A Word Reordering Model for Improved Machine Translation

Karthik Visweswariah
IBM Research India
Bangalore, India
v-karthik@in.ibm.com

Rajakrishnan Rajkumar
Dept. of Linguistics
Ohio State University
raja@ling.osu.edu

Ankur Gandhe
IBM Research India
Bangalore, India
ankugand@in.ibm.com

Ananthakrishnan Ramanathan
IBM Research India
Bangalore, India
aramana2@in.ibm.com

Jiri Navratil
IBM T.J. Watson Research Center
Yorktown Heights, New York
jiri@us.ibm.com

Abstract

Preordering of source side sentences has proved to be useful in improving statistical machine translation. Most work has used a parser in the source language along with rules to map the source language word order into the target language word order. The requirement to have a source language parser is a major drawback, which we seek to overcome in this paper. Instead of using a parser and then using rules to order the source side sentence we learn a model that can directly reorder source side sentences to match target word order using a small parallel corpus with high-quality word alignments. Our model learns pairwise costs of a word immediately preceding another word. We use the Lin-Kernighan heuristic to find the best source reordering efficiently during training and testing and show that it suffices to provide good quality reordering.

We show gains in translation performance based on our reordering model for translating from Hindi to English, Urdu to English (with a public dataset), and English to Hindi. For English to Hindi we show that our technique achieves better performance than a method that uses rules applied to the source side English parse.

1 Introduction

Languages differ in the way they order words to produce sentences representing the same meaning. Machine translation systems need to reorder words in the source sentence to produce fluent output in the target language that preserves the meaning of the source sentence.

Current phrase based machine translation systems can capture short range reorderings via the phrase table. Even the capturing of these local reordering phenomena is constrained by the amount of training data available. For example, if adjectives precede nouns in the source language and follow nouns in the target language we still need to see a particular adjective noun pair in the parallel corpus to handle the reordering via the phrase table. Phrase based systems also rely on the target side language model to produce the right target side order. This is known to be inadequate (Al-Onaizan and Papineni, 2006), and this inadequacy has spurred various attempts to overcome the problem of handling differing word order in languages.

One approach is through distortion models, that try to model which reorderings are more likely than others. The simplest models just penalize long jumps in the source sentence when producing the target sentence. These models have also been generalized (Al-Onaizan and Papineni, 2006; Tillman, 2004) to allow for lexical dependencies on the source. While these models are simple, and can be integrated with the decoder they are insufficient to capture long-range reordering phenomena especially for language pairs that differ significantly.

The weakness of these simple distortion models has been overcome using syntax of either the source or target sentence (Yamada and Knight, 2002; Galley et al., 2006; Liu et al., 2006; Zollmann and Venugopal, 2006). While these methods have shown to be useful in improving machine translation perfor-

Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, pages 486–496,
Edinburgh, Scotland, UK, July 27–31, 2011.

mance they generally involve joint parsing of the source and target language which is significantly more computationally expensive when compared to phrase based translation systems. Another approach that overcomes this weakness, is to to reorder the source sentence based on rules applied on the source parse (either hand written or learned from data) both when training and testing (Collins et al., 2005; Genzel, 2010; Visweswariah et al., 2010).

In this paper we propose a novel method for dealing with the word order problem that is efficient and does not rely on a source or target side parse being available. We cast the word ordering problem as a Traveling Salesman Problem (TSP) based on previous work on word-based and phrased-based statistical machine translation (Tillmann and Ney, 2003; Zaslavskiy et al., 2009). Words are the cities in the TSP and the objective is to learn the distance between words so that the shortest tour corresponds to the ordering of the words in the source sentence in the target language. We show that the TSP distances for reordering can be learned from a small amount of high-quality word alignment data by means of pairwise word comparisons and an informative feature set involving words and part-of-speech (POS) tags adapted and extended from prior work on dependency parsing (McDonald et al., 2005b). Obtaining high-quality word alignments that we require for training is fairly easy compared with obtaining a treebank required to obtain parses for use in syntax based methods.

We show experimentally that our reordering model, even when used to reorder sentences for training and testing (rather than being used as an additional score in the decoder) improves machine translation performance for: Hindi $\rightarrow$ English, English $\rightarrow$ Hindi, and Urdu $\rightarrow$ English. Although Urdu is similar to Hindi from the point of reordering phenomena we include it in our experiments since there are publicly available datasets for Urdu-English. For English $\rightarrow$ Hindi we obtained better machine translation performance with our reordering model as compared to a method that uses reordering rules applied to the source side parse.

The rest of the paper is organized as follows. Section 2 reviews related work and places our work in context. Section 3 outlines reordering issues due to syntactic differences between Hindi and English. Section 4 presents our reordering model, Section 5 presents experimental results and Section 6 presents our conclusions and possible future work.

2 Related work

There have been several studies demonstrating improved machine translation performance by reordering source side sentences based on rules applied to the source side parse during training and decoding. Much of this work has used hand written rules and several language pairs have been studied e.g German to English (Collins et al., 2005), Chinese to English (Wang et al., 2007), English to Hindi (Ramanathan et al., 2009), English to Arabic (Badr et al., 2009) and Japanese to English (Lee et al., 2010). There have also been some studies where the rules are learned from the data (Genzel, 2010; Visweswariah et al., 2010; Xia and McCord, 2004). In addition there has been work (Yamada and Knight, 2002; Zollmann and Venugopal, 2006; Galley et al., 2006; Liu et al., 2006) which uses source and/or target side syntax in a Context Free Grammar framework which results in machine translation decoding being considered as a parsing problem. In this paper we propose a model that does not require either source or target side syntax while also preserving the efficiency of reordering techniques based on rules applied to the source side parse.

In work that is closely related to ours, (Tromble and Eisner, 2009) formulated word reordering as a Linear Ordering Problem (LOP), an NP-hard permutation problem. They learned LOP model weights capable of assigning a score to every possible permutation of the source language sentence from an aligned corpus by using a averaged perceptron learning model. The key difference between our model and the model in (Tromble and Eisner, 2009) is that while they learn costs of a word w_i appearing *anywhere* before w_j, we learn costs of w_i *immediately preceding* w_j. This results in more compact models and (as we show in Section 5) better models.

Our model results in us having to solve a TSP instance. The relation between the TSP and machine translation decoding has been explored before. (Knight, 1999) showed that TSP is a sub-class of MT decoding and thus established that the latter is NP-hard. (Zaslavskiy et al., 2009) casts phrase-based

decoding as a TSP and they show favorable speed performance trade-offs compared with *Moses*, an existing state-of-the-art decoder. In (Tillmann and Ney, 2003), a beam-search algorithm used for TSP is adapted to work with an IBM-4 word-based model and phrase-based model respectively. As opposed to calculating TSP distances from existing machine translation components (viz. the translation, distortion and language model probabilities) we *learn model weights* to reorder source sentences to match target word order using an informative feature set adapted from graph-based dependency parsing (McDonald et al., 2005a).

3 Hindi-English reordering issues

This section provides a brief survey of constructions that the two languages in question differ as well as have in common. (Ramanathan et al., 2009) notes the following divergences:

- English follows SVO order while Hindi follows SOV order
- English uses prepositions while Hindi uses post-positions
- Hindi allows greater word order freedom
- Hindi has a relatively richer case-marking system

In addition to these differences, (Visweswariah et al., 2010) mention the similarity in word order in the case of adjective noun sequences (*some books* vs. *kuch kitab*).

4 Reordering model

Consider a source sentence $\mathbf{w}$ consisting of a sequence of n words w_1, w_2, ... w_n that we would like to reorder into the target language order. Given a permutation π of the indices $1..n$, let the candidate reordering be w_{π_1}, w_{π_2}, ..., w_{π_n}. Thus, π_i denotes the index of the word in the source sentence that maps to position i in the candidate reordering. Clearly there are $n!$ such permutations. Our reordering model assigns costs to candidate permutations as:

$$C(\pi|\mathbf{w}) = \sum_i c(\pi_{i-1}, \pi_i).$$

The cost $c(m, n)$ can be thought of as the cost of the word at index m immediately preceding the word with index n in the candidate reordering. In this paper, we parametrize the costs as:

$$c(m, n) = \theta^T \Phi(\mathbf{w}, m, n),$$

where θ is a learned vector of weights and Φ is a vector of feature functions.

Given a source sentence $\mathbf{w}$ we reorder it according to the permutation π that minimizes the cost $C(\pi|\mathbf{w})$. Thus, we would like our cost function $C(\pi|\mathbf{w})$ to be such that the correct reordering π^* has the lowest cost of all possible reorderings π. In Section 4.1 we describe the features Φ that we use, and in Section 4.2 we describe how we train the weights θ to obtain a good reordering model.

Given our model structure, the minimization problem that we need to solve is identical to solving a Asymmetric Traveling Salesman Problem (ATSP) with each word corresponding to a city, and the costs $c(m, n)$ representing the pairwise distances between the cities. Consider the following example:

English input: John eats apples

Hindi: John seba(apples) khaataa hai(eats)

Desired reordered English: John apples eats

The ATSP that we need to solve is represented pictorially in Figure 1 with sample costs. Note that we have one extra node numbered 0. We start and end the tour at node 0, and this determines the first word in the reordered sentence. In this example the minimum cost tour is:

Start $\rightarrow$ *John* $\rightarrow$ *apple* $\rightarrow$ *eats*

recovering the right reordering for translation into Hindi.

Solving the ATSP (which is a well known NP hard problem) efficiently is crucial for the efficiency of our reordering model. To solve the ATSP, we first convert the ATSP to a symmetric TSP and then use the Lin-Kernighan heuristic as implemented in *Concorde*, a state-of-the-art TSP solver (Applegate et al., 2005). We also experimented with using the exact TSP solver in *Concorde* but since it was slower and did not improve performance we preferred using the Lin-Kernighan heuristic. To convert the ATSP to a symmetric TSP we double the size of the original problem creating a node $N^{'}$ for every node N in the original graph. Following (Hornik and

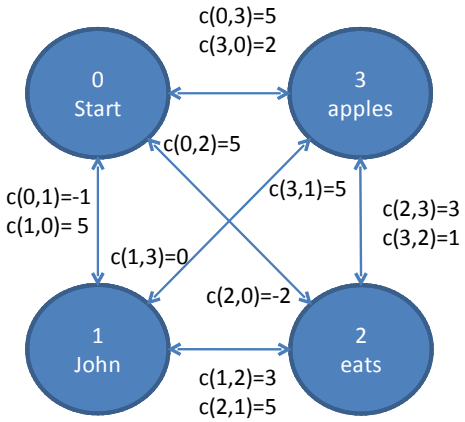

Figure 1: Example of an ATSP for reordering the sentence: *John eats apples*.

w_i	p_i	w_j	p_j
×	×	×	×
×			
	×		
		×	
			×
×	×		
×		×	
×			×
	×	×	
	×		×
		×	×
×	×		×

Table 1: Bigram feature templates used to calculate the cost that word at position i immediately precedes word at position j in the target word order. w_i (p_i) denotes the word (POS tag) at position i in the source sentence. Each of the templates is also conjoined with i-j the signed distance between the two words in the source sentence.

Hahsler, 2009), we then set new costs as follows: $c^{'}(A, B) = \infty$, $c^{'}(A, B^{'}) = c^{'}(B^{'}, A) = c(A, B)$ and $C(A, A^{'}) = -\infty$. Even with this doubling of the number of nodes, we observed that solving the TSPs with the Lin-Kernighan heuristic is very fast, taking roughly 10 milliseconds per sentence on average. Overall, this means that our reordering model is as fast as parsing and hence our model is comparable in performance to techniques based on applying rules to the parse tree.

4.1 Features

Since we would like to model reordering phenomena which are largely related to analyzing the syntax of the source sentence, we chose to use features based on those that have in the past been used for parsing (McDonald et al., 2005a). A subset of the features we use was also used for reordering in (Tromble and Eisner, 2009).

To be able to generalize from relatively small amounts of data, we use features that in addition to depending on the words in the input sentence $\mathbf{w}$ depend on the part-of-speech (POS) tags of the words in the input sentence. All features $\Phi(\mathbf{w}, i, j)$ we use are binary features, that fire based on the identities of the words and POS tags at or surrounding positions i and j in the source sentence. The first set of feature templates we use are given in Table 1. These features depend only on the identities of the word and POS tag of the two positions i and j and we call these Bigram features.

The second set of feature templates we use are given in Table 2. These features, in addition to examining positions i and j examine the surrounding positions. We instantiate these feature templates separately for the POS tag sequence and for the word sequence. We call these two feature sets ContextPOS and ContextWord respectively. When instantiated with POS tags, the first row of Table 2 looks at all POS tags between positions i and j. (Tromble and Eisner, 2009) use Bigram and ContextPOS features, while we extend their feature set with the use of ContextWord features. Since Hindi is verb final, in Hindi sentences with multiple verb groups it is rare for words with a verb in between to be placed together in the reordering to match English. Looking at the POS tags of words between positions i and j allows us to penalize such reorderings.

Each of the templates described in Table 1 and Table 2 is also conjoined with i-j the signed distance between the two words in the source sentence. The values of i-j between 5 and 10, and greater than 10 are quantized (negative values are similarly quantized).

In Section 5.2 we report on experiments showing the relative performance of these different feature

o_{i-1}	o_i	o_{i+1}	o_b	o_{j-1}	o_j	o_{j+1}
	×		×		×	
×	×				×	×
	×				×	×
×					×	×
×	×					×
×	×				×	
	×	×		×	×	
		×		×	×	
	×			×	×	
	×				×	
	×	×		×		
×	×			×	×	
×				×	×	
×	×			×		
	×	×			×	×
		×			×	×
	×	×				×

Table 2: Context feature templates used to calculate the cost that word at position i immediately precedes word at position j in the target word order. o_i denotes the observation at position i in the source sentence and o_b denotes an observation at a position between i and j (i.e $i+1 \leq b \leq j-1$). Each of the templates is instantiated with the observation sequence $\mathbf{o}$ taken to be the word sequence $\mathbf{w}$ and the POS tag sequence $\mathbf{p}$. Each of the templates is also conjoined with i-j the signed distance between the two positions in the source sentence.

types for the task of reordering Hindi sentences to be in English word order.

4.2 Training

To train the weights θ in our model, we need a collection of sentences, where we have the desired reference reordering $\pi^*(\mathbf{x})$ for each input sentence $\mathbf{x}$. To obtain these reference reorderings we use word aligned source-target sentence pairs. The quality and consistency of these reference reorderings will depend on the quality of the word alignments that we use. Given word aligned source and target sentences, we drop the source words that are not aligned. Let m_i be the mean of the target word positions that the source word at index i is aligned to. We then sort the source indices in increasing order of m_i. If $m_i = m_j$ (for example, because w_i and w_j are aligned to the same set of words) we keep them in the same order that they occurred in the source sentence. Obtaining the target ordering in this manner, is certainly not the only possible way and we would like to explore better treatment of this in future work.

We used the single best Margin Infused Relaxed Algorithm (MIRA) ((McDonald et al., 2005b), (Crammer and Singer, 2003)) with the online updates to our parameters being given by

$$\begin{aligned} \theta_{i+1} = & \arg\min_{\theta} ||\theta - \theta_i|| \\ & s.t. \quad C(\pi^*|\mathbf{w}) < C(\hat{\pi}|\mathbf{w}) - L(\pi^*, \hat{\pi}). \end{aligned}$$

In the equation above,

$$\hat{\pi} = \arg\min_{\pi} C(\pi|\mathbf{x})$$

is the best reordering based on the current parameter value and L is a loss function. We take the loss of a reordering to be the number of words for which the preceding word is wrong relative to the reference target order.

We also experimented with the averaged perceptron algorithm (Collins, 2002), but found single best MIRA to work slightly better and hence used MIRA for all our experiments.

5 Experiments

In this section we report on experiments to evaluate our reordering model. The first method we use for evaluation (monolingual BLEU) is by generating the desired reordering of the source sentence (as described in Section 4.2) and compare the reordered output to this desired reordered sentence using the BLEU metric. In addition, to these monolingual BLEU results, we also evaluate (in Section 5.5) the reordering by its effect on eventual machine translation performance.

We note that our reordering techniques uses POS information for the input sentence. The POS taggers used in this paper are Maximum Entropy Markov models trained using manually annotated POS corpora. For Hindi, we used roughly fifty thousand words with twenty six tags from the corpus described in (Dalal et al., 2007). For Urdu we used roughly fifty thousand words and forty six tags from the CRULP corpus (Hussain, 2008) and for English we used the Wall Street Journal section of the Penn Treebank.

5.1 Reordering model training data and alignment quality

To train our reordering models we need training data where we have the input source language sentence and the desired reordering in the target language. As described in Section 4.2 we derive the reference reordered sentence using word alignments. Table 3 presents our monolingual BLEU results for Hindi to English reordering as the source of the word alignments is varied. All results in Table 3 are with Bigram and ContextPOS features. We have word alignments from three sources: A small set of hand aligned sentences, HMM alignments (Vogel et al., 1996) and alignments obtained using a supervised Maximum Entropy aligner (Ittycheriah and Roukos, 2005) trained on the hand alignments. The F-measure for the HMM alignments were 65% and 78% for the Maximum Entropy model alignments. We see that the quality of the alignments is an important determiner of reordering performance. Row 1 shows the BLEU for unreordered (baseline) Hindi compared with the Hindi sentences reordered in English Order. Using just HMM alignments to train our model we do worse than unreordered Hindi. Although using the Maximum Entropy alignments is better than using HMM alignments, we do not improve upon a small number of hand alignments by using all the Maximum Entropy alignments.

To improve upon the model trained with only hand alignments we selected a small number of snippets of sentences from our Maximum Entropy alignments. The goal was to pick parts of sentences where the alignment is reliable enough to use for training. The heuristic we used in the selection of snippets was to pick maximal snippets of at least 7 consecutive Hindi words with all Hindi words aligned to a consecutive span of English words, with no unaligned English words in the span and no English words aligned to Hindi words outside the span. Adding snippets selected with this heuristic improves the reordering performance of our model as seen in the last row of Table 3.

5.2 Feature set comparison

In this section we report on experiments to determine the performance of the different classes of features (Bigram, ContextPos and ContextWord) discussed in Section 4.1. Table 4 shows monolingual BLEU results for training with different features sets for Hindi to English reordering. In all cases, we use a set of 6000 sentence pairs which were hand aligned to generate the training data. It is clear that all three sets of features contribute to performance of the reordering model, however the number of ContextWord features is larger than the number of Bigram and ContextPOS features put together, and it may be desirable to select from this set of features especially when training on large amounts of data.

HMM	MaxEnt	Hand	BLEU
-	-	-	35.9
220K	-	-	35.4
-	220K	-	47.0
-	220K	6K	48.4
-	-	6K	49.0
-	Good 17K	6K	**51.3**

Table 3: Monolingual BLEU scores for Hindi to English reordering using models trained on different alignment types and tested on a development set of 280 Hindi sentences (5590 tokens).

Feature template			
Bigram	ContextPOS	ContextWord	BLEU
-	-	-	35.9
×	-	-	43.8
×	×	-	49.0
×	×	×	**51.3**

Table 4: Monolingual BLEU scores for Hindi to English reordering using models trained with different feature sets and tested on a development set of 280 Hindi sentences (5590 tokens).

5.3 Monolingual reordering comparisons

Table 5 compares our reordering model with a reimplementation of the reordering model proposed in (Tromble and Eisner, 2009). Both the models use exactly the same features (bigram features and ContextPOS features) and are trained on the same data. To generate our training data, for Hindi to English and English to Hindi we use a set of 6000 hand aligned sentences, for Urdu to English we use a set of 8500 hand aligned sentences and for English to French we use a set of 10000 hand aligned sentences (a subset of Europarl and Hansards corpus). Our

Language pair		Monolingual BLEU		
Source	Target	Unreordered	LOP	TSP
Hindi	English	35.9	36.6	**49.0**
English	Hindi	34.4	48.4	**56.7**
Urdu	English	35.6	39.5	**49.9**
English	French	64.4	78.2	**81.2**

Table 5: Monolingual BLEU scores comparing the original source order with desired target reorder without reordering, and reordering using our model (TSP) and the model proposed in (Tromble and Eisner, 2009) (LOP).

test data consisted of 280 sentences for Hindi to English and 400 sentences for all other language pairs generated from hand aligned sentences. We include English-French here to compare on a fairly similar language pair with local reordering phenomena (the main difference being that in French adjectives generally follow nouns). We note that our model outperforms the model proposed in (Tromble and Eisner, 2009) in all cases.

5.4 Analysis of reordering performance

To get a feel for the qualitative performance of our reordering algorithm and the kind of phenomena it is able to capture, we analyze the reordering performance in terms of (i) whether the clause restructuring is done correctly – these can be thought of as medium-to-long range reorderings, (ii) whether clause boundaries are respected, and (iii) whether local (short range) reordering is performed correctly. The following analysis is for Hindi to English reordering with the best model (this is also the model used for Machine Translation experiments reported on in Section 5.5).

- **Clause structure**: As discussed in Section 3, the canonical clause order in Hindi is SOV, while in English it is SVO. However, variations on this structure are possible and quite frequent (e.g., clauses with two objects). To evaluate clause restructuring, we compared sequences of subjects, objects and verbs in the output and reference reorderings.

  We had a set of 70 sentences annotated with subject, direct object, indirect object and verb information – these annotations were made on the head word of each phrase, and the comparisons were on sequences of these words alone and not the entire constituent phrase. 52 sentences were reordered by the model to match the order of the corresponding reference. Eight sentences were ordered correctly but differently from the reference, because the reference was expressed in non-canonical fashion (e.g., in the passive) – note that these cases negatively impact the monolingual BLEU score. The following example shows a sentence being reordered correctly, where, however, the reference is expressed differently (note the position of the subject "policy" (*niiti*) in the reference and the reordered output) [1]:

  Input: aba$_1$ (now) taka$_2$ (till) aisii$_3$ (this) *niiti*$_4$ (policy) kabhii$_5$ (ever) nahii$_6$ (not) rahii$_7$ (has) hai$_8$ (been)

  Reordered: taka$_2$ (till) aba$_1$ (now) aisii$_3$ (this) *niiti*$_4$ (policy) hai$_8$ (been) kabhii$_5$ (ever) nahii$_6$ (not) rahii$_7$ (has)

  Reference: taka$_2$ (till) aba$_1$ (now) aisii$_3$ (this) kabhii$_5$ (ever) nahii$_6$ (not) rahii$_7$ (has) hai$_8$ (been) *niiti*$_4$ (policy)

  English: Till now this never has been the *policy*

  The remaining ten sentences were reordered incorrectly. These errors are largely in clauses which deviate from the SVO order in some way – clauses with multiple subjects or objects, clauses with no object, etc.. For example, the following sentence with two subjects and objects corresponding to the verb *wearing* has not been reordered correctly.

  Input: sabhii$_1$ (all) purusha$_2$ (men) safeda$_3$ (white) evama$_4$ (and) mahilaaen$_5$ (women) kesariyaa$_6$ (saffron) vastra$_7$ (clothes) dhaarana$_8$ (wear) kiye$_9$ hue$_{10}$ (-ing) thiin$_{11}$ (were)

  Reordered: sabhii$_1$ (all) purusha$_2$ (men) safeda$_3$ (white) evama$_4$ (and) mahilaaen$_5$ (women) kesariyaa$_6$ (saffron) vastra$_7$ (clothes) dhaarana$_8$ (wear) thiin$_{11}$ (were) kiye$_9$ hue$_{10}$ (-ing)

  Reference: sabhii$_1$ (all) purusha$_2$ (men) thiin$_{11}$ (were) dhaarana$_8$ (wear) kiye$_9$ hue$_{10}$ (-

[1] The numeric subscripts in the examples indicate word positions in the input.

ing) safeda$_3$ (white) evama$_4$ (and) mahilaaen$_5$ (women) kesariyaa$_6$ (saffron)

English: All men were wearing white and the women saffron

The model possibly needs more data with patterns that deviate from the standard SOV order to learn to reorder them correctly. We could also add to the model, features pertaining to subject, object, etc.

- **Clause boundaries**: Measured on a set of 844 sentences which were marked with clause boundaries, 37 sentences (4.4 %) had reorderings that violated these boundaries. An example of such a clause-boundary violation is below:

  Input: main$_1$ (I) sarakaara$_2$ (government) kaa$_3$ (of) dhyaana$_4$ (attention) *maananiiya$_5$ (honourable) pradhaana$_6$ (prime) mantri$_7$ (minister) dvaaraa$_8$ (by) isa$_9$ (this) sabhaa$_{10}$ (house) me$_{11}$ (in) kiye$_{12}$ gaye$_{13}$ (made) isa$_{14}$ (this) vaade$_{15}$ (promise)* ki$_{16}$ ora$_{17}$ (towards) dilaanaa$_{18}$ (to bring) chaahuungaa$_{19}$ (would like)

  Reordered: main$_1$ (I) chahuungaa$_{19}$ (would like) dilaanaa$_{18}$ (to bring) kii$_{16}$ ora$_{17}$ (towards) *isa$_9$ (this) vaade$_{15}$ (promise) kiye$_{12}$ gaye$_{13}$ (made) dvaaraa$_8$ (by) maananiiya$_5$ (honourable) mantri$_7$ (minister) pradhaana$_6$ (prime)* dhyaana$_4$ (attention) kaa$_3$ (of) sarakaara$_2$ (government) *men$_{11}$ (in) isa$_{14}$ (this) sabhaa$_{10}$ (house)*

  Reference: main$_1$ (I) chahuungaa$_{19}$ (would like) dilaanaa$_{18}$ (to bring) dhyaana$_4$ (attention) kaa$_3$ (of) sarakaara$_2$ (government) *kii$_{16}$ ora$_{17}$ (towards) isa$_9$ (this) vaade$_{15}$ (promise) kiye$_{12}$ gaye$_{13}$ (made) dvaaraa$_8$ (by) maananiiya$_5$ (honourable) mantri$_7$ (minister) pradhaana$_6$ (prime) men$_{11}$ (in) isa$_9$ (this) sabhaa$_{10}$ (house)*

  English I would like to bring the attention of the government towards this promise *made by the honourable prime minister in this house*.

  Note how the italicized clause, which is kept together in the reference, is split up incorrectly in the reordered output. The proportion of such boundary violations is, however, quite low, because Hindi being a verb-final language, most clauses end with a verb and it is probably quite straightforward for the model to keep clauses separate. A clause boundary detection program should make it possible to eliminate the remaining errors.

- **Local reordering**: To estimate the short range reordering performance, we consider how often different POS bigrams in the input are reordered correctly. Here, we expect the model to reorder prepositions correctly, and to avoid any reordering that moves apart nouns and their adjectival pre-modifiers or components of compound nouns (see Section 3). Table 6 summarizes the reordering performance for these categories for a set of 280 sentences (same as the test set used in Section 5.1). Each row in Table 6 indicates the total number of correct instances for the pair, i.e., the number of instances of the pair in the reference (column titled *Total*), the number of instances that already appear in the correct order in the input (column *Input*), and the number that are ordered correctly by the reordering model (column *Reordered*). The first two rows show that adjective-noun and noun-noun (compounds) are in most cases correctly retained in the original order by the model. The final row shows that while many prepositions have been moved into their correct positions, there are still quite a few mismatches with the reference. An important reason why this happens is that nouns modified by prepositional phrases can often also be expressed as noun compounds. For example, *vidyuta (electricity) kii (of) aavashyakataaen (requirements)* in Hindi can be expressed either as "requirements of electricity" or "electricity requirements". The latter expression results in a match with the input (explaining many of the 104 correct orders in the input) and a mismatch with the model's reordering. The same problem in the training data would also adversely impact the learning of the preposition reordering rule.

POS pair	Total	Input	Reordered
adj-noun	234	192	196
noun-noun	46	44	42
prep-noun	436	104	250

Table 6: An analyis of reordering for a few POS bigrams

Language pair		BLEU	
Source	Target	Unreordered	Reordered
Hindi	English	14.7	**16.7**
Urdu	English	23.3	**24.8**
English	Hindi	20.7	**22.5**

Table 7: Translation performance without reordering (baseline) compared with performance after preordering with our reordering model.

5.5 Machine translation results

We now present experiments in incorporating the reordering model in machine translation systems. For all results presented here, we reorder the training and test data using the single best reordering based on our reordering model for each sentence. For each of the language pairs we evaluated, we trained Direct Translation Model 2 (DTM) systems (Ittycheriah and Roukos, 2007) with and without reordering and compared performance on test data. We note that the DTM system includes features that allow it to model lexicalized reordering phenomena. The reordering window size was set to +/-8 words for both the baseline and our reordered input. In our experiments, we left the word alignments fixed, i.e we reordered the existing word alignments rather than realigning the sentences after reordering. Redoing the word alignments with the reordered data could potentially give further small improvements. We note that we obtained better baseline performance using DTM systems than the standard Moses/Giza++ pipeline (e.g we obtained a BLEU of 14.9 for English to Hindi with a standard Moses/Giza++ pipeline). For all of our systems we used a combination of HMM (Vogel et al., 1996) and MaxEnt alignments (Ittycheriah and Roukos, 2005).

For our Hindi-English experiments we use a training set of roughly 250k sentences (5.5M words) consisting of the Darpa-TIDES dataset (Bojar et al., 2010) and an internal dataset from several domains but dominated by news. Our test set was roughly 1.2K sentences from the news domain with a single reference. To train our reordering model, we used roughly 6K alignments plus 17K snippets selected from MaxEnt alignments as described in Section 5.1 with bigram, ContextPOS and ContextWord features. The monolingual reordering BLEU (on the same data reported on in Section 5.3) was 54.0 for Hindi to English and 60.8 for English to Hindi.

For our Urdu-English experiments we used 70k sentences from the NIST MT-08 training corpus and used the MT-08 eval set for testing. We note that the MT-08 eval set has four references as compared to one reference for our Hindi-English test set. This largely explains the improved baseline performance for Urdu-English as compared to Hindi-English. We present averaged results for the Web and News part of the test sets. To train the reordering model we used 9K hand alignments and 11K snippets extracted from MaxEnt alignments as described in Section 5.1 with bigram, ContextPOS and ContextWord context feature. The monolingual reordering BLEU for the reordering model thus obtained (on the same data reported on in Section 5.3) was 52.7.

Table 7 shows that for Hindi to English, English to Hindi and for Urdu to English we see a gain of 1.5 - 2 BLEU points. For English $\rightarrow$ Hindi we also experimented with a system that uses rules (learned from the data using the methods described in (Visweswariah et al., 2010)) applied to a parse to reorder source side English sentences. This system had a BLEU score of 21.2, which is an improvement over the baseline, but our reordering model is better by 1.3 BLEU points.

An added benefit of our reordering model is that the decoder can be run with a smaller search space exploring only a small amount of reordering without losing accuracy but running substantially faster. Table 8 shows the variation in machine Hindi to English translation performance with varying skip size (this parameter sets the maximum number of words skipped during decoding, lower values are associated with a restricted decoder search space and increased speed).

skip	Unreordered	Reordered
2	12.2	16.7
4	13.4	16.7
8	14.7	16.4

Table 8: Translation performance with/without reordering with varying decoder search space.

6 Conclusion and future work

In this paper we presented a reordering model to reorder source language data to make it resemble the target language word order without using either a source or target parser. We showed consistent gains of up to 2 BLEU points in machine translation performance using this model to preorder training and test data. We show better performance compared to syntax based reordering rules for English to Hindi translation. Our model used only a part of speech tagger (sometimes trained with fairly small amounts of data) and a small corpus of word alignments. Considering the fact that treebanks required to build high quality parsers are costly to obtain, we think that our reordering model is a viable alternative to using syntax for reordering. We also note, that with the preordering based on our reordering model we can achieve the best BLEU scores with a much tighter search space in the decoder. Even accounting for the cost of finding the best reordering according to our model, this usually results in faster processing than if we did not have the reordering in place.

In future work we plan to explore using more data from automatic alignments, perhaps by considering a joint model for aligning and reordering. We would also like to explore doing away with the requirement of having a POS tagger, using completely unsupervised methods to class words. We currently only look at word pairs in calculating the loss function used in MIRA updates. We would like to investigate the use of other loss functions and their effect on reordering performance. We also would like to explore whether the use of scores from our reordering model directly in machine translation systems can improve performance relative to using just the single best reordering.

References

Yaser Al-Onaizan and Kishore Papineni. 2006. Distortion models for statistical machine translation. In *Proceedings of ACL*, ACL-44, pages 529–536, Morristown, NJ, USA. Association for Computational Linguistics.

David L. Applegate, Robert E. Bixby, Vasek Chvatal, and William J. Cook. 2005. Concorde tsp solver. In *http://www.tsp.gatech.edu/*.

Ibrahim Badr, Rabih Zbib, and James Glass. 2009. Syntactic phrase reordering for English-to-Arabic statistical machine translation. In *Proceedings of EACL*.

Ondrej Bojar, Pavel Stranak, and Daniel Zeman. 2010. Data issues in English-to-Hindi machine translation. In *LREC*.

Michael Collins, Philipp Koehn, and Ivona Kučerová. 2005. Clause restructuring for statistical machine translation. In *Proceedings of ACL*, pages 531–540, Morristown, NJ, USA. Association for Computational Linguistics.

Michael Collins. 2002. Discriminative training methods for hidden Markov models: theory and experiments with perceptron algorithms. In *Proceedings of EMNLP*.

K. Crammer and Y. Singer. 2003. Ultraconservative online algorithms for multiclass problems. *Journal of Machine Learning Research*.

Aniket Dalal, Kumar Nagaraj, Uma Sawant, Sandeep Shelke, and Pushpak Bhattacharyya. 2007. Building feature rich pos tagger for morphologically rich languages: Experiences in Hindi. In *Proceedings of International Conference on Natural Language Processing*.

M. Galley, J. Graehl, K. Knight, D. Marcu, S. DeNeefe, W. Wang, and I. Thayer. 2006. Scalable inference and training of context-rich syntactic translation models. In *Proceedings of ACL*.

D. Genzel. 2010. Automatically learning source-side reordering rules for large scale machine translation. In *Proceedings of the 23rd International Conference on Computational Linguistics*.

Kurt Hornik and Michael Hahsler. 2009. TSP–infrastructure for the traveling salesperson problem. *Journal of Statistical Software*, 23(i02).

Sarmad Hussain. 2008. Resources for Urdu language processing. In *Proceedings of the 6th Workshop on Asian Language Resources, IJCNLP'08*.

Abraham Ittycheriah and Salim Roukos. 2005. A maximum entropy word aligner for Arabic-English machine translation. In *Proceedings of HLT/EMNLP*, HLT '05, pages 89–96, Stroudsburg, PA, USA. Association for Computational Linguistics.

Abraham Ittycheriah and Salim Roukos. 2007. Direct translation model 2. In *Proceedings of HLT-NAACL*, pages 57–64.

Kevin Knight. 1999. Decoding complexity in word-replacement translation models. *Comput. Linguist.*, 25:607–615, December.

Young-Suk Lee, Bing Zhao, and Xiaoqian Luo. 2010. Constituent reordering and syntax models for English-to-Japanese statistical machine translation. In *COLING*.

Y. Liu, Q. Liu, and S. Lin. 2006. Tree-to-String alignment template for statistical machine translation. In *Proceedings of ACL*.

R. McDonald, K. Crammer, and F. Pereira. 2005a. Online large-margin training of dependency parsers. In *Proceedings of ACL*.

Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajič. 2005b. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of HLT*.

Ananthakrishnan Ramanathan, Hansraj Choudhary, Avishek Ghosh, and Pushpak Bhattacharyya. 2009. Case markers and morphology: addressing the crux of the fluency problem in English-Hindi smt. In *Proceedings of ACL-IJCNLP*.

Christoph Tillman. 2004. A unigram orientation model for statistical machine translation. In *Proceedings of HLT-NAACL*.

Christoph Tillmann and Hermann Ney. 2003. Word reordering and a dynamic programming beam search algorithm for statistical machine translation. *Computational Linguistics*, 29(1):97–133.

Roy Tromble and Jason Eisner. 2009. Learning linear ordering problems for better translation. In *Proceedings of EMNLP*.

Karthik Visweswariah, Jiri Navratil, Jeffrey Sorensen, Vijil Chenthamarakshan, and Nandakishore Kambhatla. 2010. Syntax based reordering with automatically derived rules for improved statistical machine translation. In *Proceedings of the 23rd International Conference on Computational Linguistics*.

Stephan Vogel, Hermann Ney, and Christoph Tillmann. 1996. HMM-based word alignment in statistical translation. In *Proceedings of the 16th conference on Computational Linguistics*.

Chao Wang, Michael Collins, and Philipp Koehn. 2007. Chinese syntactic reordering for statistical machine translation. In *Proceedings of EMNLP-CoNLL*.

Fei Xia and Michael McCord. 2004. Improving a statistical MT system with automatically learned rewrite patterns. In *COLING*.

Kenji Yamada and Kevin Knight. 2002. A decoder for syntax-based statistical mt. In *Proceedings of ACL*.

Mikhail Zaslavskiy, Marc Dymetman, and Nicola Cancedda. 2009. Phrase-based statistical machine translation as a traveling salesman problem. In *Proceedings of ACL-IJCNLP*.

Andreas Zollmann and Ashish Venugopal. 2006. Syntax augmented machine translation via chart parsing. In *Proceedings on the Workshop on Statistical Machine Translation*.