have compatible suffixes. To model the first phenomenon, we generate affixes of a word conditioned on its latent POS tag. After the lexicon is generated, the HMM generates words using the lexicon and also encourages adjacent words that participate in grammatical agreement to have the same suffix. Evaluation on the Arabic Treebank (ATB) show that modeling these two properties bring substantial gains to segmentation performance.

Apart from improving computational models of morphology and POS, we also showcase the utility of unsupervised word segmentation technology in an end-to-end natural language processing application. We demonstrate that unsupervised models are crucial to the success of machine translation (MT) for low-resource languages, particularly the Levantine Arabic dialect. To incorporate segmentations to a state-of-the-art string-to-dependency-tree MT system, we segment the Levantine corpus before feeding it to the MT training and decoding pipeline. The primary finding is that using our segmentation model outperforms supervised and knowledge-based alternatives developed for the Modern Standard Arabic. Apart from results in MT, we also find that employing maximum marginal decoding which reduces the number of spurious morphemes. Experiments show that this decoding method improves our previous segmentation results on the ATB.

5.1 Discussion and Future Work

Relaxing one-tag-per-word constraint There are opportunities to enrich the lexicon model, which is a key component that allows us to bridge type-level and token-level cues. We have only explored a concise lexicon which encodes direct dependencies between a word's orthographic form and its single POS tag. One approach is to relax the one-tag-per-word assumption so that higher-order morpho-syntactic dependencies involving ambiguity classes [32, 126] of tags can be modeled. However, we caution against increasing the expressiveness of the model without encoding more linguistic knowledge to constraint unsupervised learning. For the task of POS induction, we

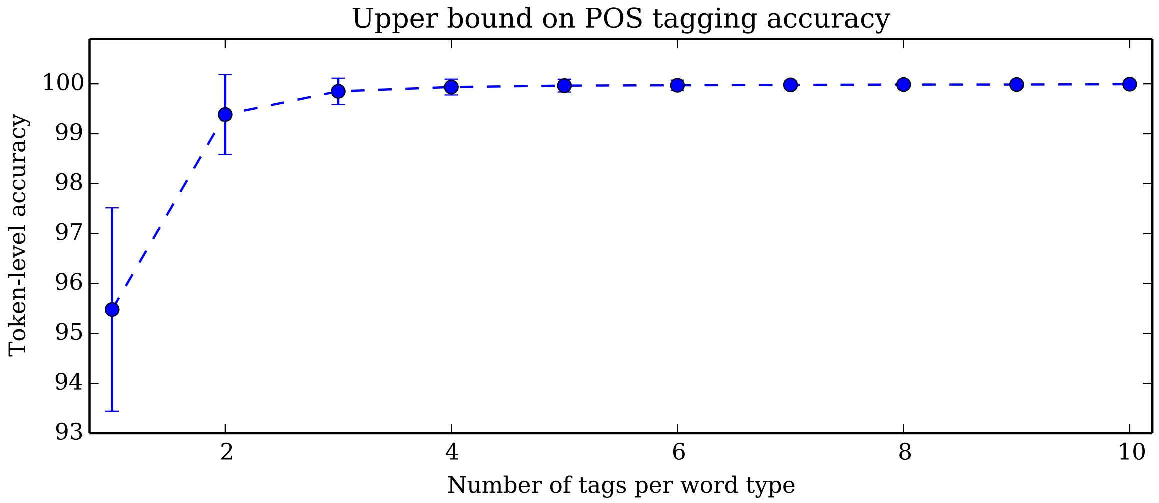


Figure 5-1: Upper bound on POS tagging: We plot the the upper bound on POS tagging (averaged over multiple languages) against k, the number of tags for each word type. For each language, the upper bound is obtained by computing the proportion of tokens that are assigned the k most frequent tags. The languages and data sets are detailed in Section 2.5.1.

find that the returns to using more number of tags per word type are marginal (see Figure 5-1), and the correlation between performance of our full model and the upper bound imposed by the one-tag-per-word lexicon is weak (see Table 5.1).

Deeper connections between Syntax and Morphology Another direction is to model more complex interactions between syntax and morphology that builds up towards an unified unsupervised model of morpho-syntactic induction. Table 1-1 illustrates irregularities of the segmentation representation that are captured by other models of morphology. For instance, the word-and-paradigm model relates correlations among grammatical features of a word (such as number, gender, case, and tense) with possibly non-concatenative but systematic alterations of the base word form. This framework expresses rich regularities that span across morphological related words (in contrast to just correlations between the orthographic form of a word and its single POS tags or POS tag ambiguity class). Another interesting framework is the distributed morphology theory of Halle and Marantz [50] whose basic tenet of is to eliminate the need for a traditional lexicon which list the surface forms of

Language	Our Model	Upper bound
English	74.6	94.6
Arabic	62.1	95.1
Bulgarian	73.1	97.9
Chinese	66.3	92.9
Czech	65.1	99.2
Danish	72.2	96.3
Dutch	69.0	96.6
German	74.9	95.5
Japanese	79.9	94.0
Portuguese	75.3	95.5
Slovene	64.2	98.5
Spanish	74.2	95.4
Swedish	68.4	93.3
Turkish	59.9	91.9

Table 5.1: POS induction performance and accuracy upper bound: For each language we show the many-to-one accuracy of our full model and the upper bound imposed by one-tag-per-word constraint. The correlation between the two series is 0.036.

morphemes and words. The role of the traditional lexicon is distributed into other components that operate on syntactic, morphological, and semantic features of items that are eventually spell-out phonologically or orthographically to form sentences and words.

Quantifying Generalization of Unsuperivsed Models with Output Sparsity

As we enrich unsupervised models with more intricate relationships between syntax and morphology, the need for more sophisticate methods to encourage sparse representations becomes crucial. The supervised framework uses labeling or regression errors as a metric for measuring the ability of a model to generalize to new examples. The unsupervised framework can use the ability to produce sparse outputs on novel examples as a measure of generalizability. The posterior regularization framework now operates in a transductive fashion where the unlabled test data is used for unsupervised learning. This framework can potentially be extended so that it also decodes unseen examples such that the latent structures it produces satisfy pre-defined constraints with high probability.

Application-driven Discovery of Linguistic Structures Apart improving unsupervised models by incorporating insights, an orthogonal direction is to allow applications to uncover language structures. Our experiments in Section 4.5 reveals that segmentation accuracy and machine translation quality is loosely correlated. It is still an open question as to what morphological segmentation annotation scheme is optimal for a machine translation system. Current research specifies annotation criteria before application requirements are drawn up. A fundamental question of language understanding is how the representation of linguistic knowledge interacts with end-to-end applications, and if applications can discover linguistic structures would otherwise remain unknown.

Appendix A

Qualitative Comparision of POS Induction Model Variants

Language	Top 5 (type-level)	Bottom 5 (type-level)
Arabic	N A VI X VP	G FN FI I -
	N X P C A	P C A VP S
	N A P VP C	P VP C S X
	N A P VP C	P VP C X S
Bulgarian	Nc Vpp Vpi Np Am	Vii Nm Tg P V
	Nc Vpp R A Af	Vpi Ps Cs Cp Tx
	Nc Vpp A R Np	Ps Punct Cp Pp Tx
	Nc Vpi Np Vpp Am	Cs Vyp Ps Tn Tx
Chinese	N V DM D Ne	DE I Head H11 Str
	N V DM D DE	N V DM D DE
	N V D DE	N V D DE
	N V D DE	N V D DE
Czech	N A V C D	J T I X Z
	N A Z V R	A Z V R J
	N A V Z R	A V Z R J
	N V A C R	A C R J Z
Danish	NC VA AN NP RG	XP XR U XS PC
	NC AN RG VA PP	AC PI PD CC CS
	NC AN VA RG PP	CC SP PI PD CS
	NC AN VA NP PP	XP PD SP U CC
Dutch	N V Adj Num Adv	Prep Conj Int Punc Art
	N V Adj Adv Punc	Adv Punc Art Prep Pron
	N V Adv Pron Prep	Adv Pron Prep Art Punc
	N V Adj Adv Pron	Adv Pron Art Prep Punc
German	NN ADJA NE VVFIN ADJD	VAIMP PRELAT PTKNEG VMPP $,
	NN NE ADJA ART ADV	KON PPER $. APPRART $,
	NN ADJA NE VVPP ART	PPER VAINF $. PTKZU $,
	NN NE ADJA VVPP ART	VAINF APPRART $. PTKZU $,
Japanese	NN Vfin Vte VN –	VS ? Pgen ADJsf VSimp
	NN Vfin Vte ADV ADJifin	UNIT VSfin Pacc Pgen .
	NN ADJifin ADV Vte VADJi	UNIT PSE . Pnom PNsf
	NN Vfin Vte ADV VN	PSSa PSE PNsf . Pacc
Portuguese	n prop v-fin adj v-pcp	conj-c art ec ? vp
	n prop art punc adj	v-pcp v-inf adv v-fin prp
	n prop adj v-fin v-pcp	art adv num punc prp
	n prop v-fin v-pcp v-inf	punc art num adv prp
Spanish	nc vm aq np rg	Fg Fc Fh X sn
	nc vm aq np rg	Fe cs Fc p0 va
	nc vm aq rg np	cc Fc di va pr
	nc vm aq np rg	Fe Fc pr p0 Fp
Swedish	NN VV AJ VN TP	IS IU MV IT I?
	NN AJ PO AB VV	IT ++ IK UK IP
	NN AJ PO VV AB	QV ++ IK UK IP
	NN AJ VV PO PR	QV ++ IK UK IP
Turkish	Noun Verb Adj NInf Prop	Real Dup Distrib Num Range
	Noun Verb NInf Punc APresPart	Noun Verb NInf Punc APresPart
	Noun Verb APresPart NInf Adv	Adv Punc Adj Prop Postp
	Noun Verb APresPart NInf Prop	Prop Adv Punc Zero Postp

Table A.1: Token-level POS tag ranking for all languages except English and Slovene: Each row lists the gold standard tags and outputs from the 1TW, +PRIOR, and +FEATS models.

Language	Top 5 (token-level)	Bottom 5 (token-level)
Arabic	N P X A C	Q FI Y - I
	N P X C S	X C S A VP
	N P A X C	A X C S VP
	N P X C A	X C A VP S
Bulgarian	Nc Punct R Vpp Vpi	Vii Nm Tg P V
	Nc R Punct Vpp Pp	Mc Dq Np Vpi Am
	Nc R Punct Vpp A	Mc Pd Dq Mo Dt
	Nc Punct R Vpi Pp	Mc Tn Dq An Vyp
Chinese	N V D DE C	A I Str Head H11
	N V D DE DM	N V D DE DM
	N V D DE	N V D DE
	N V D DE	N V D DE
Czech	N Z V A R	D C T I X
	N Z V R A	Z V R A J
	N V Z A R	V Z A R J
	N V Z A R	Z A R C J
Danish	NC VA XP SP AN	I XS XA PC XR
	XP NC SP VA PP	CC PD AC CS PI
	NC XP VA SP PP	CC PI PD AC CS
	NC XP VA SP AN	PI NP PD CC AC
Dutch	N V Prep Punc Art	Adj Conj Num Misc Int
	N V Art Prep Adv	Prep Adv Punc Pron Adj
	N V Prep Art Punc	Prep Art Punc Adv Pron
	N Prep V Art Adv	Art Adv Pron Punc Adj
German	NN ART APPR ADJA NE	PTKANT ITJ PPOSS VMPP VAIMP
	NN ART APPR ADV ADJA	APPRART KOUS PPER ADJD PTKZU
	NN ART APPR VAFIN ADV	KOUS CARD VAINF ADJD PTKZU
	NN ART ADV APPR NE	VAINF KOUS VVFIN PTKZU APPRART
Japanese	. PVfin NN P NF	VSbas VAUX VAUXbas VSimp VS
	. NN Pgen PVfin PSE	, PNsf Nwh UNIT Vcnd
	. PVfin Pgen P NN	, ADJiku PNsf Vcnd UNIT
	. PVfin Pgen P NN	, ADJiku NAMEper UNIT PNsf
Portuguese	n prp punc art v-fin	pp in ec ? vp
	art n punc prp adv	adj prop v-inf v-fin v-pcp
	art n punc prp v-fin	prop v-pcp v-inf num adv
	n art punc prp v-fin	v-inf adv num v-pcp adj
Spanish	nc sp vm da aq	Fh Y sn pe X
	nc da sp vm Fc	p0 np dn Fe va
	nc sp da vm rg	pr np di Fe va
	nc da sp vm aq	Fe p0 va cs z
Swedish	NN PO PR VV AB	PU IU YY IS XX
	NN PO PR AB VV	UK EN IC RO IT
	NN PO PR AB AJ	IC QV RO ID PN
	NN PR PO AJ VV	IC RO IT PN ID
Turkish	Noun Verb Punc Adj Adv	Real Dup Distrib Num Range
	Noun Verb Punc NInf APresPart	Noun Verb Punc NInf APresPart
	Noun Verb Punc APresPart NInf	NInf Adv Prop Adj Postp
	Noun Verb Punc APresPart NInf	NInf Prop Adv Postp Zero

Table A.2: Token-level POS tag ranking for all languages except English and Slovene. Each row lists the gold standard tags and outputs from the 1TW, +PRIOR, and +FEATS models.

Top 5 (type-level)
1. Noun-common (NC) Verb-main (VM) Adjective-qualificative (Adj-Q) Adverb-general (Avd-G) Adjective-ordinal (Adj-O)
2. NC VM Adv-G Adposition-preposition (Adp-Pre) Verb-copula (VC)
3. NC Adv-G Adj-Q VM Adp-Pre
4. NC Adv-G Adj-Q VM VC

Bottom 5 (type-level)
1. PUNC Abbreviation Interjection Numeral-special Numeral-multiple
2. VC Adj-Q PUNC Pronoun-reflexive (Pro-R) Conjunction-subordinating (Conj-S)
3. Conj-S Noun-proper Pro-R Conj-S PUNC
4. Conj-S Adp-Pre Conj-S Pro-R PUNC

Table A.3: Token-level POS tag ranking for Slovene. The four lists correponds to the gold standard tags and the outputs of the 1TW, +PRIOR, and +FEATS models respectively.

Top 5 (token-level)
1. Noun-common (NC) PUNC Verb-copula (VC) Verb-main (VM) Adposition-preposition (Adp-Pre)
2. NC PUNC VC Adverb-general (Adv-G) Adp-Pre
3. NC PUNC Adv-G VC Adp-Pre
4. NC PUNC Adv-G VC Adjective-qualificative (Adj-Q)

Bottom 5 (token-level)
1. Numeral-ordinal Interjection Abbreviation Numeral-special Numeral-multiple
2. Adp-Pre VM Pronoun-reflexive (Pro-R) Conjunction-subordinating (Conj-S) Adj-Q
3. Adj-Q VM Particle Conj-S Noun-proper (NP)
4. VM Conj-S Conjunction-coordinating Pro-R NP

Table A.4: Token-level POS tag ranking for Slovene. The four lists correponds to the gold standard tags and the outputs of the 1TW, +PRIOR, and +FEATS models respectively.

Bibliography

[1] Meni Adler and Michael Elhadad. An unsupervised morpheme-based hmm for hebrew morphological disambiguation. In *Proceedings of the ACL/CONLL*, pages 665–672, 2006.

[2] Rie Kubota Ando and Lillian Lee. Mostly unsupervised statistical segmentation of japanese: Application to kanji. In *Proceedings of the joint meeting of the conference on applied natural language processing and the North American chapter of the association for computational linguistics (ANLP-NAACL)*, pages 241–248, 2000.

[3] Richard N. Aslin, Jenny R. Saffran, and Elissa L. Newport. Computation of conditional probabilities statistics by 8-month-old infants. *Psychological Science*, 9(4):321–324, 1998.

[4] Eleftherios Avramidis and Philipp Koehn. Enriching morphologically poor languages for statistical machine translation. In *Proceedings of ACL-08: HLT*, 2008.

[5] Ibrahim Badr, Rabih Zbib, and James Glass. Segmentation for english-to-arabic statistical machine translation. In *Proceedings of ACL-08: HLT, Short Papers*, 2008.

[6] Ricardo Baeza-Yates and Berthier Ribiero-Neto. *Modern Information Retrieval*. Addison-Wesley, 1999.

[7] Michele Banko and Robert C. Moore. Part-of-speech tagging in context. In *Proceedings of Coling 2004*, pages 556–561, 2004.

[8] Taylor Berg-Kirkpatrick, Alexandre Bouchard-Côté, John DeNero, and Dan Klein. Painless unsupervised learning with features. In *Proceedings of NAACL-HLT*, pages 582–590, 2010.

[9] Phil Blunsom and Trevor Cohn. A hierarchical pitman-yor process hmm for unsupervised part of speech induction. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 865–874, 2011.

[10] Michael Brent. Minimal generative explanations: A middle ground between neurons and triggers. In *Proceedings of the 15th Annual Meeting of the Cognitive Science Society*, pages 28–36, 1993.

[11] Michael R. Brent. An efficient, probabilistically sound algorithm for segmentation and word discovery. *Machine Learning*, 34:71 – 105, 1999.

[12] Eric Brill. Unsupervised learning of disambiguation rules for part of speech tagging. In *Proceedings of the third workshop on very large corpora*, pages 1–13, 1995.

[13] Peter F. Brown, Peter V. deSouza, Robert L. Mercer, T. J. Watson, Vincent J. Della Pietra, and Jenifer C. Lai. Class-based n-gram models of natural language. *Computational Linguistics*, 18(4):467–479, 1992.

[14] Sabine Buchholz and Erwin Marsi. Conll-x shared task on multilingual dependency parsing. In *In Proc. of CoNLL*, pages 149–164, 2006.

[15] Tim Buckwalter. *Buckwalter Arabic Morphological Analyzer Version 2.0.* LDC catalog number LDC2002L49, 2004. ISBN 1-58563-324-0.

[16] Burcu Can and Suresh Manandhar. Unsupervised learning of morphology by using syntactic categories. In *Working Notes, CLEF 2009 Workshop*, 2009.

[17] Christos Christodoulopoulos, Sharon Goldwater, and Mark Steedman. Two decades of unsupervised POS induction: How far have we come? In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 575–584, 2010.

[18] Christos Christodoulopoulos, Sharon Goldwater, and Mark Steedman. A bayesian mixture model for pos induction using multiple features. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 638–647, 2011.

[19] Kenneth Ward Church. A stochastic parts program and noun phrase parser for unrestricted text. In *ANLP*, pages 136–143, 1988.

[20] Alexander Clark. Combining distributional and morphological information for part of speech induction. In *Proceedings of EACL*, pages 59–66, 2003.

[21] Ann Clifton and Anoop Sarkar. Combining morpheme-based machine translation with post-processing morpheme prediction. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 2011.

[22] Paul Cohen, Niall Adams, and Brent Heeringa. Voting experts: An unsupervised algorithm for segmenting sequences. *Intelligent Data Analysis*, 11(6): 607–625, 2007.

[23] Trevor Cohn, Phil Blunsom, and Sharon Goldwater. Inducing tree-substitution grammars. *Journal of Machine Learning Research*, 11:3053–3096, 2010.

[24] Michael Collins. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*, pages 1–8, 2002.

[25] Mathias Creutz and Krista Lagus. Unsupervised discovery of morphemes. In

Proceedings of the ACL-02 Workshop on Morphological and Phonological Learning, pages 21–30, 2002.

[26] Mathias Creutz and Krista Lagus. Unsupervised models for morpheme segmentation and morphology learning. *ACM Transactions on Speech and Language Processing*, 4(1), 2007.

[27] Sajib Dasgupta and Vincent Ng. Unsupervised part-of-speech acquisition for resource-scarce languages. In *Proceedings of the EMNLP-CoNLL*, pages 218–227, 2007.

[28] Sajib Dasgupta and Vincent Ng. High-performance, language-independent morphological segmentation. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 155–163, 2007.

[29] A. de Gispert and J.B. Mario. On the impact of morphology in english to spanish statistical mt. *Speech Communication*, 50(11-12):1034 – 1046, 2008. doi: 10.1016/j.specom.2008.05.003.

[30] Carl G. de Marcken. *Unsupervised Language Acquisition*. PhD thesis, Massachusetts Institute of Technology, 1996.

[31] Gregory Druck, Gideon Mann, and Andrew McCallum. Learning from labeled features using generalized expectation criteria. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 595–602, 2008.

[32] Gregory Dubbin and Phil Blunsom. Modelling the lexicon in unsupervised part of speech induction. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 116–125, 2014.

[33] Christopher J. Dyer. The "noisier channel": Translation from morphologically complex languages. In *Proceedings of the Second Workshop on Statistical Machine Translation*, 2007.

[34] J.G. Fiscus. A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (rover). In *Automatic Speech Recognition and Understanding, 1997. Proceedings., 1997 IEEE Workshop on*, pages 347 –354, dec 1997.

[35] Edward Gammon. Quantitative approximations to the word. In *Proceedings of the conference on Computational linguistics (COLING)*, pages 1–28, 1969.

[36] Jianfeng Gao and Mark Johnson. A comparison of bayesian estimators for unsupervised hidden markov model pos taggers. In *Proceedings of the EMNLP*, pages 344–352, 2008.

[37] Felix Golcher. Statistical text segmentation with partial structure analysis. In *Proceedings of KONVENS*, pages 44–51, 2006.

[38] Yoav Goldberg, Meni Adler, and Michael Elhadad. EM can find pretty good HMM POS-taggers (when given a good start). In *Proceedings of ACL-08: HLT*, pages 746–754, 2008.

[39] John A. Goldsmith. Unsupervised learning of the morphology of a natural language. *Computational Linguistics*, 27(2):153–198, 2001.

[40] Sharon Goldwater and Thomas L. Griffiths. A fully Bayesian approach to unsupervised part-of-speech tagging. In *Proceedings of the ACL*, pages 744–751, 2007.

[41] Sharon Goldwater and David McClosky. Improving statistical mt through morphological analysis. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, 2005.

[42] Sharon Goldwater, Thomas L. Griffiths, and Mark Johnson. Contextual dependencies in unsupervised word segmentation. In *Proceedings of the ACL*, pages 673–680, 2006.

[43] João Graça, Kuzman Ganchev, Ben Taskar, and Fernando Pereira. Posterior vs. parameter sparsity in latent variable models. In *Proceeding of NIPS*, pages 664–672, 2009.

[44] Peter D. Grünwald. *The Minimum Description Length Principle.* MIT Press, 2007.

[45] Nizar Habash. Four techniques for online handling of out-of-vocabulary words in arabic-english statistical machine translation. In *Proceedings of ACL-08: HLT, Short Papers*, 2008.

[46] Nizar Habash and Owen Rambow. Arabic tokenization, part-of-speech tagging and morphological disambiguation in one fell swoop. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 573–580. Association for Computational Linguistics, June 2005.

[47] Nizar Habash and Fatiha Sadat. Arabic preprocessing schemes for statistical machine translation. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, 2006.

[48] Margaret A. Hafer and Stephen F. Weiss. Word segmentation by letter successor varieties. *Information Storage and Retrieval*, 10:371–385, 2002.

[49] Aria Haghighi and Dan Klein. Prototype-driven learning for sequence models. In *Proceedings of the HLT-NAACL*, pages 320–327, 2006.

[50] Morris Halle and Alec Marantz, editors. *Some key features of Distributed Morphology.* Number 21 in MIT Working Papers in Linguistics. MIT Press, 1994.

[51] Harald Hammarström. *Unsupervised Learning of Morphology and the Languages of the World.* PhD thesis, University of Technology and University of Golthenburg, 2009.

[52] Heidi Harley and Colin Phillips, editors. *The Morphology-Syntax Connection.* Number 22 in MIT Working Papers in Linguistics. MIT Press, 1994.

[53] Zellig S. Harris. *Methods in Structural Linguistics.* The University of Chicago Press, 1951.

[54] Zellig S. Harris. From phoneme to morpheme. *Language*, 31(2):190–222, 1955.

[55] Zellig S. Harris. Morpheme boundaries within words: Report on a computer test. In *Transformation and Discourse Analysis Papers 73.* Department of Linguistics, University of Pennsylvania, Philadelphia, 1967.

[56] Kazi Saidul Hasan and Vincent Ng. Weakly supervised part-of-speech tagging for morphologically-rich, resource-scarce languages. In *Proceedings of EACL*, pages 363–371, 2009.

[57] William P. Headden III, Mark Johnson, and David McClosky. Improving unsupervised dependency parsing with richer contexts and smoothing. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 101–109, 2009.

[58] Charles F. Hockett. Two models of grammatical description. *Word*, 10:210–234, 1954.

[59] Mark Johnson. Why doesn't em find good hmm pos-taggers? In *Proceedings of EMNLP-CoNLL*, pages 296–305, 2007.

[60] Mark Johnson. Unsupervised word segmentation for Sesotho using adaptor grammars. In *Proceedings of the Tenth Meeting of ACL Special Interest Group on Computational Morphology and Phonology*, pages 20–27, June 2008.

[61] Mark Johnson and Sharon Goldwater. Improving nonparameteric bayesian inference: experiments on unsupervised word segmentation with adaptor grammars. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 317–325, 2009.

[62] Mark Johnson, Thomas L. Griffiths, and Sharon Goldwater. Adaptor grammars: A framework for specifying compositional nonparametric bayesian models. In B. Schölkopf, J.C. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 641–648. MIT Press, 2007.

[63] Patrick Juola, Chris Hall, and Adam Boggs. Corpus-based morphological segmentation by entropy changes. In *Third Conference on the Cognitive Science of Natural Language Processing*, 1994.

[64] Peter W. Jusczyk, Derek M. Houston, and Mary Newsome. The beginnings of word segmentation in english-learning infants. *Cognitive Psychology*, 39:159–207, 1999.

[65] Chunyu Kit. *Unsupervised Lexical Learning as Inductive Inference.* PhD thesis, University of Sheffield, 2000.

[66] Chunyu Kit and Yorick Wilks. Unsupervised learning of word boundary with description length gain. In *Proceedings of the CoNLL99 ACL Workshop*, pages 1–6, 1999.

[67] Dan Klein and Christopher Manning. Corpus-based induction of syntactic structure: Models of dependency and constituency. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04), Main Volume*, pages 478–485, 2004.

[68] Philipp Koehn. Statistical significance tests for machine translation evaluation. In *Proceedings of EMNLP 2004*, 2004.

[69] Philipp Koehn and Hieu Hoang. Factored translation models. In *Proceedings of EMNLP-CoNLL*, pages 868–876, 2007.

[70] Kimmo Koskenniemi. *Two-level Morphology: A General Computational Model for Word-Form Recognition and Production.* PhD thesis, University of Helsinki, 1983.

[71] Michael Lamar, Yariv Maron, and Elie Bienenstock. Latent-descriptor clustering for unsupervised POS induction. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 799–809, 2010.

[72] Michael Lamar, Yariv Maron, Marko Johnson, and Elie Bienstock. Svd Clustering for Unsupervised POS Tagging. In *Proceedings of ACL*, pages 215–219, 2010.

[73] Chia-ying Lee and James Glass. A nonparametric bayesian approach to acoustic model discovery. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 40–49, 2012.

[74] Yoong Keok Lee, Aria Haghighi, and Regina Barzilay. Simple type-level unsupervised POS tagging. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 853–861, 2010.

[75] Yoong Keok Lee, Aria Haghighi, and Regina Barzilay. Modeling syntactic context improves morphological segmentation. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning*, 2011.

[76] Young-Suk Lee. Morphological analysis for statistical machine translation. In *HLT-NAACL 2004: Short Papers*, 2004.

[77] Percy Liang. Semi-supervised learning for natural language. Master's thesis, Massachusetts Institute of Technology, 2005.

[78] Percy Liang, Michael I. Jordan, and Dan Klein. Type-based MCMC. In *Proceedings of NAACL-HLT*, pages 573–581, 2010.

[79] Minh-Thang Luong, Preslav Nakov, and Min-Yen Kan. A hybrid morpheme-word representation for machine translation of morphologically rich languages. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, 2010.

[80] David J.C. MacKay. *Information theory, inference, and learning algorithms.* Cambridge University Press, 2003.

[81] Christopher D. Manning. Part-of-speech tagging from 97% to 100%: Is it time for some linguistics? In Alexander Gelbukh, editor, *Computational Linguistics and Intelligent Text Processing, 12th International Conference, CICLing 2011, Proceedings, Part I*, volume 6608 of *Lecture Notes in Computer Science*, pages 171–189. Springer, 2011.

[82] Alfonso Medina-Urrea. Affix discovery based on entropy and economy measurements. In Nicholas Gaylord, Alexis Palmer, and Elias Ponvert, editors, *Computational Linguistics for Less-Studied Languages*, volume 10, pages 99–112. Texas Linguistics Society, 2008.

[83] Bernard Mérialdo. Tagging english text with a probabilistic model. *Computational Linguistics*, 20(2):155–171, 1994.

[84] Coşkun Mermer and Ahmet Afşın Akın. Unsupervised search for the optimal segmentation for statistical machine translation. In *Proceedings of the ACL 2010 Student Research Workshop*, pages 31–36, Uppsala, Sweden, 2010.

[85] Coskun Mermer and Murat Saraclar. Unsupervised turkish morphological segmentation for statistical machine translation. In *Workshop on Machine Translation and Morphologically-rich languages*, 2011.

[86] Einat Minkov, Kristina Toutanova, and Hisami Suzuki. Generating complex morphology for machine translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, 2007.

[87] Daichi Mochihashi, Takeshi Yamada, and Naonori Ueda. Bayesian unsupervised word segmentation with nested pitman-yor language modeling. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 100–108, 2009.

[88] Preslav Nakov and Hwee Tou Ng. Translating from morphologically complex languages: A paraphrase-based approach. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 2011.

[89] Jason Naradowsky and Kristina Toutanova. Unsupervised bilingual morpheme segmentation and alignment with context-rich hidden semi-markov models. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 2011.

[90] Tahira Naseem, Benjamin Snyder, Jacob Eisenstein, and Regina Barzilay. Multilingual part-of-speech tagging: Two unsupervised approaches. *JAIR*, 36:341–385, 2009.

[91] Tahira Naseem, Harr Chen, Regina Barzilay, and Mark Johnson. Using universal linguistic knowledge to guide grammar induction. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1234–1244, 2010.

[92] Radford M. Neal. Slice sampling. *Annals of Statistics*, 3(31):705–767, 2003.

[93] Craig G. Nevill-Manning and Ian H. Witten. Identifying hierarchical structure in sequences: A linear-time algorithm. *Journal of Artificial Intelligence Research*, 7(1):67–82, 1997.

[94] Franz Josef Och. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, 2003.

[95] Franz Josef Och and Hermann Ney. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51, 2003.

[96] Timothy O'Donnell, Jesse Snedeker, Joshua Tenenbaum, and Noah Goodman. Productivity and reuse in language. In *Proceedings of the 33rd Annual Meeting of the Cognitive Science Society*, pages 1613–1618, 2011.

[97] Kemal Oflazer. Two-level description of turkish morphology. In *Proceedings of the Sixth Conference of the European Chapter of the Association for Computational Linguistics*, 1993.

[98] Kemal Oflazer and Ilknur Durgar El-Kahlout. Exploring different representational units in English-to-Turkish statistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, 2007.

[99] Donald Cort Olivier. *Stochastic Grammars and Language Acquisition Mechanisms*. PhD thesis, Harvard University, 1968.

[100] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, 2002.

[101] Hoifung Poon, Colin Cherry, and Kristina Toutanova. Unsupervised morphological segmentation with log-linear models. In *Proceedings of HLT-NAACL 2009*, pages 209–217. Association for Computational Linguistics, June 2009.

[102] Maja Popović and Hermann Ney. Towards the use of word stems and suffixes for statistical machine translation. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)*, 2004.

[103] Sujith Ravi and Kevin Knight. Minimized models for unsupervised part-of-speech tagging. In *Proceedings of ACL-IJCNLP*, pages 504–512, 2009.

[104] Jorma Rissanen. A universal prior for integers and estimation by minimum description length. *The Annals of Statistics*, 11:416–431, 1983.

[105] Andrew Rosenberg and Julia Hirschberg. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 410–420, 2007.

[106] Antti-Veikko Rosti, Spyros Matsoukas, and Richard Schwartz. Improved word-level system combination for machine translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 312–319, 2007.

[107] Ryan Roth, Owen Rambow, Nizar Habash, Mona Diab, and Cynthia Rudin. Arabic morphological tagging, diacritization, and lemmatization using lexeme models and feature ranking. In *Proceedings of ACL-08: HLT, Short Papers*, 2008.

[108] Bertrand Russell. *In praise of idleness: and other essays*. Routledge Classics, 2004.

[109] Fatiha Sadat and Nizar Habash. Combination of arabic preprocessing schemes for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, 2006.

[110] Jenny R. Saffran, Elissa L. Newport, and Richard N. Aslin. Word segementation: The role of distributional cues. *Journal of Memory and Language*, 35: 606–621, 1996.

[111] Gerard Salton and Michael J. Mcgill. *Introduction to Modern Information Retrieval*. McGraw Hill, 1983.

[112] Hinrich Schutze. Distributional part of speech tagging. In *Proceedings of the EACL*, pages 141–148, 1995.

[113] Libin Shen, Jinxi Xu, and Ralph Weischedel. A new string-to-dependency machine translation algorithm with a target dependency language model. In *Proceedings of ACL-08: HLT*, 2008.

[114] Noah A. Smith and Jason Eisner. Contrastive estimation: Training log-linear models on unlabeled data. In *Proceedings of the ACL*, 2005.

[115] Benjamin Snyder. *Unsupervised Multilingual Learning.* PhD thesis, Massachusetts of Technology, 2010.

[116] Benjamin Snyder and Regina Barzilay. Crosslingual propagation for morphological analysis. In *Proceedings of the AAAI*, pages 848–854, 2008.

[117] Benjamin Snyder and Regina Barzilay. Unsupervised multilingual learning for morphological segmentation. In *Proceedings of ACL-08: HLT*, pages 737–745. Association for Computational Linguistics, June 2008.

[118] Benjamin Snyder, Tahira Naseem, Jacob Eisenstein, and Regina Barzilay. Unsupervised multilingual learning for POS tagging. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 1041–1050, 2008.

[119] Benjamin Snyder, Tahira Naseem, Jacob Eisenstein, and Regina Barzilay. Adding more languages improves unsupervised multilingual part-of-speech tagging: a bayesian non-parametric approach. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 83–91, 2009.

[120] Richard Sproat, William Gales, Chilin Shih, and Nancy Chang. A stochastic finite-state word-segmentation algorithm for chinese. *Computational Linguistics*, 22(3):377–404, 1996.

[121] David Stallard, Jacob Devlin, Michael Kayser, Yoong Keok Lee, and Regina Barzilay. 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)*, pages 322–327, 2012.

[122] Maosong Sun, Dayang Shen, and Benjamin K. Tsou. Chinese word segmentation without using lexicon and hand-crafted training data. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and*

17th International Conference on Computational Linguistics, Volume 2, pages 1265–1271, 1998.

[123] David Talbot and Miles Osborne. Modelling lexical redundancy for machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, 2006.

[124] Kristina Toutanova and Colin Cherry. A global model for joint lemmatization and part-of-speech prediction. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 486–494, August 2009.

[125] Kristina Toutanova and Colin Cherry. A global model for joint lemmatization and part-of-speech prediction. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 486–494, 2009.

[126] Kristina Toutanova and Mark Johnson. A bayesian lda-based model for semi-supervised part-of-speech tagging. In *Advances in Neural Information Processing Systems 20*, pages 1521–1528, 2008.

[127] Kristina Toutanova, Hisami Suzuki, and Achim Ruopp. Applying morphology generation models to machine translation. In *Proceedings of ACL-08: HLT*, 2008.

[128] Anand Venkataraman. A statistical model for word discovery in transcribed speech. *Computational Linguistics*, 27(3):351–372, 2001.

[129] Sami Virpioja, Jaakko J. Väyrynen, Mathias Creutz, and Markus Sadeniemi. Morphology-aware statistical machine translation based on morphs induced in an unsupervised manner. In *Proceedings of the Machine Translation Summit XI*, 2007.

[130] Paul MB Vitányi and Ming Li. Minimum description length induction, bayesianism, and kolmogorov complexity. *IEEE Transactions on Information Theory*, 46(2):446–464, 2000.

[131] Larry Wasserman. *All of nonparametric statistics*. Springer, 2006.

[132] J.G. Wolff. An algorithm for the segmentation of an artificial language analogue. *British Journal of Pscyhology*, 66(1):79–90, 1975.

[133] Dekai Wu. An algorithm for simultaneously bracketing parallel texts by aligning words. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, pages 244–251, 1995.

[134] Mei Yang and Katrin Kirchhoff. Phrase-based backoff models for machine translation of highly inflected languages. In *Proceedings of EACL*, 2006.

[135] Deniz Yuret, Mehmet Ali Yatbaz, and Enis Sert. Unsupervised instance-based part of speech induction using probable substitutes. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, 2014.

[136] Rabih Zbib, Erika Malchiodi, Jacob Devlin, David Stallard, Spyros Matsoukas, Richard Schwartz, John Makhoul, Omar F. Zaidan, and Chris Callison-Burch. Machine translation of arabic dialects. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 49–59, 2012.