Context-Based Morphological Disambiguation with Random Fields*

Noah A. Smith and **David A. Smith** and **Roy W. Tromble**
Department of Computer Science / Center for Language and Speech Processing
Johns Hopkins University, Baltimore, MD 21218 USA
{nasmith,dasmith,royt}@cs.jhu.edu

Abstract

Finite-state approaches have been highly successful at describing the morphological processes of many languages. Such approaches have largely focused on modeling the phone- or character-level processes that generate candidate lexical *types*, rather than *tokens* in *context*. For the full analysis of words in context, *disambiguation* is also required (Hakkani-Tür et al., 2000; Hajič et al., 2001). In this paper, we apply a novel source-channel model to the problem of morphological disambiguation (segmentation into morphemes, lemmatization, and POS tagging) for concatenative, templatic, and inflectional languages. The channel model exploits an existing morphological dictionary, constraining each word's analysis to be linguistically valid. The source model is a factored, conditionally-estimated random field (Lafferty et al., 2001) that learns to disambiguate the full sentence by modeling local contexts. Compared with baseline state-of-the-art methods, our method achieves statistically significant error rate reductions on Korean, Arabic, and Czech, for various training set sizes and accuracy measures.

1 Introduction

One of the great successes in computational linguistics has been the construction of morphological analyzers for diverse languages. Such tools take in words and enumerate the possible morphological analyses—typically a sequence of morphemes, perhaps part-of-speech tagged. They are often encoded as finite-state transducers (Kaplan and Kay, 1981; Koskenniemi, 1983; Beesley and Karttunen, 2003).

What such tools do not provide is a means to *disambiguate* a word in *context*. For languages with complex morphological systems (inflective, agglutinative, and polysynthetic languages, for example), a word form may have many analyses. To pick the right one, we must consider the word's context. This problem has been tackled using statistical sequence models for Turkish (Hakkani-Tür et al., 2000) and Czech (Hajič et al., 2001); their approaches (and ours) are not unlike POS tagging, albeit with complex tags.

*This work was supported by a Fannie and John Hertz Foundation Fellowship, a NSF Fellowship, and a NDSEG Fellowship (sponsored by ARO and DOD). The views expressed are not necessarily endorsed by sponsors. We thank Eric Goldlust and Markus Dreyer for Dyna language support and Jason Eisner, David Yarowsky, and three anonymous reviewers for comments that improved the paper. We also thank Jan Hajič and Pavel Krbec for sharing their Czech tagger.

In this paper, we describe context-based models for morphological disambiguation that take full account of existing morphological dictionaries by estimating *conditionally* against only dictionary-accepted analyses of a sentence (§2). These models are an instance of conditional random fields (CRFs; Lafferty et al., 2001) and include overlapping features. Our applications include diverse disambiguation frameworks and we make use of linguistically-inspired features, such as local lemma dependencies and inflectional agreement. We apply our model to Korean and Arabic, demonstrating state-of-the-art results in both cases (§3). We then describe how our model can be expanded to complex, structured morphological tagging, including an efficient estimation method, demonstrating performance on Czech (§4).

2 Modeling Framework

Our framework is a source-channel model (Jelinek, 1976). The *source* (modeled probabilistically by p_s) generates a sequence of unambiguous tagged morphemes $\mathbf{y} = \langle y_1, y_2, ... \rangle \in \mathcal{Y}^+$ ($\mathcal{Y}$ is the set of unambiguous tagged morphemes in the language).[1] The precise contents of the tag will vary by language and corpus but will minimally include POS. $\mathbf{y}$ passes through a *channel* (modeled by p_c), which outputs $\mathbf{x} = \langle x_1, x_2, ... \rangle \in (\mathcal{X} \cup \{\text{OOV}\})^+$, a sequence of surface-level words in the language and out-of-vocabulary words (OOV; $\mathcal{X}$ is the language's vocabulary). Note that $|\mathbf{x}|$ may be smaller than $|\mathbf{y}|$, since some morphemes may combine to make a word. We will denote by $\mathbf{y}_i$ the contiguous subsequence of $\mathbf{y}$ that generates x_i; $\vec{y}$ will refer to a dictionary-recognized *type* in $\mathcal{Y}^+$.

At test time, we *decode* the observed $\mathbf{x}$ into the most probable sequence of tag/morpheme pairs:

$$\hat{\mathbf{y}} = \operatorname*{argmax}_{\mathbf{y}} p(\mathbf{y} \mid \mathbf{x}) = \operatorname*{argmax}_{\mathbf{y}} p_s(\mathbf{y}) \cdot p_c(\mathbf{x} \mid \mathbf{y}) \quad (1)$$

Training involves constructing p_s and p_c. We assume that there exists a training corpus of text (each word x_i annotated with its correct analysis $\mathbf{y}_i^*$) and a morphological dictionary. We next describe the channel model and the source model.

[1] The sequence also includes segmentation markings between words, not shown to preserve clarity.

Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP), pages 475–482, Vancouver, October 2005.

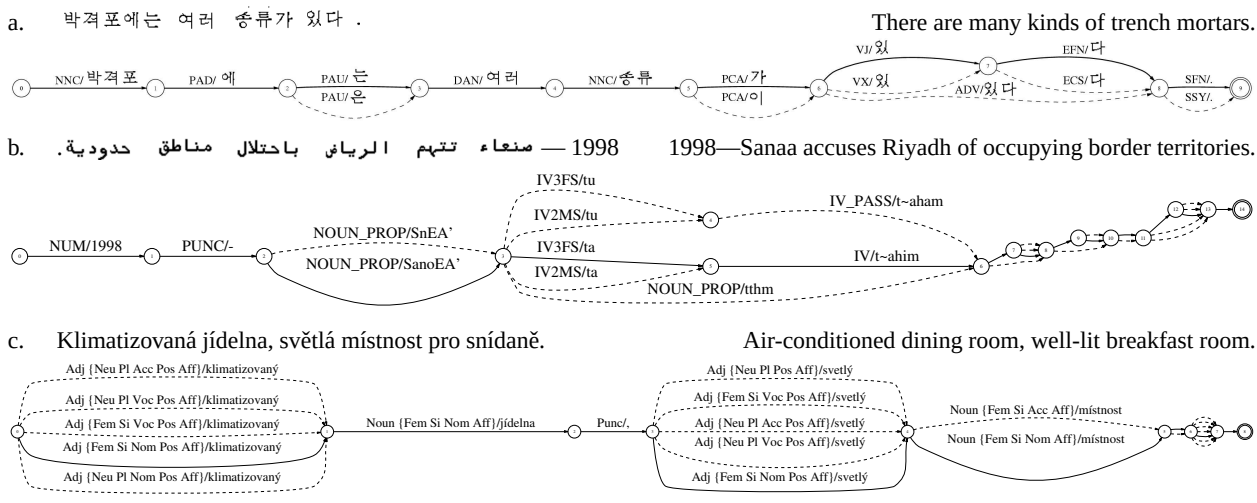

Figure 1: Lattices for example sentences in Korean (a), Arabic (b), and Czech (c). Arabic lemmas are not shown, and some Arabic and Czech arcs are unlabeled, for readability. The Arabic morphemes are shown in Buckwalter's encoding. The arcs in the correct path through each lattice are solid (incorrect arcs are dashed). Note the adjective-noun agreement in the correct path through the Czech lattice (c). The Czech lattice has no lemma-ambiguity; this is typical in Czech (see §4).

2.1 Morphological dictionaries and the channel

A great deal of research has gone into developing morphological analysis tools that enumerate valid analyses $\vec{y} \in \mathcal{Y}^+$ for a particular word $x \in \mathcal{X}$. Typically these tools are unweighted and therefore do not enable token disambiguation.[2]

They are available for many languages. We will refer to this source of categorial lexical information as a morphological dictionary d that maps $\mathcal{X} \rightarrow 2^{\mathcal{Y}^+}$. The set $d(x)$ is the set of analyses for word x; the set $d(\mathbf{x})$ is the set of whole-sentence analyses for sentence $\mathbf{x} = \langle x_1, x_2, ... \rangle$.

$d(\mathbf{x})$ can be represented as an acyclic lattice with a "sausage" shape familiar from work in speech recognition (Mangu et al., 1999). Note that for languages with bound morphemes, $d(x)$ will consist of a set of sequences of tokens, so a given "link" in the sausage lattice may contain paths of different lengths. Fig. 1 shows sausage lattices for sentences in three languages.

In this paper, the dictionary defines the support set of the channel model. That is, $p_c(\mathbf{x} \mid \mathbf{y}) > 0$ if and only if $\mathbf{y} \in d(\mathbf{x})$. This is a clean way to incorporate domain knowledge into the probabilistic model; this kind of constraint has been applied in previous work at decoding time (Hakkani-Tür et al., 2000; Hajič et al., 2001). In such a model, each word is independent of its neighbors (because the dictionary ignores context).

Estimation. A *unigram* channel model defines

$$p_c(\mathbf{x} \mid \mathbf{y}) \stackrel{\text{def}}{=} \prod_{i=1}^{|\mathbf{x}|} p(x_i \mid \mathbf{y}_i) \quad (2)$$

The simplest estimate of this model is to make $p(\cdot, \cdot)$ *uniform* over $(x, \vec{y})$ such that $\vec{y} \in d(x)$. Doing so and marginalizing to get $p(x \mid \vec{y})$ makes the channel model encode categorial information only, leaving all learning to the source model.[3]

Another way to estimate this model is, of course, from data. This is troublesome, because—modulo optionality—x is expected to be *known* given $\vec{y}$, resulting in a huge model with mostly 1-valued probabilities. Our solution is to take a *projection* π of $\vec{y}$ and let $p(\cdot \mid \vec{y}) \approx p(\cdot \mid \pi(\vec{y}))$. In this paper, π maps the analysis to its morphological tag (or tag sequence). We will refer to this as the "tag channel."

OOV. Morphological dictionaries typically do not have complete coverage of a language. We can augment them in two ways using the training data. If a known word x (one for which $d(x)$ is non-empty) appears in the training dataset with an analysis not in $d(x)$, we add the entry to the dictionary. Unknown words (those not recognized by the dictionary) are replaced by an OOV symbol. $d(\text{OOV})$ is taken to be the set of all analyses for any OOV word seen in training. Rather than attempt to recover the morpheme sequence for an OOV word, in this paper we try only for the tag sequence, replacing all of an OOV's morphemes with the OOV symbol. Since OOV symbols account for less than 2% of words in our corpora, we leave

[2] *Probabilistic* modeling of what we call the morphological channel was first carried out by Levinger et al. (1995), who used unlabeled data to estimate $p(\vec{y} \mid x)$ for Hebrew, with the support defined by a dictionary.

[3] Note that this makes the channel term in Eq. 1 a constant. Then decoding means maximizing $p_s(\mathbf{y})$ over $\mathbf{y} \in d(\mathbf{x})$, equivalently maximizing $p(\mathbf{y} \mid d(\mathbf{x}))$.

more sophisticated channel models to future work.

2.2 The source model

The source model p_s defines a probability distribution over $\mathcal{Y}^+$, sequences of (tag, morpheme) pairs. Our source models can be viewed as weighted multi-tape finite-state automata, where the weights are associated with local, often overlapping features of the path through the automaton.

Estimation. We estimate the source *conditionally* from annotated data. That is, we maximize

$$\sum_{(\mathbf{x},\mathbf{y})\in\mathcal{X}^+\times\mathcal{Y}^+} \tilde{p}(\mathbf{x},\mathbf{y}) \log p_s\left(\mathbf{y} \mid d(\mathbf{x}), \vec{\theta}\right) \quad (3)$$

where $\tilde{p}(\cdot,\cdot)$ is the empirical distribution defined by the training data and $\vec{\theta}$ are the model parameters. In terms of Fig. 1, our learner maximizes the weight of the correct (solid) path through each lattice, at the expense of the other incorrect (dashed) paths. Note that

$$\log p_s\left(\mathbf{y} \mid d(\mathbf{x}), \vec{\theta}\right) = \log \frac{p_s\left(\mathbf{y} \mid \vec{\theta}\right)}{\sum_{\mathbf{y}'\in d(\mathbf{x})} p_s\left(\mathbf{y}' \mid \vec{\theta}\right)} \quad (4)$$

The sum in the denominator is computed using a dynamic programming algorithm (akin to the forward algorithm); it involves computing the sum of all paths through the "sausage" lattice of possible analyses for $\mathbf{x}$. By doing this, we allow knowledge of the support of the *channel* model to enter into our estimation of the *source* model. It is important to note that the *estimation* of the model (the objective function used in training, Eq. 3) is distinct from the source-channel *structure* of the model (Eq. 1).

The lattice-conditional estimation approach was first used by Kudo et al. (2004) for Japanese segmentation and hierarchical POS-tagging and by Smith and Smith (2004) for Korean morphological disambiguation. The resulting model is an instance of a *conditional random field* (CRF; Lafferty et al., 2001). When training a CRF for POS tagging, IOB chunking (Sha and Pereira, 2003), or word segmentation (Peng et al., 2004), one typically structures the conditional probabilities (in the objective function) using domain knowledge: in POS tagging, the set of allowed tags for a word is used; in IOB chunking, the bigram "O I" is disallowed; and in segmentation, a lexicon is used to enumerate the possible word boundaries.[4]

[4]This refinement is in the same vein as the move from *maximum likelihood* estimation to *conditional* estimation. MLE would make the sum in the denominator of Eq. 4 $\mathcal{Y}^+$, which for log-linear models is often intractable to compute (and for sequence models may not converge). Conditional estimation limits the sum to the subset of $\mathcal{Y}^+$ that is consistent with $\mathbf{x}$, and our variant further stipulates consistency with the dictionary entries for $\mathbf{x}$.

Our approach is the same, with two modifications. First, we model the relationship between labels y_i and words x_i in a separately-estimated channel model (§2.1). Second, our labels are complex. Each word x_i is tagged with a *sequence* of one or more tagged morphemes; the tags may include multiple fields. This leads to models with more parameters. It also makes the dictionary especially important for limiting the size of the sum in the denominator, since a complex label set $\mathcal{Y}$ could in principle lead to a huge hypothesis space for a given sentence $\mathbf{x}$. Importantly, it makes training conditions more closely match testing conditions, ruling out hypotheses a dictionary-aware decoder would never consider.

Optimization. The objective function (Eq. 3) is concave and known to have a unique global maximum. Because log-linear models and CRFs have been widely described elsewhere (e.g., Lafferty, 2001), we note only that we apply a standard first-order numerical optimization method (L-BFGS; Liu and Nocedal, 1989). The structure, features, and regularization of our models will be described in §3 and §4.

Prior work (morphological source models). Hakkani-Tür et al. (2000) described a system for Turkish that was essentially a source model; Hajič et al. (2001) described an HMM-based system for Czech that could be viewed as a combined source and channel. Both used dictionaries and estimated their (generative) models using maximum likelihood (with smoothing).[5] Given enough data, a ML-estimated model will learn to recognize a good path $\mathbf{y}$, but it may not learn to discriminate a good $\mathbf{y}$ from wrong alternatives *per se*. The generative framework is limiting as well, disallowing the straightforward inclusion of arbitrary overlapping features. We present a competitive Czech model in §4.

3 Concatenative Models

The beauty of log-linear models is that estimation is straightforward given *any* features, even ones that are not orthogonal (i.e., "overlap"). This permits focusing on feature (or feature template) selection without worries about the mathematics of training.

We consider two languages modeled by concatenative processes with surface changes at morpheme boundaries: Korean and Arabic.

Our model includes features for tag n-grams, morpheme n-grams, and pairs of the two (possibly of different lengths and offsets). Fig. 2 illustrates TM3, our base model. TM3 includes feature templates for some tuples of three or fewer elements, plus begin and end templates.

[5]Hajič et al. also included a rule-based system for pruning hypotheses, which gave slight performance gains.

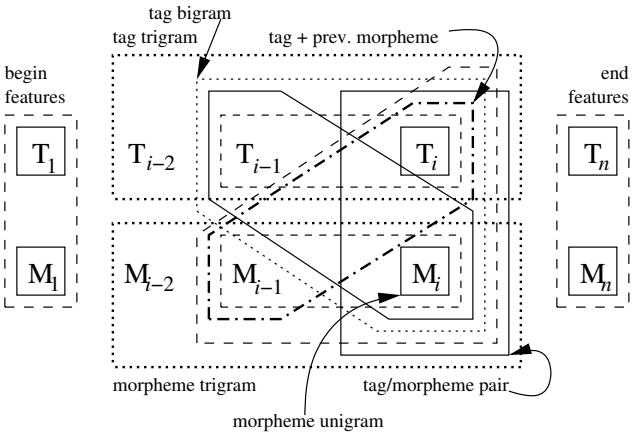

Figure 2: The base two-level trigram source model, TM3. Each polygon corresponds to a feature template. This is a two level, second-order Markov model (weighted finite-state machine) parameterized with overlapping features. Note that only some features are labeled in the diagram.

A variant, TM3H, includes all of the same templates, plus a similar set of templates that look only at *head* morphemes. For instance, a feature fires for each trigram of heads, even though there are (bound) morphemes between them. This increases the domain of locality for semantic content-bearing morphemes. This model requires slight changes to the dynamic programming algorithms for inference and training (the previous two heads must be remembered at each state).

Every instantiation of the templates seen in *any* lattice $d(\mathbf{x})$ built from training data is included in the model, not just those seen in correct analyses $\mathbf{y}^*$.[6]

3.1 Experimental design

In all of our experiments, we vary the training set size and the amount of smoothing, which is enforced by a diagonal Gaussian prior (L_2 regularizer) with variance σ^2. The $\sigma^2 = \infty$ case is equivalent to not smoothing. We compare performance to the expected performance of a randomized baseline that picks for each word token x an analysis from $d(x)$; this gives a measure of the amount of ambiguity and is denoted "channel only." Performance of unigram, bigram, and trigram HMMs estimated using maximum likelihood (barely smoothed, using add-10^{-14}) is also reported. (The unigram HMM simply picks the most likely $\vec{y}$ for each x, based on training data and is so marked.)

In the experiments in this section, we report three performance measures. *Tagging* accuracy is the fraction of words whose tag sequence was correctly identified in entirety; *morpheme* accuracy is defined analogously. *Lemma* accuracy is the fraction of words whose lemma was correctly identified.

3.2 Korean experiments

We applied TM3 and TM3H to Korean. The dataset is the Korean Treebank (Han et al., 2002), with up to 90% used for training and 10% (5K words) for test. The morphological dictionary is `klex` (Han, 2004). There are 27 POS tags in the tag set; the corpus contains 10K word types and 3,272 morpheme types. There are 1.7 morphemes per word token on average ($\sigma = 0.75$). A Korean word generally consists of a head morpheme with a series of enclitic suffixes. In training the head-augmented model TM3H, we assume the first morpheme of every word is the head and lemma.

Results are shown in Tab. 1. TM3H achieved very slight gains over TM3, and the tag channel model was helpful only with the smaller training set. The oracle (last line of Tab. 1) demonstrates that the coverage of the dictionary remains an obstacle, particularly for recovering morphemes. Another limitation is the small amount of training data, which may be masking differences among estimation conditions. We report the performance of TM3H with "factored" estimation. This will be discussed in detail in §4; it means that a model containing *only* the head features was trained on its own, then combined with the independently trained TM3 model at test time. Factored training was slightly faster and did not affect performance at all; accuracy scores were identical with unfactored training.

Prior work (Korean). Similar results were presented by Smith and Smith (2004), using a similar estimation strategy with a model that included far more feature templates. TM3 has about a third as many parameters and TM3H about half; performance is roughly the same (numbers omitted for space). Korean disambiguation results were also reported by Cha et al. (1998), who applied a deterministic morpheme pattern dictionary to segment words, then used a bigram HMM tagger. They also applied transformation-based learning to fix common errors. Due to differences in tag set and data, we cannot compare to that model; a bigram baseline is included.

3.3 Arabic experiments

We applied TM3 and TM3H to Arabic. The dataset is the Arabic Treebank (Maamouri et al., 2003), with up to 90% used for training and 10% (13K words) for test. The morphological dictionary is Buckwalter's analyzer (version 2), made available by the LDC (Buckwalter, 2004).[7] This analyzer has total coverage of the corpus; there are no

[6] If we used only features observed to occur in $\mathbf{y}^*$, we would not be able to learn negative weights for *unlikely* bits of structure seen in the lattice $d(\mathbf{x})$ but not in $\mathbf{y}^*$.

[7] Arabic morphological processing was also addressed by Kiraz (2000), who gives a detailed review of symbolic work in that area, and by Darwish (2002).

			Korean						Arabic								
			POS tagging accuracy		morpheme accuracy		lemma accuracy		POS tagging accuracy			morpheme accuracy			lemma accuracy		
		σ^2	32K	49K	32K	49K	32K	49K	38K	76K	114K	38K	76K	114K	38K	76K	114K
	most likely $\vec{y}$		86.0	86.9	87.5	88.8	95.3	95.7	84.5	87.0	88.3	83.2	86.2	87.0	37.9	39.8	40.9
uniform channel	channel only		62.6	62.6	70.3	70.8	86.4	86.4	43.7	43.7	43.7	41.2	41.2	41.2	27.2	27.2	27.2
	bigram HMM		90.7	91.2	83.2	86.1	96.9	97.2	90.3	92.0	92.8	89.2	91.4	91.6	85.7*	87.8*	87.9*
	trigram HMM		91.5	91.8	83.3	86.0	97.0	97.2	89.8	92.0	93.0	88.5	91.3	91.3	85.2*	87.8*	87.7*
	TM3	∞	90.7	91.3	**89.3**	**90.5**	97.1	97.4	**94.6**	**95.4**	**95.9**	**93.4**	**94.3**	**94.9**	**89.7***	**90.5***	**90.7***
		10	91.2	91.7	**89.4**	**90.6**	97.1	**97.6**	**95.3**	**95.7**	**96.1**	**93.9**	**94.5**	**95.0**	**90.2***	**90.6***	**91.1***
		1	91.5	92.2	**89.4**	**90.6**	97.1	**97.5**	**95.2**	**95.7**	**96.0**	**93.9**	**94.5**	**94.7**	**90.0***	**90.7***	**91.0***
	TM3H (factored)	∞	91.1	91.1	**89.3**	**90.4**	97.2	97.5	**95.0**	**95.7**	**96.0**	**94.0**	**94.8**	**95.3**	**93.3**	**93.9**	**94.2**
		10	91.3	91.9	**89.5**	**90.6**	**97.3**	**97.6**	**95.3**	**95.7**	**96.1**	**94.2**	**94.7**	**95.4**	**93.4**	**93.6**	**94.4**
		1	91.4	92.2	**89.5**	**90.7**	**97.3**	**97.6**	**95.4**	**95.8**	**96.1**	**94.4**	**94.8**	**95.1**	**93.3**	**93.8**	**94.2**
tag channel	channel only		51.4	51.3	60.6	60.4	81.2	81.7	41.4	40.6	40.1	39.9	39.1	38.6	26.7*	26.5*	26.4*
	bigram HMM		91.2	90.9	**88.9**	**90.1**	97.0	97.3	**91.0**	**92.3**	**93.4**	**89.7**	91.5	**91.9**	**88.1***	**89.9***	**90.0***
	trigram HMM		91.6	91.9	**88.9**	**90.2**	97.1	**97.4**	**91.1**	**92.9**	**93.7**	**89.6**	**92.2**	**92.0**	**88.1***	**90.6***	**90.4***
	TM3	∞	90.8	91.0	**89.5**	**90.5**	**97.4**	97.5	**95.1**	**95.7**	**96.0**	**93.8**	**94.6**	**95.0**	**92.2***	**93.1***	**93.2***
		10	90.6	91.1	**89.5**	**90.7**	97.2	**97.6**	**95.2**	**95.6**	**96.0**	**93.9**	**94.7**	**95.0**	**92.4***	**93.2***	**93.5***
		1	90.1	90.9	**89.5**	**90.7**	97.1	**97.6**	**94.9**	**95.5**	**95.8**	**93.8**	**94.5**	**94.8**	**92.2***	**93.0***	**93.1***
	TM3H (factored)	∞	91.0	91.0	**89.4**	**90.5**	97.2	**97.6**	**95.1**	**95.8**	**96.0**	**94.0**	**95.1**	**95.4**	**93.3**	**94.3**	**94.4**
		10	90.4	91.2	**89.6**	**90.7**	**97.4**	**97.6**	**95.2**	**95.7**	**96.0**	**94.1**	**94.8**	**95.4**	**93.3**	**94.0**	**94.6**
		1	90.1	91.0	**89.5**	**90.7**	**97.3**	**97.6**	**95.1**	**95.5**	**95.9**	**94.1**	**94.9**	**95.1**	**93.3**	**94.0**	**94.4**
	oracle given $d(\mathbf{x})$		95.3	95.7	90.2	91.2	98.1	98.3	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0

Table 1: Korean (left, 5K test-set) and Arabic (right, 13K test-set) disambiguation. A word is marked correct only if its entire tag (or morpheme) sequence (or lemma) was correctly identified. Morpheme and lemma accuracy do not include OOV words. The oracle is an upper bound on accuracy given the morphological dictionary. *These models do not explicitly predict lemmas; the lemma is chosen arbitrarily from those that match the hypothesized tag/morpheme sequence for each word. **Bold** scores indicate a significant improvement over the trigram HMM (binomial sign test, $p < 0.05$).

OOV words. There are 139 distinct POS tags; these contain some inflectional information which we treat atomically. For speed, TM3H was trained in two separate pieces: TM3 and the lemma features added by TM3H.

Arabic has a templatic morphology in which consonantal roots are transformed into surface words by the insertion of vowels and ancillary consonants. Our system does not model this process except through the use of Buckwalter's dictionary to define the set of analyses for each word (cf., Daya et al., 2004, who modeled interdigitation in Hebrew). We treat the analysis of an Arabic word as a sequence $\vec{y}$ of pairs of morphemes and POS tags, plus a lemma. The lemma, given in the dictionary, provides further disambiguation beyond the head morpheme. The lemma is a standalone dictionary headword and not merely the consonantal root, as in some other work. The "heads" modeled by TM3H correspond to these lemmas. There are 20K word types, and 34K morpheme types. There are 1.7 morphemes per word token on average ($\sigma = 0.77$).

Results are shown in Tab. 1. Across tasks and training set sizes, our models reduce error rates by more than 36% compared to the trigram HMM source with tag channel. The TM3H model and the tag channel offer slight gains over the base TM3 model (especially on lemmatization), though the tag channel offers no help in POS tagging.

Prior work (Arabic). Both Diab et al. (2004) and Habash and Rambow (2005) use support-vector machines with local features; the former for tokenization, POS tagging, and base phrase chunking; the latter for full morphological disambiguation. Diab et al. report results for a coarsened 24-tag set, while we use the full 139 tags from the Arabic Treebank, so the systems are not directly comparable. Habash and Rambow present even better results on the same POS tag set. Our full disambiguation results appear to be competitive with theirs. Khoja (2001) and Freeman (2001) describe Arabic POS taggers and many of the issues involved in developing them, but because tagged corpora did not yet exist, there are no comparable quantitative results.

4 Czech: Model and Experiments

Inflective languages like Czech present a new set of challenges. Our treatment of Czech is not concatenative; following prior work, the analysis for each word x is a single tag/lemma pair y. Inflectional affixes in the surface form are represented as features in the tag. While lemmatization of Czech is not hard (there is little ambiguity), tagging is quite difficult, because morphological tags are highly complex. Our tag set is the Prague Dependency Treebank (PDT; Hajič, 1998) set, which consists of fifteen-field tags that indicate POS as well as inflectional information (case, number, gender, etc.). There are over

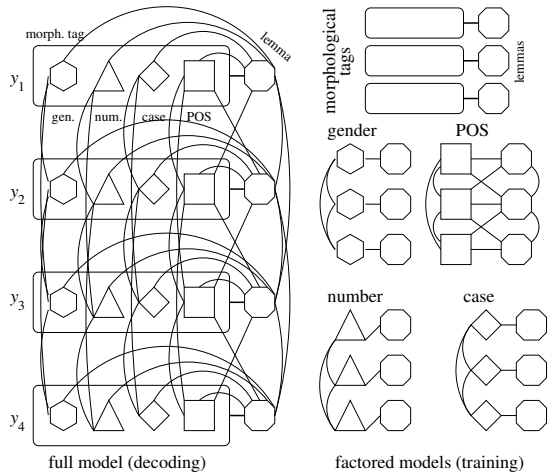

Figure 3: The Czech model, shown as an undirected graphical model. The structure of the full model is on the left; factored components for estimation are shown on the right. Each of these five models contains a subset of the TM3 features. The full model is only used to decode. The factored models make training faster and are used for pruning.

1,400 distinct tag types in the PDT.

Czech has been treated probabilistically before, perhaps most successfully by Hajič et al. (2001).[8] In contrast, we estimate conditionally (rather than by maximum likelihood for a generative HMM) and separate the training of the source and the channel. We also introduce a novel *factored* treatment of the morphological tags.

4.1 Factored tags and estimation

Because Czech morphological tags are not monolithic, the choice among them can be treated as several more or less orthogonal decisions. The case feature of one word, for example, is expected to be conditionally independent of the next word's gender, given the next word's case. Constraints in the language are expected to cause features like case, number, and gender to agree locally (on words that have such features) and somewhat independently of each other. Coarser POS tagging may be treated as another, roughly independent stream.

Log-linear models and the use of a morphological dictionary make this kind of tag factoring possible. Our approach is to separately train five log-linear models. Each model is itself an instance of some of the templates from TM3, modeling a projection of the full analysis. The model and its factored components are illustrated in Fig. 3.

POS model. The full tag is replaced by the POS tag (the first two fields); there are 60 POS tags. The TM3 feature templates are included twice: once for the full tag and once for a coarser tag (the first PDT field, for which there are 12 possible values).[9]

Gender, number, and case models. The full tag is replaced by the gender (or case or number) field. This model includes bigrams and trigrams as well as field-morpheme unigram features. These models are intended to learn to predict local agreement.

Tag-lemma model. This model contains unigram features of full PDT tags, both alone and with lemmas. It is intended to learn to penalize morphological tags that are rare, or that are rare with a particular lemma. In our formulation, this is *not* a channel model, because it ignores the surface word forms.

Each model is estimated independently of the others. The lattice $d(\mathbf{x})$ against which the conditional probabilities are estimated contains the relevant *projection* of the full morphological tags (with lemmas). To decode, we run a Viterbi-like algorithm that uses the union of all models' features to pick the best analysis (full morphological tags and lemmas) allowed by the dictionary.

There are two important advantages of factored training. First, each model is faster to train alone than a model with all features merged; in fact, training the fully merged model takes far too long to be practical. Second, factored models can be held out at test time to measure their effect on the system, without retraining.

Prior work (factored training). Separately training different models that predict the same variables (e.g., $\mathbf{x}$ and $\mathbf{y}$) then combining them for consensus-based inference (either through a mixture or a product of probabilities) is an old idea (Genest and Zidek, 1986). Recent work in learning weights for the component "expert" models has turned to *cooperative* techniques (Hinton, 1999). Decoding that finds $\mathbf{y}$ (given $\mathbf{x}$) to maximize some weighted average of log-probabilities is known as a *logarithmic opinion pool* (LOP). LOPs were applied to CRFs (for named entity recognition and tagging) by Smith et al. (2005), with an eye toward regularization. Their experts (each a CRF) contained overlapping feature sets, and the combined model achieved much the same effect as training a single model with smoothing. Note that our models, unlike theirs, *partition* the feature space; there is only one CRF, but some parameters are ignored when estimating other parameters. We have not estimated log-domain mixing coefficients—we weight all models' contributions equally. Sutton and McCallum (2005) have applied factored estimation to CRFs, motivated (like us) by speed; they also describe how factored estimation

[8] Czech morphological processing was studied by Petkevič (2001), Hlavácová (2001) (who focuses on handling OOV words), and Mráková and Sedlacek (2003) (who use partial parsing to reduce the set of possible analyses), *inter alia*.

[9] Lemma-trigram and fine POS-unigram/lemma-bigram features were eliminated to limit model size.

	σ^2	full morph. accuracy 376K	full morph. accuracy 768K	lemma accuracy 376K	lemma accuracy 768K	POS accuracy 376K	POS accuracy 768K	OOV POS accuracy 376K	OOV POS accuracy 768K
channel only		61.4	60.3	85.1	84.2	88.5	87.2	17.8	16.4
most likely $\vec{y}$		80.0	80.8	98.1	98.1	97.9	97.8	52.0	52.0
Hajič et al. HMM		**88.8**	**89.2**	97.9	97.9	95.8	95.8	52.0	52.0
+ OOV model		90.5	90.8	97.9	97.9	96.7	96.6	93.0	92.9
full	∞	88.1	88.5	**98.3**	**98.5**	**98.3**	**98.3**	**60.2**	**61.8**
oracle given pruning		98.6	99.3	99.5	99.6	99.1	99.7	60.2	90.3
	10	88.4	88.5	**98.4**	**98.4**	**98.3**	**98.2**	**61.8**	**59.4**
oracle given pruning		99.3	99.3	99.5	99.6	99.8	99.7	93.4	90.6
	1	88.6	88.6	**98.4**	**98.4**	**98.2**	**98.1**	**60.0**	**56.7**
oracle given pruning		99.3	99.3	99.5	99.6	99.8	99.8	95.0	94.0
– POS	∞	87.9	88.0†	**98.2**	**98.2**†	**98.0**	**97.9**†	**55.7**	51.7†
	10	88.1	88.3†	**98.2**	**98.3**†	**98.0**	**97.9**†	**55.4**	51.6†
	1	88.4	88.5†	**98.2**	**98.2**†	**98.0**	**97.9**†	**55.0**	51.9†
– tag-lemma	∞	87.8	88.3	**98.3**	**98.6**	**98.3**	**98.3**	**60.2**	**59.7**
	10	88.0	88.1	**98.4**	**98.5**	**98.3**	**98.2**	**59.1**	**59.1**
	1	88.0	88.1	**98.4**	**98.4**	**98.2**	**98.1**	**59.0**	**58.1**
POS only	∞	65.6*	65.5*	**98.3**	**98.6**	**98.3**	**98.4**	**60.2**	**63.7**
	10	65.7*	65.5*	**98.5**	**98.6**	**98.5**	**98.5**	**65.2**	**66.4**
	1	65.7*	65.5*	**98.6**	**98.7**	**98.6**	**98.6**	**67.2**	**67.2**
POS & tag-lemma†	∞	81.2	82.3	**98.3**	**98.6**	**98.3**	**98.4**	**60.2**	**63.9**
	10	81.9	82.3	**98.5**	**98.6**	**98.4**	**98.5**	**65.8**	**67.2**
	1	82.0	82.3	**98.4**	**98.5**	**98.5**	**98.4**	**67.8**	**66.3**
oracle given $d(\mathbf{x})$		99.8	99.8	99.5	99.6	99.9	99.9	100.0	100.0

Table 2: Czech disambiguation: test-set (109K words) accuracy. A word is marked correct only if its entire morphological tag (or morpheme or POS tag) was correctly identified. Note that the full tag is a complex, 15-field morphological label, while "POS" is a projection down to a tagset of size 60. Lemma accuracy does not include OOV words. *The POS-only model selects only POS, not full tags; these measures are expected performance if the full tag is selected randomly from those in the dictionary that match the selected POS. †Required more aggressive pruning. **Bold** scores were significantly better than the HMM of Hajič et al. (binomial sign test, $p < 0.05$). Our models were slightly but significantly worse on full tagging, but showed significant improvements on recovering POS tags and lemmas.

maximizes a lower bound on the unfactored objective. Smith and Smith (2004) applied factored estimation to a bilingual weighted grammar, driven by data limitations.

4.2 Experiments

Our corpus is the PDT (Hajič, 1998), with up to 60% used for training and 10% (109K words) used for test.[10] The morphological dictionary is the one packaged with the PDT; it covers about 98% of the tokens in the corpus. The remaining 2% have (unsurprisingly) a diverse set of 300–400 distinct tags, depending on the training set size.[11]

Results are shown in Tab. 2. We compare to the HMM of (Hajič et al., 2001) *without* its OOV component.[12] We report morphological tagging accuracy on words; we also report lemma accuracy (on non-OOV words), POS accuracy on all words, and POS accuracy on OOV words. The channel model (not shown) tended to have a small, harmful effect on performance.

Without any explicit OOV treatment, our POS-only component model significantly reduces lemma and POS errors compared to Hajič et al.'s model. On recovering *full* morphological tags, our *full* model is close in performance to Hajič et al., but still significantly worse. It is likely that for many tasks, these performance gains are more helpful than the loss on full tagging is harmful.

Why doesn't our full model perform as well as Hajič et al.'s model? An error analysis reveals that our full model (768K, $\sigma^2 = 1$), compared to the HMM (768K) had 91% as many number errors but 0.1% more gender and 31% more case errors. Taking out those three models ("POS & tag-lemma" in Fig. 2) is helpful on all measures except full tagging accuracy, due in part to substantially increased errors on gender (87% increase), case (54%), and number (35%). The net effect of these components, then, is helpful, but not quite helpful enough to match a well-smoothed HMM on complex tagging. We compared the models on the training set and found the same pattern, demonstrating that this is not merely a matter of over-fitting.

5 Future Work

Two clear ways to improve our models present themselves. The first is better OOV handling, perhaps through an improved channel model. Possibilities include learning weights to go inside the FST-encoded dictionaries and

[10] We used less than the full corpus to keep training time down; note that the training sets are nonetheless substantially larger than in the Korean and Arabic experiments.

[11] During training, these project down to manageable numbers of hypotheses in the factored models. At test-time, however, Viterbi search is quite difficult when OOV symbols occur consecutively. To handle this, we prune OOV arcs from the lattices using the factored POS and inflectional models. For each OOV, every model prunes a projection of the analysis (e.g., the POS model prunes POS tags) until 90% of the posterior mass or 3 arcs remain (whichever is more conservative). Viterbi decoding is run on a lattice containing OOV arcs consistent with the pruned projected lattices.

[12] Results *with* the OOV component are also reported in Tab. 2, but we cannot guarantee their experimental validity, since the OOV component is pre-trained and may have been trained on data in our test set.

directly modeling spelling changes. The second is to turn our factored model into a LOP. Training the mixture coefficients should be straightforward (if time-consuming) with a development dataset.

A drawback of our system (especially for Czech) is that some components (most notably, the Czech POS model) take a great deal of time to train (up to two weeks on 2GHz Pentium systems). Speed improvements are expected to come from eliminating some of the overlapping feature templates, generalized speedups for log-linear training, and perhaps further factoring.

6 Conclusion

We have explored morphological disambiguation of diverse languages using log-linear sequence models. Our approach reduces error rates significantly on POS tagging (Arabic and Czech), morpheme sequence recovery (Korean and Arabic), and lemmatization (all three languages), compared to baseline state-of-the-art methods For complex analysis tasks (e.g., Czech tagging), we have demonstrated that factoring a large model into smaller components can simplify training and achieve excellent results. We conclude that a *conditionally*-estimated source model informed by an existing morphological dictionary (serving as an unweighted channel) is an effective approach to morphological disambiguation.

References

K. R. Beesley and L. Karttunen. 2003. *Finite State Morphology*. CSLI.

T. Buckwalter. 2004. Arabic morphological analyzer version 2.0. LDC2004L02.

J. Cha, G. Lee, and J.-H. Lee. 1998. Generalized unknown morpheme guessing for hybrid POS tagging of Korean. In *Proc. of VLC*.

K. Darwish. 2002. Building a shallow Arabic morphological analyser in one day. In *Proc. of ACL Workshop on Computational Approaches to Semitic Languages*.

E. Daya, D. Roth, and S. Wintner. 2004. Learning Hebrew roots: Machine learning with linguistic constraints. In *Proc. of EMNLP*.

M. Diab, K. Hacioglu, and D. Jurafsky. 2004. Automatic tagging of Arabic text: From raw text to base phrase chunks. In *Proc. of HLT-NAACL*.

A. Freeman. 2001. Brill's POS tagger and a morphology parser for Arabic. In *Proc. of ACL Workshop on Arabic Language Processing*.

C. Genest and J. V. Zidek. 1986. Combining probability distributions: A critique and an annotated bibliography. *Statistical Science*, 1:114–48.

N. Habash and O. Rambow. 2005. Arabic tokenization, part-of-speech tagging and morphological disambiguation in one fell swoop. In *Proc. of ACL*.

J. Hajič, P. Krbec, P. Květoň, K. Oliva, and V. Petkevič. 2001. Serial combination of rules and statistics: A case study in Czech tagging. In *Proc. of ACL*.

J. Hajič. 1998. Building a syntactically annotated corpus: The Prague Dependency Treebank. In *Issues of Valency and Meaning*.

D. Z. Hakkani-Tür, K. Oflazer, and G. Tür. 2000. Statistical morphological disambiguation for agglutinative languages. In *Proc. of COLING*.

C.-H. Han, N.-R. Han, E.-S. Ko, H. Yi, and M. Palmer. 2002. Penn Korean Treebank: Development and evaluation. In *Proc. Pacific Asian Conf. Language and Comp.*

N.-R. Han. 2004. Klex: Finite-state lexical transducer for Korean. LDC2004L01.

G. Hinton. 1999. Products of experts. In *Proc. of ICANN*.

J. Hlavácová. 2001. Morphological guesser of Czech words. In *Proc. of TSD*.

F. Jelinek. 1976. Continuous speech recognition by statistical methods. *Proc. of the IEEE*, 64(4):532–557.

R. M. Kaplan and M. Kay. 1981. Phonological rules and finite-state transducers. Presented at Linguistic Society of America.

S. Khoja. 2001. APT: Arabic part-of-speech tagger. In *Proc. of NAACL Student Workshop*.

G. Kiraz. 2000. Multitiered nonlinear morphology using multitape finite automata: A case study on Syriac and Arabic. *Computational Linguistics*, 26(1):77–105.

K. Koskenniemi. 1983. Two-level morphology: A general computational model of word-form recognition and production. Technical Report 11, University of Helsinki.

T. Kudo, K. Yamamoto, and Y. Matsumoto. 2004. Applying conditional random fields to Japanese morphological analysis. In *Proc. of EMNLP*.

J. Lafferty, A. McCallum, and F. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. of ICML*.

M. Levinger, U. Ornan, and A. Itai. 1995. Learning morpholexical probabilities from an untagged corpus with an application to Hebrew. *Computational Linguistics*, 21(3):383–404.

D. C. Liu and J. Nocedal. 1989. On the limited memory method for large scale optimization. *Mathematical Programming B*, 45(3):503–28.

M. Maamouri, A. Bies, H. Jin, and T. Buckwalter. 2003. Arabic Treebank part 1 version 2.0. LDC2003T06.

L. Mangu, E. Brill, and A. Stolcke. 1999. Finding consensus among words: Lattice-based word error minimization. In *Proc. of ECSCT*.

E. Mráková and R. Sedlacek. 2003. From Czech morphology through partial parsing to disambiguation. In *Proc. of CLITP*.

F. Peng, F. Feng, and A. McCallum. 2004. Chinese segmentation and new word detection using conditional random fields. In *Proc. of COLING*.

V. Petkevič. 2001. Grammatical agreement and automatic morphological disambiguation of inflectional languages. In *Proc. of TSD*.

F. Sha and F. Pereira. 2003. Shallow parsing with conditional random fields. In *Proc. of HLT-NAACL*.

D. A. Smith and N. A. Smith. 2004. Bilingual parsing with factored estimation: Using English to parse Korean. In *Proc. of EMNLP*.

A. Smith, T. Cohn, and M. Osborne. 2005. Logarithmic opinion pools for conditional random fields. In *Proc. of ACL*.

C. Sutton and A. McCallum. 2005. Cliquewise training for undirected models. In *Proc. of UAI*.