Multi-rate HMMs for Word Alignment

Elif Eyigöz
Computer Science
University of Rochester
Rochester, NY 14627

Daniel Gildea
Computer Science
University of Rochester
Rochester, NY 14627

Kemal Oflazer
Computer Science
Carnegie Mellon University
PO Box 24866, Doha, Qatar

Abstract

We apply multi-rate HMMs, a tree structured HMM model, to the word-alignment problem. Multi-rate HMMs allow us to model reordering at both the morpheme level and the word level in a hierarchical fashion. This approach leads to better machine translation results than a morpheme-aware model that does not explicitly model morpheme reordering.

1 Introduction

We present an HMM-based word-alignment model that addresses transitions between morpheme positions and word positions simultaneously. Our model is an instance of a multi-scale HMM, a widely used method for modeling different levels of a hierarchical stochastic process. In multi-scale modeling of language, the deepest level of the hierarchy may consist of the phoneme sequence, and going up in the hierarchy, the next level may consist of the syllable sequence, and then the word sequence, the phrase sequence, and so on. By the same token, in the hierarchical word-alignment model we present here, the lower level consists of the morpheme sequence and the higher level the word sequence.

Multi-scale HMMs have a natural application in language processing due to the hierarchical nature of linguistic structures. They have been used for modeling text and handwriting (Fine et al., 1998), in signal processing (Willsky, 2002), knowledge extraction (Skounakis et al., 2003), as well as in other fields of AI such as vision (Li et al., 2006; Luettgen et al., 1993) and robotics (Theocharous et al., 2001). The model we propose here is most similar to multi-rate HMMs (Çetin et al., 2007), which were applied to a classification problem in industrial machine tool wear.

The vast majority of languages exhibit morphology to some extent, leading to various efforts in machine translation research to include morphology in translation models (Al-Onaizan et al., 1999; Niessen and Ney, 2000; Čmejrek et al., 2003; Lee, 2004; Chung and Gildea, 2009; Yeniterzi and Oflazer, 2010). For the word-alignment problem, Goldwater and McClosky (2005) and Eyigöz et al. (2013) suggested word alignment models that address morphology directly.

Eyigöz et al. (2013) introduced two-level alignment models (TAM), which adopt a hierarchical representation of alignment: the first level involves word alignment, the second level involves morpheme alignment. TAMs jointly induce word and morpheme alignments using an EM algorithm. TAMs can align rarely occurring words through their frequently occurring morphemes. In other words, they use morpheme probabilities to smooth rare word probabilities.

Eyigöz et al. (2013) introduced TAM 1, which is analogous to IBM Model 1, in that the first level is a bag of words in a pair of sentences, and the second level is a bag of morphemes. By introducing distortion probabilities at the word level, Eyigöz et al. (2013) defined the HMM extension of TAM 1, the TAM-HMM. TAM-HMM was shown to be superior to its single-level counterpart, i.e., the HMM-based word alignment model of Vogel et al. (1996).

The alignment example in Figure 1 shows a Turkish word aligned to an English phrase. The morphemes of the Turkish word are aligned to the English words. As the example shows, morphologically rich languages exhibit complex reordering phenomena at the morpheme level, which is left unutilized in TAM-HMMs. In this paper, we add morpheme sequence modeling to TAMs to capture morpheme level distortions. The example also shows that the Turkish morpheme or-

Proceedings of the Eighth Workshop on Statistical Machine Translation, pages 494–502,
Sofia, Bulgaria, August 8-9, 2013 ©2013 Association for Computational Linguistics

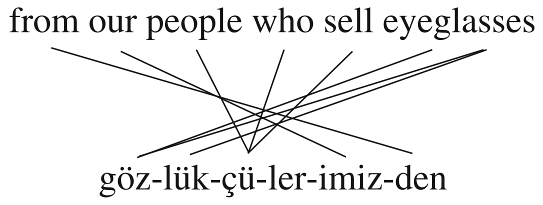

Figure 1: Turkish word aligned to an English phrase.

der is the reverse of the English word order. Because this pattern spans several English words, it can only be captured by modeling morpheme reordering across word boundaries. We chose multi-rate HMMs over other hierarchical HMM models because multi-rate HMMs allow morpheme sequence modeling across words over the entire sentence.

It is possible to model the morpheme sequence by treating morphemes as words: segmenting the words into morphemes, and using word-based word alignment models on the segmented data. Eyigöz et al. (2013) showed that TAM-HMM performs better than treating morphemes as words.

Since the multi-rate HMM allows both word and morpheme sequence modeling, it is a generalization of TAM-HMM, which allows only word sequence modeling. TAM-HMM in turn is a generalization of the model suggested by Goldwater and McClosky (2005) and TAM 1. Our results show that multi-rate HMMs are superior to TAM-HMMs. Therefore, multi-rate HMMs are the best two-level alignment models proposed so far.

2 Two-level Alignment Model (TAM)

The two-level alignment model (TAM) takes the approach of assigning probabilities to both word-to-word translations and morpheme-to-morpheme translations simultaneously, allowing morpheme-level probabilities to guide alignment for rare word pairs. TAM is based on a concept of alignment defined at both the word and morpheme levels.

2.1 Morpheme Alignment

A word alignment a_w is a function mapping a set of word positions in a target language sentence $\mathbf{e}$ to a set of word positions in a source language sentence $\mathbf{f}$, as exemplified in Figure 2. A morpheme alignment a_m is a function mapping a set of morpheme positions in a target language sentence to a set of morpheme positions in a source language sentence. A morpheme position is a pair of integers (j, k), which defines a word position j and a relative morpheme position k in the word at position j, as shown in Figure 3. The word and morpheme alignments below are depicted in Figures 2 and 3.

$$a_w(1) = 1 \quad a_m(2,1) = (1,1) \quad a_w(2) = 1$$

A morpheme alignment a_m and a word alignment a_w are *compatible* if and only if they satisfy the following conditions: If the morpheme alignment a_m maps a morpheme of e to a morpheme of f, then the word alignment a_w maps e to f. If the word alignment a_w maps e to f, then the morpheme alignment a_m maps at least one morpheme of e to a morpheme of f. If the word alignment a_w maps e to null, then all of its morphemes are mapped to null. Figure 3 shows a morpheme alignment that is compatible with, i.e., restricted by, the word alignment in Figure 2. The smaller boxes embedded inside the main box in Figure 3 depict the embedding of the morpheme level inside the word level in two-level alignment models (TAM).

2.2 TAM 1

We call TAM without sequence modeling TAM 1, because it defines an embedding of IBM Model 1 (Brown et al., 1993) for morphemes inside IBM Model 1 for words. In TAM 1, $p(\mathbf{e}|\mathbf{f})$, the probability of translating the sentence $\mathbf{f}$ into $\mathbf{e}$ is computed by summing over all possible word alignments and all possible morpheme alignments that are compatible with a given word alignment a_w:

$$\overbrace{R_w \prod_{j=1}^{|\mathbf{e}|} \sum_{i=0}^{|\mathbf{f}|}}^{\text{Word}} \left(t(e_j|f_i) \; \overbrace{R_m \prod_{k=1}^{|e_j|} \sum_{n=0}^{|f_i|}}^{\text{Morpheme}} t(e_j^k|f_i^n) \right) \tag{1}$$

where f_i^n is the n^{th} morpheme of the word at position i. The probability of translating the word f_i into the word e_j is computed by summing over all possible morpheme alignments between the morphemes of e_j and f_i. R_w substitutes $\frac{P(l_\mathbf{e}|l_\mathbf{f})}{(l_\mathbf{f}+1)^{l_\mathbf{e}}}$ for easy readability.[1] R_m is equivalent to R_w except

[1] $l_\mathbf{e} = |\mathbf{e}|$ is the number of words in sentence $\mathbf{e}$ and $l_\mathbf{f} = |\mathbf{f}|$.

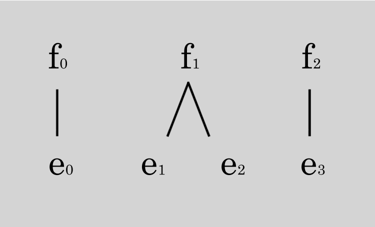

Figure 2: Word alignment

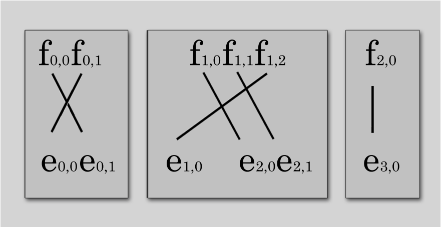

Figure 3: Morpheme alignment

for the fact that its domain is not the set of sentences but the set of words. The length of a word is the number of morphemes in the word. The length of words e_j and f_i in $R(e_j, f_i)$ are the number of morphemes of e_j and f_i. We assume that all unaligned morphemes in a sentence map to a special null morpheme.

TAM 1 with the contribution of both word and morpheme translation probabilities, as in Eqn. 1, is called 'word-and-morpheme' version of TAM 1. The model is technically deficient probabilistically, as it models word and morpheme translation independently, and assigns mass to invalid word/morpheme combinations. We can also define the 'morpheme-only' version of TAM 1 by canceling out the contribution of word translation probabilities and assigning 1 to $t(e_j|f_i)$ in Eqn. 1. Please note that, although this version of the two-level alignment model does not use word translation probabilities, it is also a word-aware model, as morpheme alignments are restricted to correspond to a valid word alignment. As such, it also allows for word level sequence modeling by HMMs. Finally, canceling out the contribution of morpheme translation probabilities reduces TAM 1 to IBM Model 1. Just as IBM Model 1 is used for initialization before HMM-based word-alignment models (Vogel et al., 1996; Och and Ney, 2003), TAM Model 1 is used to initialize its HMM extensions, which are described in the next section.

3 Multi-rate HMM

Like other multi-scale HMM models such as hierarchical HMM's (Fine et al., 1998) and hidden Markov trees (Crouse et al., 1998), the multi-rate HMM characterizes the inter-scale dependencies by a tree structure. As shown in Figure 5, scales are organized in a hierarchical manner from coarse to fine, which allows for efficient representation of both short- and long-distance context simultaneously.

We found that 51% of the dependency relations in the Turkish Treebank (Oflazer et al., 2003) are between the last morpheme of a dependent word and the first morpheme (the root) of the head word that is immediately to its right, which is exemplified below. The following examples show English sentences in Turkish word/morpheme order. The pseudo Turkish words are formed by concatenation of English morphemes, which are indicated by the '+' between the morphemes.

- - I will come from X.
  - X+ABL come+will+I

- - I will look at X.
  - X+DAT look+will+I

In English, the verb 'come' subcategorizes for a PP headed by 'from' in the example above. In the pseudo Turkish version of this sentence, 'come' subcategorizes for a NP marked with ablative case (ABL), which corresponds to the preposition 'from'. Similarly, 'look' subcategorizes for a PP headed by 'at' in English, and a NP marked with dative case (DAT) in Turkish. Just as the verb and the preposition that it subcategorizes for are frequently found adjacent to each other in English, the verb and the case that it subcategorizes for are frequently found adjacent to each other in Turkish. Thus, we have a pattern of three corresponding morphemes appearing in reverse order in English and Turkish, spanning two words in Turkish and three words in English. In order to capture such regularities, we chose multi-rate HMMs over other hierarchically structured HMM models because, unlike other models, multi-rate HMMs allow morpheme sequence modeling across words over the entire sentence. This allows us to capture morpheme-mediated syntactic relations between words (Eryiğit et al., 2008), as exemplified above.

Morpheme sequence modeling across words is shown in Figure 4 by the arrows after the nodes

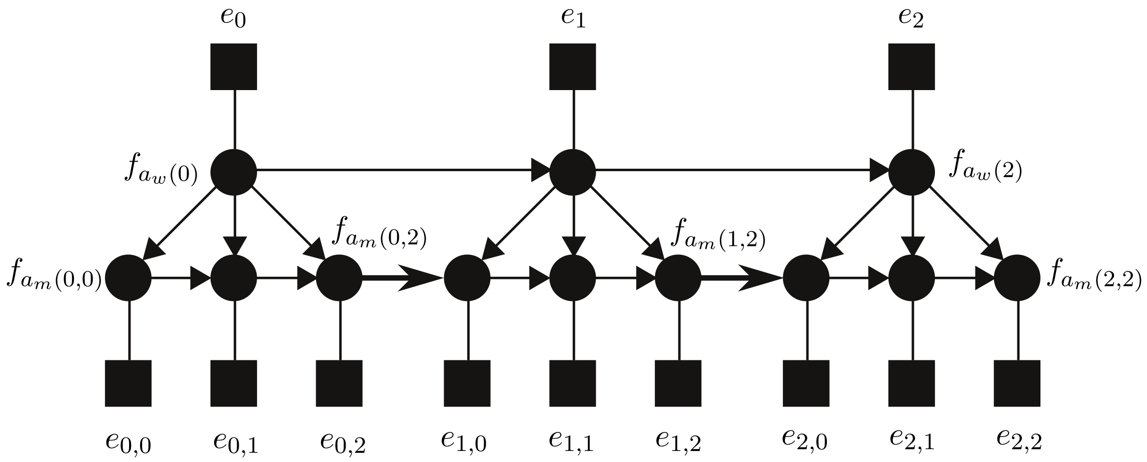

Figure 4: Multi-rate HMM graph.

representing $f_{a_m(0,2)}$ and $f_{a_m(1,2)}$. The circles represent the words and morphemes of the source language, the squares represent the words and morphemes of the target language. $e_{0,2}$ is the last morpheme of word e_0, and $e_{1,0}$ is the first morpheme of the next word e_1. $f_{a_m(1,0)}$ is conditioned on $f_{a_m(0,2)}$, which is in the previous word.

In order to model the morpheme sequence across words, we define the function $prev(j, k)$, which maps the morpheme position (j, k) to the previous morpheme position:

$$prev(j, k) = \begin{cases} (j, k-1) & \text{if } k > 1 \\ (j-1, |e_{j-1}|) & \text{if } k = 1 \end{cases}$$

If a morpheme is the first morpheme of a word, then the previous morpheme is the last morpheme of the previous word.

3.1 Transitions

3.1.1 Morpheme transitions

Before introducing the morpheme level transition probabilities, we first restrict morpheme level transitions according to the assumptions of our model. We consider only the morpheme alignment functions that are compatible with a word alignment function. If we allow unrestricted transitions between morphemes, then this would result in some morpheme alignments that do not allow a valid word alignment function.

To avoid this problem, we restrict the transition function as follows: at each time step, we allow transitions between morphemes in sentence $\mathbf{f}$ if the morphemes belong to the same word. This restriction reduces the transition matrix to a block diagonal matrix. The block diagonal matrix $\mathbf{A^b}$ below is a square matrix which has blocks of square matrices $\mathbf{A}_1 \cdots \mathbf{A}_n$ on the main diagonal, and the off-diagonal values are zero.

$$\mathbf{A^b} = \begin{bmatrix} \mathbf{A}_0 & 0 & \cdots & 0 \\ 0 & \mathbf{A}_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathbf{A}_n \end{bmatrix}$$

The square blocks $\mathbf{A}_0, \ldots, \mathbf{A}_n$ have the dimensions $|f_0|, \ldots, |f_n|$, the length of the words in sentence $\mathbf{f}$. In each step of the forward-backward algorithm, multiplying the forward (or backward) probability vectors with the block diagonal matrix restricts morpheme transitions to occur only within the words of sentence $\mathbf{f}$.

In order to model the morpheme sequence across words, we also allow transitions between morphemes across the words in sentence $\mathbf{f}$. However, we allow cross-word transitions only at certain time steps: between the last morpheme of a word in sentence $\mathbf{e}$ and the first morpheme of the next word in sentence $\mathbf{e}$. This does not result in morpheme alignments that do not allow a valid word alignment function. Instead of the block diagonal matrix $\mathbf{A^b}$, we use a transition matrix $\mathbf{A}$ which is not necessarily block diagonal, to model morpheme transitions across words.

In sum, we multiply the forward (or backward) probability vectors with either the transition matrix $\mathbf{A^b}$ or the transition matrix $\mathbf{A}$, depending on whether the transition is occurring at the last morpheme of a word in $\mathbf{e}$. We introduce the function $\delta(p, q, r, s)$ to indicate whether a transition is allowed from source position (p, q) to source position (r, s) when advancing one target position:

$$\delta(p, q, r, s) = \begin{cases} 1 & \text{if } p = r \text{ or } s = 1 \\ 0 & \text{otherwise} \end{cases}$$

Morpheme transition probabilities have four components. First, the δ function as described above. Second, the jump width:

$$\mathcal{J}(p, q, r, s) = abs(r, s) - abs(p, q)$$

where $abs(j, k)$ maps a word-relative morpheme position to an absolute morpheme position, i.e., to the simple left-to-right ordering of a morpheme in a sentence. Third, the morpheme class of the previous morpheme:[2]

$$\mathcal{M}(p, q) = Class(f_p^q)$$

Fourth, as the arrow from $f_{a_w(0)}$ to $f_{a_m(0,0)}$ in Figure 4 shows, there is a conditional dependence on the word class that the morpheme is in:

$$\mathcal{W}(r) = Class(f_r)$$

Putting together these components, the morpheme transitions are formulated as follows:

$$p(a_m(j, k) = (r, s) \mid a_m(prev(j, k)) = (p, q)) \propto \\ p\big(\mathcal{J}(p, q, r, s) | \mathcal{M}(p, q), \mathcal{W}(r)\big) \ \delta(p, q, r, s) \tag{2}$$

The block diagonal matrix $\mathbf{A^b}$ consists of morpheme transition probabilities.

3.1.2 Word transitions

In the multi-rate HMM, word transition probabilities have two components. First, the jump width:

$$\mathcal{J}(p, r) = r - p$$

Second, the word class of the previous word:

$$\mathcal{W}(p) = Class(f_p)$$

The jump width is conditioned on the word class of the previous word:

$$p(a_w(j) = r \mid a_w(j-1) = p) \propto \\ p(\mathcal{J}(p, r) \mid \mathcal{W}(p)) \tag{3}$$

The transition matrix $\mathbf{A}$, which is not necessarily block diagonal, consists of values which are the product of a morpheme transition probability, as defined in Eqn. 2, and a word transition probability, as defined in Eqn. 3.

3.2 Probability of translating a sentence

Finally, putting together Eqn. 1, Eqn. 2 and Eqn. 3, we formulate the probability of translating a sentence $p(\mathbf{e}|\mathbf{f})$ as follows:

$$R_w \sum_{a_w} \prod_{j=1}^{|e|} \Bigg(t(e_j | f_{a_w(j)}) p(a_w(j) | a_w(j-1)) \\ R_m \sum_{a_m} \prod_{k=1}^{|e_j|} t(e_{j,k} | f_{a_m(j,k)}) \\ p(a_m(j,k) | a_m(prev(j,k))) \Bigg)$$

R_w is the same as it is in Eqn. 1, whereas $R_m = P(l_e | l_f)$. If we cancel out morpheme transitions by setting $p(a_m(j, k) | a_m(prev(j, k))) = {}^1/_{|f_{a_m(j,k)}|}$, i.e., with a uniform distribution, then we get TAM with only word-level sequence modeling, which we call TAM-HMM.

The complexity of the multi-rate HMM is $O(m^3 n^3)$, where n is the number of words, and m is the number of morphemes per word. TAM-HMM differs from multi-rate HMM only by the lack of morpheme-level sequence modeling, and has complexity $O(m^2 n^3)$.

For the HMM to work correctly, we must handle jumping to and jumping from null positions. We learn the probabilities of jumping to a null position from the data. To compute the transition probability from a null position, we keep track of the nearest previous source word (or morpheme) that does not align to null, and use the position of the previous non-null word to calculate the jump width. In order to keep track of the previous non-null word, we insert a null word between words (Och and Ney, 2003). Similarly, we insert a null morpheme after every non-null morpheme.

3.3 Counts

We use Expectation Maximization (EM) to learn the word and morpheme translation probabilities, as well as the transition probabilities of the reordering model. This is done with forward-backward training at the morpheme level, collecting translation and transition counts for both the word and the morphemes from the morpheme-level trellis.

In Figure 5, the grid on the right depicts the morpheme-level trellis. The grid on the left is the abstraction of the word-level trellis over the

[2] We used the `mkcls` tool in GIZA (Och and Ney, 2003) to learn the word and the morpheme classes.

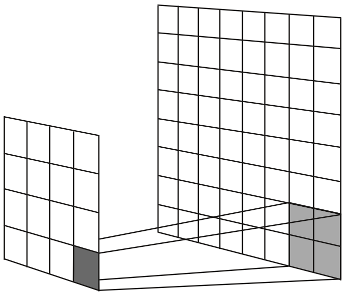

Figure 5: Multi-rate HMM trellis

morpheme-level trellis. For each target word e and for each source word f, there is a small HMM trellis with dimensions $|e| \times |f|$ inside the morpheme-level trellis, as shown by the shaded area inside the grid on the right. We collect counts for words by summing over the values in the small HMM trellis associated with the words.

3.3.1 Translation counts

Morpheme translation counts We compute expected counts over the morpheme-level trellis. The morpheme translation count function below collects expected counts for a morpheme pair (h, g) in a sentence pair $(\mathbf{e}, \mathbf{f})$:

$$c_m(h|g; \mathbf{e}, \mathbf{f}) = \sum_{\substack{(j,k) \\ \text{s.t.} \\ h=e_j^k}} \sum_{\substack{(p,q) \\ \text{s.t.} \\ g=f_p^q}} \gamma_{j,k}(p, q)$$

where $\gamma_{j,k}(p, q)$ stands for the posterior morpheme translation probabilities for source position (p, q) and target position (i, j) that are computed with the forward-backward algorithm.

Word translation counts For each target word e and source word f, we collect word translation counts by summing over posterior morpheme translation probabilities that are in the small trellis associated with e and f.

Since δ allows only within-word transitions to occur inside the small trellis, the posterior probability of observing the word e given the word f is preserved across time points within the small trellis associated with e and f. In other words, the sum of the posterior probabilities in each column of the small trellis is the same. Therefore, we collect word translation counts only from the last morphemes of the words in $\mathbf{e}$.

The word translation count function below collects expected counts from a sentence pair $(\mathbf{e}, \mathbf{f})$ for a particular source word f and target word e:

$$c_w(e|f; \mathbf{e}, \mathbf{f}) = \sum_{\substack{j \\ \text{s.t.} \\ e=e_j}} \sum_{\substack{p \\ \text{s.t.} \\ f=f_p}} \sum_{1 \le q \le |f|} \gamma_{j,|e|}(p, q)$$

3.3.2 Transition counts

Morpheme transition counts For all target positions (j, k) and all pairs of source positions (p, q) and (r, s), we compute morpheme transition posteriors:

$$\xi_{j,k}((p, q), (r, s))$$

using the forward-backward algorithm. These expected counts are accumulated to estimate the morpheme jump width probabilities $p\big(\mathcal{J}(p, q, r, s)|\mathcal{M}(p, q), \mathcal{W}(r)\big)$ used in Eqn. 2.

Word transition counts We compute posterior probabilities for word transitions by summing over morpheme transition posteriors between the morphemes of the words f_l and f_n:

$$\xi_j(p, r) = \sum_{1 \le q \le |f_p|} \sum_{1 \le s \le |f_r|} \xi_{j,|e_j|}((p, q), (r, s))$$

Like the translation counts, the transition counts are collected from the last morphemes of words in $\mathbf{e}$. These expected counts are accumulated to estimate the word jump width probabilities $p(\mathcal{J}(p, r) \mid \mathcal{W}(p))$ used in Eqn. 3.

Finally, $R_m = P(l_e|l_f)$ does not cancel out in the counts of the multi-rate HMM. To compute the conditional probability $P(l_e|l_f)$, we assume that the length of word e varies according to a Poisson distribution with a mean that is linear with length of the word f (Brown et al., 1993).

3.4 Variational Bayes

In order to prevent overfitting, we use the Variational Bayes extension of the EM algorithm (Beal, 2003). This amounts to a small change to the M step of the original EM algorithm. We introduce Dirichlet priors α to perform an inexact normalization by applying the function $f(v) = \exp(\psi(v))$ to the expected counts collected in the E step, where ψ is the digamma function (Johnson, 2007). The M-step update for a multinomial parameter $\theta_{x|y}$ becomes:

$$\theta_{x|y} = \frac{f(E[c(x|y)] + \alpha)}{f(\sum_j E[c(x_j|y)] + \alpha)}$$

		Multi-rate HMM	TAM-HMM		WORD	
			Word-Morph	Morph only	IBM 4	Baseline
BLEU	TR to EN	**30.82**	29.48	29.98	29.13	27.91
	EN to TR	**23.09**	22.55	22.54	21.95	21.82
AER		0.254	0.255	0.256	0.375	0.370

Table 1: AER and BLEU Scores

We set α to 10^{-20}, a very low value, to have the effect of anti-smoothing, as low values of α cause the algorithm to favor words which co-occur frequently and to penalize words that co-occur rarely. We used Dirichlet priors on morpheme translation probabilities.

4 Experiments and Results

4.1 Data

We trained our model on a Turkish-English parallel corpus of approximately 50K sentences which have a maximum of 80 morphemes. Our parallel data consists mainly of documents in international relations and legal documents from sources such as the Turkish Ministry of Foreign Affairs, EU, etc. The Turkish data was first morphologically parsed (Oflazer, 1994), then disambiguated (Sak et al., 2007) to select the contextually salient interpretation of words. In addition, we removed morphological features that are not explicitly marked by an overt morpheme. For English, we use part-of-speech tagged data. The number of English words is 1,033,726 and the size of the English vocabulary is 28,647. The number of Turkish words is 812,374, the size of the Turkish vocabulary is 57,249. The number of Turkish morphemes is 1,484,673 and the size of the morpheme vocabulary is 16,713.

4.2 Experiments

We initialized our implementation of the single level 'word-only' model, which we call 'baseline' in Table 1, with 5 iterations of IBM Model 1, and further trained the HMM extension (Vogel et al., 1996) for 5 iterations. Similarly, we initialized TAM-HMM and multi-rate HMM with 5 iterations of TAM 1 as explained in Section 2.2. Then we trained TAM-HMM and the multi-rate HMM for 5 iterations. We also ran GIZA++ (IBM Model 1–4) on the data. We translated 1000 sentence test sets.

We used Dirichlet priors in both IBM Model 1 and TAM 1 training. We experimented with using Dirichlet priors on the HMM extensions of both IBM-HMM and TAM-HMM. We report the best results obtained for each model and translation direction.

We evaluated the performance of our model in two different ways. First, we evaluated against gold word alignments for 75 Turkish-English sentences. Table 1 shows the AER (Och and Ney, 2003) of the word alignments; we report the grow-diag-final (Koehn et al., 2003) of the Viterbi alignments. Second, we used the Moses toolkit (Koehn et al., 2007) to train machine translation systems from the Viterbi alignments of our various models, and evaluated the results with BLEU (Papineni et al., 2002).

In order to reduce the effect of nondeterminism, we run Moses three times per experiment setting, and report the highest BLEU scores obtained. Since the BLEU scores we obtained are close, we did a significance test on the scores (Koehn, 2004). In Table 1, the colors partition the table into equivalence classes: If two scores within the same row have different background colors, then the difference between their scores is statistically significant. The best scores in the leftmost column were obtained from multi-rate HMMs with Dirichlet priors only during the TAM 1 training. On the contrary, the best scores for TAM-HMM and the baseline-HMM were obtained with Dirichlet priors both during the TAM 1 and the TAM-HMM training. In Table 1, as the scores improve gradually towards the left, the background color gets gradually lighter, depicting the statistical significance of the improvements. The multi-rate HMM performs better than the TAM-HMM, which in turn performs better than the word-only models.

5 Conclusion

We presented a multi-rate HMM word alignment model, which models the word and the morpheme sequence simultaneously. We have tested our model on the Turkish-English pair and showed that our model is superior to the two-level word alignment model which has sequence modeling only at the word level.

Acknowledgments Partially funded by NSF award IIS-0910611. Kemal Oflazer acknowledges the generous support of the Qatar Foundation through Carnegie Mellon University's Seed Research program. The statements made herein are solely the responsibility of this author(s), and not necessarily that of Qatar Foundation.

References

Yaser Al-Onaizan, Jan Curin, Michael Jahr, Kevin Knight, John Lafferty, Dan Melamed, Franz-Josef Och, David Purdy, Noah A. Smith, and David Yarowsky. 1999. Statistical machine translation. Technical report, Final Report, JHU Summer Workshop.

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

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

Özgür Çetin, Mari Ostendorf, and Gary D. Bernard. 2007. Multirate coupled Hidden Markov Models and their application to machining tool-wear classification. *IEEE Transactions on Signal Processing*, 55(6):2885–2896, June.

Tagyoung Chung and Daniel Gildea. 2009. Unsupervised tokenization for machine translation. In *EMNLP*, pages 718–726.

Martin Čmejrek, Jan Cuřín, and Jiří Havelka. 2003. Czech-English dependency-based machine translation. In *EACL*, pages 83–90.

Matthew Crouse, Robert Nowak, and Richard Baraniuk. 1998. Wavelet-based statistical signal processing using Hidden Markov Models. *IEEE Transactions on Signal Processing*, 46(4):886–902.

Gülşen Eryiğit, Joakim Nivre, and Kemal Oflazer. 2008. Dependency parsing of Turkish. *Computational Linguistics*, 34(3):357–389.

Elif Eyigöz, Daniel Gildea, and Kemal Oflazer. 2013. Simultaneous word-morpheme alignment for statistical machine translation. In *NAACL*.

Shai Fine, Yoram Singer, and Naftali Tishby. 1998. The hierarchical Hidden Markov model: Analysis and applications. *Machine Learning*, 32(1):41–62, July.

Sharon Goldwater and David McClosky. 2005. Improving statistical MT through morphological analysis. In *HLT-EMNLP*.

Mark Johnson. 2007. Why doesn't EM find good HMM POS-taggers? In *EMNLP-CoNLL*, pages 296–305, Prague, Czech Republic, June. Association for Computational Linguistics.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *HLT-NAACL*.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *ACL*, pages 177–180.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *EMNLP*, pages 388–395.

Young-suk Lee. 2004. Morphological analysis for statistical machine translation. In *HLT-NAACL*, pages 57–60.

Jia Li, Robert Gray, and Richard Olshen. 2006. Multiresolution image classification by hierarchical modeling with two-dimensional Hidden Markov Models. *IEEE Transactions on Information Theory*, 46(5):1826–1841, September.

Mark R. Luettgen, William C. Karl, Alan S. Willsky, and Robert R. Tenney. 1993. Multiscale representations of Markov Random Fields. *IEEE Transactions on Signal Processing*, 41(12):3377–3396.

Sonja Niessen and Hermann Ney. 2000. Improving SMT quality with morpho-syntactic analysis. In *COLING*, pages 1081–1085.

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

Kemal Oflazer, Bilge Say, Dilek Z. Hakkani-Tür, and Gökhan Tür. 2003. Building a Turkish treebank. In A. Abeillé, editor, *Treebanks: Building and Using Parsed Corpora*, pages 261–277. Kluwer, London.

Kemal Oflazer. 1994. Two-level description of Turkish morphology. *Literary and Linguistic Computing*, 9(2).

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Conference of the Association for Computational Linguistics (ACL-02)*, pages 311–318.

Haşim Sak, Tunga Güngör, and Murat Saraçlar. 2007. Morphological disambiguation of Turkish text with perceptron algorithm. In *CICLing*, pages 107–118.

Marios Skounakis, Mark Craven, and Soumya Ray. 2003. Hierarchical Hidden Markov Models for information extraction. In *International Joint Conference on Artificial Intelligence*, volume 18, pages 427–433.

Georgios Theocharous, Khashayar Rohanimanesh, and Sridhar Maharlevan. 2001. Learning hierarchical observable Markov decision process Models for robot navigation. In *ICRA 2001*, volume 1, pages 511–516.

Stephan Vogel, Hermann Ney, and Christoph Tillmann. 1996. HMM-based word alignment in statistical translation. In *COLING*, pages 836–841.

Alan S. Willsky. 2002. Multiresolution Markov Models for signal and image processing. In *Proceedings of the IEEE*, pages 1396–1458.

Reyyan Yeniterzi and Kemal Oflazer. 2010. Syntax-to-morphology mapping in factored phrase-based statistical machine translation from English to Turkish. In *ACL 2010*, pages 454–464.