Source-side Preordering for Translation using Logistic Regression and Depth-first Branch-and-Bound Search[*]

Laura Jehl[⋆] **Adrià de Gispert**[‡] **Mark Hopkins**[‡] **William Byrne**[‡]

[⋆]Dept. of Computational Linguistics, Heidelberg University. 69120 Heidelberg, Germany
jehl@cl.uni-heidelberg.de
[‡] SDL Research. East Road, Cambridge CB1 1BH, U.K.
{agispert,mhopkins,bbyrne}@sdl.com

Abstract

We present a simple preordering approach for machine translation based on a feature-rich logistic regression model to predict whether two children of the same node in the source-side parse tree should be swapped or not. Given the pair-wise children regression scores we conduct an efficient depth-first branch-and-bound search through the space of possible children permutations, avoiding using a cascade of classifiers or limiting the list of possible ordering outcomes. We report experiments in translating English to Japanese and Korean, demonstrating superior performance as (a) the number of crossing links drops by more than 10% absolute with respect to other state-of-the-art preordering approaches, (b) BLEU scores improve on 2.2 points over the baseline with lexicalised reordering model, and (c) decoding can be carried out 80 times faster.

1 Introduction

Source-side preordering for translation is the task of rearranging the order of a given source sentence so that it best resembles the order of the target sentence. It is a divide-and-conquer strategy aiming to decouple long-range word movement from the core translation task. The main advantage is that translation becomes computationally cheaper as less word movement needs to be considered, which results in faster and better translations, if preordering is done well and efficiently. Preordering also can facilitate better estimation of alignment and translation models as the parallel data becomes more monotonically-aligned, and translation gains can be obtained for various system architectures, e.g. phrase-based, hierarchical phrase-based, etc.

For these reasons, preordering has a clear research and commercial interest, as reflected by the extensive previous work on the subject (see Section 2). From these approaches, we are particularly interested in those that (i) involve little or no human intervention, (ii) require limited computational resources at runtime, and (iii) make use of available linguistic analysis tools.

In this paper we propose a novel preordering approach based on a logistic regression model trained to predict whether to swap nodes in the source-side dependency tree. For each pair of sibling nodes in the tree, the model uses a feature-rich representation that includes lexical cues to make relative reordering predictions between them. Given these predictions, we conduct a depth-first branch-and-bound search through the space of possible permutations of all sibling nodes, using the regression scores to guide the search. This approach has multiple advantages. First, the search for permutations is efficient and does not require specific heuristics or hard limits for nodes with many children. Second, the inclusion of the regression prediction directly into the search allows for finer-grained global decisions as the predictions that the model is more confident about are preferred. Finally, the use of a single regression model to handle any number of child nodes avoids incurring sparsity issues, while allowing the integration of a vast number of features into the preordering model.

We empirically contrast our proposed method against another preordering approach based on automatically-extracted rules when translating English into Japanese and Korean. We demonstrate a significant reduction in number of crossing links of more than 10% absolute, as well as translation gains of over 2.2 BLEU points over the baseline.

[*]This work was done during an internship of the first author at SDL Research, Cambridge.

We also show it outperforms a multi-class classification approach and analyse why this is the case.

2 Related work

One useful way to organize previous preordering techniques is by how they incorporate linguistic knowledge.

On one end of the spectrum we find those approaches that rely on syntactic parsers and human knowledge, typically encoded via a set of hand-crafted rules for parse tree rewriting or transformation. Examples of these can be found for French-English (Xia and McCord, 2004), German-English (Collins et al., 2005), Chinese-English (Wang et al., 2007), English-Arabic (Badr et al., 2009), English-Hindi (Ramanathan et al., 2009), English-Korean (Hong et al., 2009), and English-Japanese (Lee et al., 2010; Isozaki et al., 2010). A generic set of rules for transforming SVO to SOV languages has also been described (Xu et al., 2009). The main advantage of these approaches is that a relatively small set of good rules can yield significant improvements in translation. The common criticism they receive is that they are language-specific.

On the other end of the spectrum, there are preordering models that rely neither on human knowledge nor on syntactic analysis, but only on word alignments. One such approach is to form a cascade of two translation systems, where the first one translates the source to its preordered version (Costa-jussà and Fonollosa, 2006). Alternatively, one can define models that assign a cost to the relative position of each pair of words in the sentence, and search for the sequence that optimizes the global score as a linear ordering problem (Tromble and Eisner, 2009) or as a traveling salesman problem (Visweswariah et al., 2011). Yet another line of work attempts to automatically induce a parse tree and a preordering model from word alignments (DeNero and Uszkoreit, 2011; Neubig et al., 2012). These approaches are attractive due to their minimal reliance on linguistic knowledge. However, their findings reveal that the best performance is obtained when using human-aligned data which is expensive to create.

Somewhere in the middle of the spectrum are works that rely on automatic source-language syntactic parses, but no direct human intervention. Preordering rules can be automatically extracted from word alignments and constituent trees (Li et al., 2007; Habash, 2007; Visweswariah et al., 2010), dependency trees (Genzel, 2010) or predicate-argument structures (Wu et al., 2011), or simply part-of-speech sequences (Crego and Mariño, 2006; Rottmann and Vogel, 2007). Rules are assigned a cost based on Maximum Entropy (Li et al., 2007) or Maximum Likelihood estimation (Visweswariah et al., 2010), or directly on their ability to make the training corpus more monotonic (Genzel, 2010). The latter performs very well in practice but comes at the cost of a brute-force extraction heuristic that cannot incorporate lexical information. Recently, other approaches treat ordering the children of a node as a learning to rank (Yang et al., 2012) or discriminative multi-classification task (Lerner and Petrov, 2013). These are appealing for their use of finer-grained lexical information, but they struggle to adequately handle nodes with multiple children.

Our approach is closely related to this latter work, as we are interested in feature-rich discriminative approaches that automatically learn preordering rules from source-side dependency trees. Similarly to Yang et al. (2012) we train a large discriminative linear model, but rather than model each child's position in an ordered list of children, we model a more natural pair-wise swap / no-swap preference (like Tromble and Eisner (2009) did at the word level). We then incorporate this model into a global, efficient branch-and-bound search through the space of permutations. In this way, we avoid an error-prone cascade of classifiers or any limit on the possible ordering outcomes (Lerner and Petrov, 2013).

3 Preordering using logistic regression and branch-and-bound search

Like Genzel (2010), our method starts with dependency parses of source sentences (which we convert to shallow constituent trees; see Figure 1 for an example), and reorders the source text by permuting sibling nodes in the parse tree. For each non-terminal node, we first apply a logistic regression model which predicts, for each pair of child nodes, the probability that they should be swapped or kept in their original order. We then apply a depth-first branch-and-bound search to find the global optimal reordering of children.

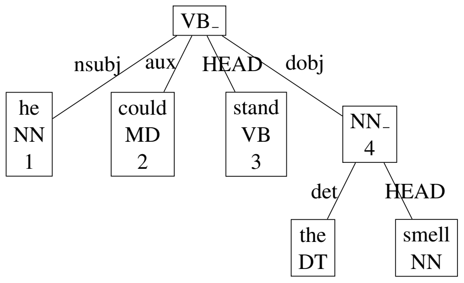


Figure 1: Shallow constituent tree generated from the dependency tree. Non-terminal nodes inherit the tag from the head.

3.1 Logistic regression

We build a regression model that assigns a probability of swapping any two sibling nodes, a and b, in the source-side dependency tree. The probability of swapping them is denoted $p(a, b)$ and the probability of keeping them in their original order is $1 - p(a, b)$. We use LIBLINEAR (Fan et al., 2008) for training an L1-regularised logistic regression model based on positively and negatively labelled samples.

3.1.1 Training data

We generate training examples for the logistic regression from word-aligned parallel data which is annotated with source-side dependency trees. For each non-terminal node, we extract all possible pairs of child nodes. For each pair, we obtain a binary label $y \in \{-1, 1\}$ by calculating whether swapping the two nodes would reduce the number of crossing alignment links. The crossing score of having two nodes a and b in the given order is

$$\mathrm{cs}(a, b) := |\{(i, j) \in A_a \times A_b : i > j\}|$$

where A_a and A_b are the target-side positions to which the words spanned by a and b are aligned. The label is then given as

$$y(a, b) = \begin{cases} 1 & , \quad \mathrm{cs}(a, b) > \mathrm{cs}(b, a) \\ -1 & , \quad \mathrm{cs}(b, a) > \mathrm{cs}(a, b) \end{cases}$$

Instances for which $\mathrm{cs}(a, b) = \mathrm{cs}(b, a)$ are not included in the training data. This usually happens if either A_a or A_b is empty, and in this case the alignments provide no indication of which order is better. We also discard any samples from nodes that have more than 16 children, as these are rare cases that often result from parsing errors.

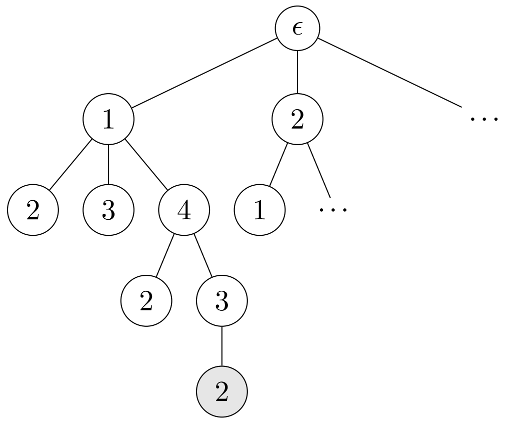


Figure 2: Branch-and-bound search: Partial search space of permutations for a dependency tree node with four children. The gray node marks a goal node. For the root node of the tree in Figure 1, the permutation corresponding to this path (1,4,3,2) would produce "he the smell stand could".

3.1.2 Features

Using a machine learning setup allows us to incorporate fine-grained information in the form of features. We use the following features to characterise pairs of nodes:

l	The dependency labels of each node
t	The part-of-speech tags of each node.
hw	The head words and classes of each node.
lm, rm	The left-most and right-most words and classes of a node.
dst	The distances between each node and the head.
gap	If there is a gap between nodes, the left-most and right-most words and classes in the gap.

In order to keep the size of our feature space manageable, we only consider features which occur at least 5 times[1]. For the lexical features, we use the top 100 vocabulary items from our training data, and 51 clusters generated by `mkcls` (Och, 1999). Similarly to previous work (Genzel, 2010; Yang et al., 2012), we also explore feature conjunctions. For the tag and label classes, we generate all possible combinations up to a given size. For the lexical and distance features, we explicitly specify conjunctions with the tag and label features. Results for various feature configurations are discussed in Section 4.3.1.

3.2 Search

For each non-terminal node in the source-side dependency tree, we search for the best possible

[1]Additional feature selection is achieved through L1-regularisation.

permutation of its children. We define the score of a permutation π as the product of the probabilities of its node pair orientations (swapped or unswapped):

$$\mathsf{score}(\pi) = \prod_{1 \le i < j \le k | \pi[i] > \pi[j]} p(i,j) \cdot \prod_{1 \le i < j \le k | \pi[i] < \pi[j]} 1 - p(i,j)$$

Here, we represent a permutation π of k nodes as a k-length sequence containing each integer in $\{1, ..., k\}$ exactly once. Define a *partial permutation* of k nodes as a $k' < k$ length sequence containing each integer in $\{1, ..., k\}$ at most once. We can construct a search space over partial permutations in the natural way (see Figure 2). The root node represents the empty sequence ϵ and has score 1. Then, given a search node representing a k'-length partial permutation π', its successor nodes are obtained by extending it by one element:

$$\mathsf{score}(\pi' \cdot \langle i \rangle) = \mathsf{score}(\pi') \cdot \prod_{j \in V | i > j} p(i,j) \cdot \prod_{j \in V | i < j} 1 - p(i,j)$$

where $V = \{1, ..., k\} \backslash (\pi' \cdot \langle i \rangle)$ is the set of source child positions that have not yet been visited. Observe that the nodes at search depth k correspond exactly to the set of complete permutations. To search this space, we employ depth-first branch-and-bound (Balas and Toth, 1983) as our search algorithm. The idea of branch-and-bound is to remember the best scoring goal node found thus far, abandoning any partial paths that cannot lead to a better scoring goal node. Algorithm 1 gives pseudocode for the algorithm[2]. If the initial bound ($bound_0$) is set to 0, the search is guaranteed to find the optimal solution. By raising the bound, which acts as an under-estimate of the best scoring permutation, search can be faster but possibly fail to find any solution. All our experiments were done with $bound_0 = 0$, i.e. exact search, but we discuss search time in detail and pruning alternatives in Section 4.3.2.

Since we use a logistic regression model and incorporate its predictions directly as swap probabilities, our search prefers those permutations with swaps which the model is more confident about.

[2]See (Poole and Mackworth, 2010) for more details and a worked example.

Algorithm 1 Depth-first branch-and-bound

```
Require: $k$: maximum sequence length, $\epsilon$: empty sequence,
  $bound_0$: initial bound

procedure BNBSEARCH($\epsilon$, $bound_0$, $k$)
    $best\_path \leftarrow \perp$
    $bound \leftarrow bound_0$
    SEARCH($\langle \epsilon \rangle$)
    return $best\_path$
end procedure

procedure SEARCH($\pi'$)
    if $score(\pi') > bound$ then
        if $|\pi'| = k$ then
            $best\_path \leftarrow \langle \pi' \rangle$
            $bound \leftarrow score(\pi')$
            return
        else
            for $each\ i \in \{1, ..., k\} \backslash \pi'$ do
                SEARCH($\pi' \cdot \langle i \rangle$)
            end for
        end if
    end if
end procedure
```

4 Experiments

4.1 Setup

We report translation results in English-to-Japanese/Korean. Our corpora are comprised of generic parallel data extracted from the web, with some documents extracted manually and some automatically crawled. Both have about 6M sentence pairs and roughly 100M words per language.

The dev and test sets are also generic. Source sentences were extracted from the web and one target reference was produced by a bilingual speaker. These sentences were chosen to evenly represent 10 domains, including world news, chat/SMS, health, sport, science, business, and others. The dev/test sets contain 602/903 sentences and 14K/20K words each. We do English part-of-speech tagging using SVMTool (Giménez and Màrquez, 2004) and dependency parsing using MaltParser (Nivre et al., 2007).

For translation experiments, we use a phrase-based decoder that incorporates a set of standard features and a hierarchical reordering model (Galley and Manning, 2008) with weights tuned using MERT to optimize the character-based BLEU score on the dev set. The Japanese and Korean language models are 5-grams estimated on $>$ 350M words of generic web text.

For training the logistic regression model, we automatically align the parallel training data and intersect the source-to-target and target-to-source alignments. We reserve a random 5K-sentence

approach	EJ cs (%)	EK cs (%)
rule-based (Genzel, 2010)	61.9	64.2
multi-class	65.2	-
df-bnb	51.4	51.8

Table 1: Percentage of the original crossing score on the heldout set, obtained after applying each preordering approach in English-Japanese (EJ, left) and Korean (EK, right). Lower is better.

subset for intrinsic evaluation of preordering, and use the remainder for model parameter estimation.

We evaluate our preordering approach with logistic regression and depth-first branch-and-bound search (in short, 'df-bnb') both in terms of reordering via crossing score reduction on the heldout set, and in terms of translation quality as measured by character-based BLEU on the test set.

4.2 Preordering baselines

We contrast our work against two data-driven preordering approaches. First, we implemented the rule-based approach of Genzel (2010) and optimised its multiple parameters for our task. We report only the best results achieved, which correspond to using ∼100K training sentences for rule extraction, applying a sliding window width of 3 children, and creating rule sequences of ∼60 rules. This approach cannot incorporate lexical features as that would make the brute-force rule extraction algorithm unmanageable.

We also implemented a multi-class classification setup where we directly predict complete permutations of children nodes using multi-class classification (Lerner and Petrov, 2013). While this is straightforward for small numbers of children, it leads to a very large number of possible permutations for larger sets of children nodes, making classification too difficult. While Lerner and Petrov (2013) use a cascade of classifiers and impose a hard limit on the possible reordering outcomes to solve this, we follow Genzel's heuristic: rather than looking at the complete set of children, we apply a sliding window of size 3 starting from the left, and make classification/reordering decisions for each window separately. Since the windows overlap, decisions made for the first window affect the order of nodes in the second window, etc. We address this by soliciting decisions from the classifier on the fly as we preorder. One limitation of this approach is that it is able to move children only within the window. We try to remedy this by applying the method iteratively, each time re-training the classifier on the preordered data from the previous run.

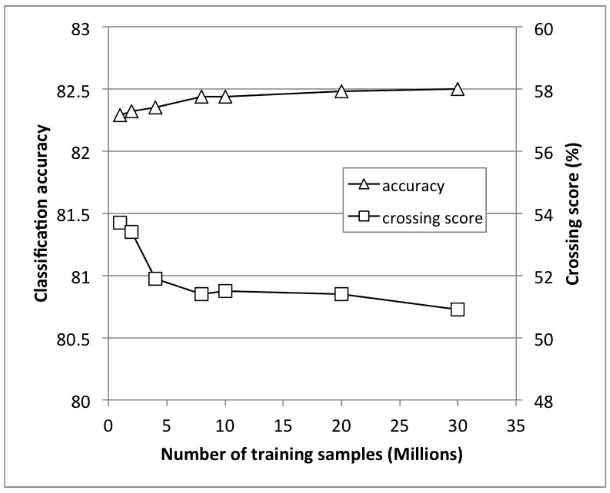


Figure 3: Crossing scores and classification accuracy improve with training data size.

4.3 Crossing score

We now report contrastive results in the intrinsic preordering task, as measured by the number of crossing links (Genzel, 2010; Yang et al., 2012) on the 5K held-out set. Without preordering, there is an average of 22.2 crossing links in English-Japanese and 20.2 in English-Korean. Table 1 shows what percentage of these links remain after applying each preordering approach to the data. We find that the 'df-bnb' method outperforms the other approaches in both language pairs, achieving more than 10 additional percentage points reduction over the rule-based approach. Interestingly, the multi-class approach is not able to match the rule-based approach despite using additional lexical cues. We hypothesise that this is due to the sliding window heuristic, which causes a mismatch in train-test conditions: while samples are not independent of each other at test time due to window overlaps, they are considered to be so when training the classifier.

4.3.1 Impact of training size and feature configuration

We now report the effects of feature configuration and training data size for the English-Japanese case. We assess our 'df-bnb' approach in terms of the classification accuracy of the trained logistic

features used	acc (%)	cs (%)
l,t,hw,lm,rm,dst,gap	82.43	51.3
l,t,hw,lm,rm,dst	82.44	51.4
l,t,hw,lm,rm	82.32	53.1
l,t,hw	82.02	55
l,t	81.07	58.4

Table 2: Ablation tests showing crossing scores and classification accuracy as features are removed. All models were trained on 8M samples.

regression model (using it to predict ±1 labels in the held-out set) and by the percentage of crossing alignment links reduced by preordering.

Figure 3 shows the performance of the logistic regression model over different training set sizes, extracted from the training corpus as described in Section 3. We observe a constant increase in prediction accuracy, mirrored by a steady decrease in crossing score. However, gains are less for more than 8M training examples. Note that a small variation in accuracy can produce a large variation in crossing score if two nodes are swapped which have a large number of crossing alignments.

Table 2 shows an ablation test for various feature configurations. We start with all features, including head word and class (*hw*), left-most and right-most word in each node's span (*lm*, *rm*), each node's distance to the head (*dst*), and left-most and right-most word of the gap between nodes (*gap*). We then proceed by removing features to end with only label and tag features (*l,t*), as in Genzel (2010). For each configuration, we generated all tag- and label- combinations of size 2. We then specified combinations between tag and label and all other features. For the lexical features we always used conjunctions of the word itself, and its class. Class information is included for all words, not just those in the top 100 vocabulary. Table 2 shows that lexical and distance feature groups contribute to prediction accuracy and crossing score, except for the *gap* features, which we omit from further experiments.

4.3.2 Run time

We now demonstrate the efficiency of branch-and-bound search for the problem of finding the optimum permutation of n children at runtime. Even though in the worst case the search could explore all $n!$ permutations, making it prohibitive for

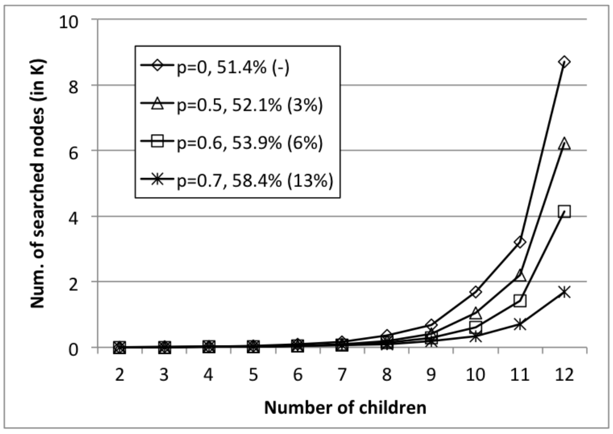


Figure 4: Average number of nodes explored in branch-and-bound search by number of children.

nodes with many children, in practice this does not happen. Many low-scoring paths are discarded early by branch-and-bound search so that the optimal solution can be found quickly. The top curve in Figure 4 shows the average number of nodes explored in searches run on our validation set (5K sentences) as a function of the number of children. All instances are far from the worst case[3].

In our experiments, the time needed to conduct exact search ($bound_0 = 0$) was not a problem except for a few bad cases (nodes with more than 16 children), which we simply chose not to preorder; in our data, 90% of the nodes have less than 6 children, while only 0.9% have 10 children or more, so this omission does not affect performance noticeably. We verified this on our held-out set, by carrying out exhaustive searches. We found that not preordering nodes with 16 children did not worsen the crossing score. In fact, setting a harsher limit of 10 nodes would still produce a crossing score of 51.9%, compared to the best score of 51.4%.

There are various ways to speed up the search, if needed. First, one could impose a hard limit on the number of explored nodes[4]. As shown in Figure 4, a limit of 4K would still allow exact search on average for permutations of up to 11 children, while stopping search early for more children. We tested this for limits of 1K/4K nodes and obtained crossing scores of 51.9/51.5%. Alternatively, one could define a higher initial bound; since the score of a path is a product of probabilities, one would select a threshold probability

[3]Note that 12!$\approx$479M nodes, whereas our search finds the optimal permutation path after exploring $<$10K nodes.

[4]As long as the limit exceeds the permutation length, a solution will always be found as search is depth-first.

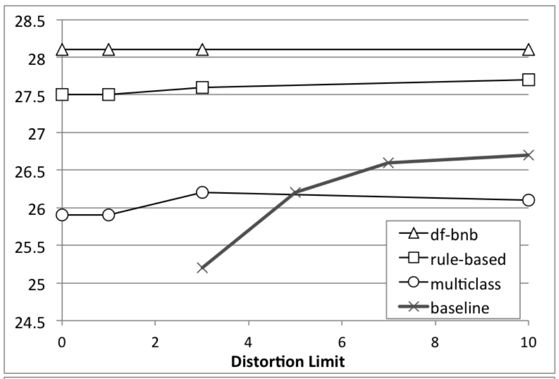
d	approach	−LRM	Δ	+LRM	Δ
10	baseline	25.39	-	26.62	-
	rule-based	25.93	**+0.54**	27.65	**+1.03**
	multi-class	25.60	**+0.21**	26.10	**−0.52**
	df-bnb	26.73	**+1.34**	28.09	**+1.47**
4	baseline	25.07	-	25.92	-
	rule-based	26.35	**+1.28**	27.54	**+1.62**
	multi-class	25.37	**+0.30**	26.31	**+0.39**
	df-bnb	26.98	**+1.91**	28.13	**+2.21**

Table 3: English-Japanese BLEU scores with various preordering approaches (and improvement over baseline) under two distortion limits d. Results reported both excluding and including lexicalised reordering model features (LRM).

p and calculate a bound depending on the size n of the permutation as $bound_0 = p^{\frac{n \cdot (n-1)}{2}}$. Examples of this would be the lower curves of Figure 4. The curve labels show the crossing score produced with each threshold, and in parenthesis the percentage of searches that fail to find a solution with a better score than $bound_0$, in which case children are left in their original order. As shown, this strategy proves less effective than simply limiting the number of explored nodes, because the more frequent cases with less children remain unaffected.

4.4 Translation performance

Table 3 reports English-Japanese translation results for two different values of the distortion limit d, i.e. the maximum number of source words that the decoder is allowed to jump during search. We draw the following conclusions. Firstly, all the preordering approaches outperform the baseline and the BLEU score gain they provide increases as the distortion limit decreases. This is further analysed in Figure 5, where we report BLEU as a function of the distortion limit in decoding for both English-Japanese and English-Korean. This reveals the power of preordering as a targeted strategy to obtain high performance at fast decoding times, since d can be drastically reduced without performance degradation which leads to huge decoding speed-ups; this is consistent with the observations in (Xu et al., 2009; Genzel, 2010; Visweswariah et al., 2011). We also find that with preordering it is possible to apply harsher pruning conditions in decoding while still maintaining the exact same performance, achieving further speed-ups. With preordering, our system is able to decode 80 times faster while producing translation output of the same quality.



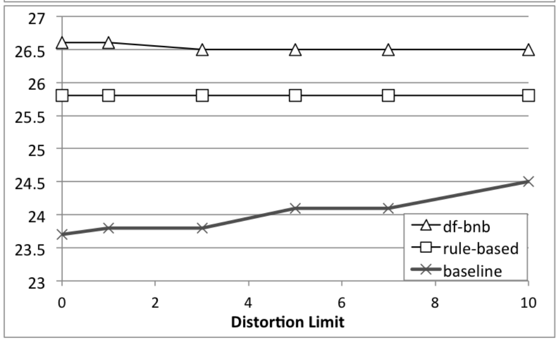


Figure 5: BLEU scores as a function of distortion limit in decoder (+LRM case). Top: English-Japanese. Bottom: English-Korean.

Secondly, we observe that the preordering gains, which are correlated with the crossing score reductions of Table 1, are largely orthogonal to the gains obtained when incorporating a lexicalised reordering model (LRM). In fact, preordering gains are slightly larger with LRM, suggesting that this reordering model can be better estimated with preordered text. This echoes the notion that reordering models are particularly sensitive to alignment noise (DeNero and Uszkoreit, 2011; Neubig et al., 2012; Visweswariah et al., 2013), and that a 'more monotonic' training corpus leads to better translation models.

Finally, 'df-bnb' outperforms all other preordering approaches, and achieves an extra 0.5–0.8 BLEU over the rule-based one even at zero distortion limit. This is consistent with the substantial crossing score reductions reported in Section 4.3.

We argue that these improvements are due to the usage of lexical features to facilitate finer-grained ordering decisions, and to our better search through the children permutation space which is not restricted by sliding windows, does

Example 1	reference	[$_1$バーローは]$_{\text{Barlow}}$ [$_2$悪臭に]$_{\text{the smell}}$ [$_3$我慢]$_{\text{endure}}$ [$_4$できることを]$_{\text{could}}$ [$_5$願った]$_{\text{hoped}}$ [$_6$。]
	source	[$_1$Barlow] [$_5$hoped] he [$_4$could] [$_3$stand] [$_2$the smell] [$_6$.]
	preordered	[$_1$Barlow] he [$_2$the smell] [$_3$stand] [$_4$could] [$_5$hoped] [$_6$.]
Example 2	reference	[$_1$私自身の]$_{\text{my own}}$ [$_2$経験]$_{\text{experience}}$ [$_3$において]$_{\text{in}}$, [$_4$ローザパル クス]$_{\text{Rosa Parks}}$ [$_5$という]$_{\text{called}}$ [$_6$黒人の]$_{\text{black}}$ [$_7$女性は]$_{\text{woman}}$，[$_8$ある日]$_{\text{one day}}$ [$_9$とにかくとにかく]$_{\text{somehow}}$ [$_{10}$バスの]$_{\text{bus of}}$ [$_{11}$後部座席に]$_{\text{back seat in}}$ [$_{12}$坐る]$_{\text{sit}}$ ように [$_{13}$言われる]$_{\text{told being}}$ [$_{14}$ことに]$_{\text{of}}$ [$_{15}$うんざりす]$_{\text{was fed up with}}$ 。
	source	[$_3$In] [$_1$my own] [$_2$experience] , a [$_6$black] [$_7$woman] [$_5$named] [$_4$Rosa Parks] [$_{14}$was just tired] [$_8$one day] [$_{14}$of] [$_{13}$being told] [$_{12}$to sit] [$_{11}$in the back] [$_{10}$of the bus] .
	rule-based	[$_1$my own] [$_2$experience] [$_3$In] [$_{14}$was just tired] [$_{13}$being told] [$_{10}$the bus of] [$_{11}$the back in] [$_{12}$sit to] [$_{14}$of] [$_8$one day] , [$_6$a black] [$_7$woman] [$_4$Rosa Parks] [$_5$named] .
	df-bnb	[$_1$my own] [$_2$experience] [$_3$In] , [$_5$named] [$_6$a black] [$_7$woman] [$_4$Rosa Parks] [$_{10}$the bus of] [$_{11}$the back in] [$_{12}$sit to] [$_{13}$told being] [$_{14}$of] [$_8$one day] [$_{14}$was just tired] .
Example 3	reference	[$_1$私たちは]$_{\text{we}}$、[$_2$すっかり]$_{\text{quite}}$ [$_3$西安が]$_{\text{Xi'an}}$ [$_4$好き]$_{\text{like}}$ [$_5$に]$_{\text{to}}$ [$_6$なりました]$_{\text{come have}}$ 。
	source	[$_1$we] [$_6$have come] [$_5$to] [$_2$quite] [$_4$like] [$_1$xi'an] .
	rule-based	[$_1$we] [$_2$quite] [$_4$like] [$_3$xi'an] [$_5$to] [$_6$come have] .
	df-bnb	[$_1$we] have [$_2$quite] [$_3$xi'an] [$_4$like] [$_5$to] [$_6$come] .
	baseline	私たちはをかなり西安と同様です 。
	rule-based	私たちはかなりのように西安に来ます。
	df-bnb	私たちはかなり西安が好きになる。

Table 4: Examples from our test data illustrating the differences between the preordering approaches.

not depend heavily on getting the right decision in a multi-class scenario, and which incorporates regression to carry out a score-driven search.

4.5 Analysis

Table 4 gives three English-Japanese examples to illustrate the different preordering approaches. The first, very short, example is preordered correctly by the rule-based and the df-bnb approach, as the order of the brackets matches the order of the Japanese reference.

For longer sentences we see more differences between approaches, as illustrated by Example 2. In this case, both approaches succeed at moving prepositions to the back of the phrase ("my experience in", "the bus of"). However, while the df-bnb approach correctly moves the predicate of the second clause ("was just tired") to the back, the rule-based approach incorrectly moves the subject ("a black woman named Rosa Parks") to this position - possibly because of the verb "named" which occurs in the phrase. This could be an indication that the df-bnb is better suited for more complicated constructions. With the exception of phrases 4 and 8, all other phrases are in the correct order in the df-bnb reordering. None of the approaches manage to reorder "a black woman named Rosa Parks" to the correct order.

Example 3 shows that the translations into Japanese also reflect preordering quality. The original source results in "like" being translated as the main verb (which is incorrectly interpreted as "to be like, to be equal to"). The rule-based version correctly moves "have come" to the end, but fails to swap "xi'an" and "like", resulting in "come" being interpreted as a full verb, rather than an auxiliary. Only the df-bnb version achieves almost perfect reordering, resulting in the correct word choice of なる (to get to, to become) for "have come to".[5]

5 Conclusion

We have presented a novel preordering approach that estimates a preference for swapping or not swapping pairs of children nodes in the source-side dependency tree by training a feature-rich logistic regression model. Given the pair-wise scores, we efficiently search through the space of possible children permutations using depth-first branch-and-bound search. The approach is able to incorporate large numbers of features including lexical cues, is efficient at runtime even with a large number of children, and proves superior to other state-of-the-art preordering approaches both in terms of crossing score and translation performance.

[5]This translation is still not perfect, since it uses the wrong level of politeness, an important distinction in Japanese.

References

Ibrahim Badr, Rabih Zbib, and James Glass. 2009. Syntactic Phrase Reordering for English-to-Arabic Statistical Machine Translation. In *Proceedings of EACL*, pages 86–93, Athens, Greece.

Egon Balas and Paolo Toth. 1983. *Branch and Bound Methods for the Traveling Salesman Problem*. Carnegie-Mellon Univ. Pittsburgh PA Management Sciences Research Group.

Michael Collins, Philipp Koehn, and Ivona Kucerova. 2005. Clause Restructuring for Statistical Machine Translation. In *Proceedings of ACL*, pages 531–540, Ann Arbor, Michigan.

Marta R. Costa-jussà and José A. R. Fonollosa. 2006. Statistical Machine Reordering. In *Proceedings of EMNLP*, pages 70–76, Sydney, Australia.

Josep M. Crego and José B. Mariño. 2006. Integration of POStag-based Source Reordering into SMT Decoding by an Extended Search Graph. In *Proceedings of AMTA*, pages 29–36, Cambridge, Massachusetts.

John DeNero and Jakob Uszkoreit. 2011. Inducing Sentence Structure from Parallel Corpora for Reordering. In *Proceedings of EMNLP*, pages 193–203, Edinburgh, Scotland, UK.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A Library for Large Linear Classification. *Journal of Machine Learning Research*, 9:1871–1874.

Michel Galley and Christopher D. Manning. 2008. A Simple and Effective Hierarchical Phrase Reordering Model. In *Proceedings of EMNLP*, pages 847–855, Honolulu, Hawaii.

Dmitriy Genzel. 2010. Automatically learning source-side reordering rules for large scale machine translation. In *Proceedings of COLING*, pages 376–384, Beijing, China.

Jesús Giménez and Lluís Màrquez. 2004. SVMTool: A general POS tagger generator based on Support Vector Machines. In *Proceedings of LREC*, Lisbon, Portugal.

Nizar Habash. 2007. Syntactic Preprocessing for Statistical Machine Translation. In *Proceedings of MT-Summit*, pages 215–222, Copenhagen, Denmark.

Gumwon Hong, Seung-Wook Lee, and Hae-Chang Rim. 2009. Bridging Morpho-Syntactic Gap between Source and Target Sentences for English-Korean Statistical Machine Translation. In *Proceedings of ACL-IJCNLP*, pages 233–236, Suntec, Singapore.

Hideki Isozaki, Katsuhito Sudoh, Hajime Tsukada, and Kevin Duh. 2010. Head Finalization: A Simple Reordering Rule for SOV Languages. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 244–251, Uppsala, Sweden.

Young-Suk Lee, Bing Zhao, and Xiaoqian Luo. 2010. Constituent Reordering and Syntax Models for English-to-Japanese Statistical Machine Translation. In *Proceedings of COLING*, pages 626–634, Beijing, China.

Uri Lerner and Slav Petrov. 2013. Source-Side Classifier Preordering for Machine Translation. In *Proceedings of EMNLP*, Seattle, USA.

Chi-Ho Li, Minghui Li, Dongdong Zhang, Mu Li, Ming Zhou, and Yi Guan. 2007. A Probabilistic Approach to Syntax-based Reordering for Statistical Machine Translation. In *Proceedings of ACL*, pages 720–727, Prague, Czech Republic.

Graham Neubig, Taro Watanabe, and Shinsuke Mori. 2012. Inducing a Discriminative Parser to Optimize Machine Translation Reordering. In *Proceedings of EMNLP-CoNLL*, pages 843–853, Jeju Island, Korea.

Joakim Nivre, Johan Hall, Jens Nilsson, Atanas Chanev, Gülsen Eryigit, Sandra Kübler, Svetoslav Marinov, and Erwin Marsi. 2007. Maltparser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(2):95–135.

Franz Josef Och. 1999. An efficient method for determining bilingual word classes. In *Proceedings of EACL*, pages 71–76, Bergen, Norway.

David L. Poole and Alan K. Mackworth. 2010. *Artificial Intelligence: Foundations of Computational Agents*. Cambridge University Press. Full text online at http://artint.info.

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*, pages 800–808, Suntec, Singapore.

Kay Rottmann and Stephan Vogel. 2007. Word Reordering in Statistical Machine Translation with a POS-Based Distortion Model. In *Proceedings of TMI*, pages 171–180, Skövde, Sweden.

Roy Tromble and Jason Eisner. 2009. Learning linear ordering problems for better translation. In *Proceedings of EMNLP*, pages 1007–1016, Singapore.

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 COLING*, pages 1119–1127, Beijing, China.

Karthik Visweswariah, Rajakrishnan Rajkumar, Ankur Gandhe, Ananthakrishnan Ramanathan, and Jiri Navratil. 2011. A word reordering model for improved machine translation. In *Proceedings of EMNLP*, pages 486–496, Edinburgh, United Kingdom.

Karthik Visweswariah, Mitesh M. Khapra, and Ananthakrishnan Ramanathan. 2013. Cut the noise: Mutually reinforcing reordering and alignments for improved machine translation. In *Proceedings of ACL*, pages 1275–1284, Sofia, Bulgaria.

Chao Wang, Michael Collins, and Philipp Koehn. 2007. Chinese Syntactic Reordering for Statistical Machine Translation. In *Proceedings of EMNLP-CoNLL*, pages 737–745, Prague, Czech Republic.

Xianchao Wu, Katsuhito Sudoh, Kevin Duh, Hajime Tsukada, and Masaaki Nagata. 2011. Extracting Pre-ordering Rules from Predicate-Argument Structures. In *Proceedings of IJCNLP*, pages 29–37, Chiang Mai, Thailand.

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

Peng Xu, Jaeho Kang, Michael Ringgaard, and Franz Och. 2009. Using a Dependency Parser to Improve SMT for Subject-Object-Verb Languages. In *Proceedings of HTL-NAACL*, pages 245–253, Boulder, Colorado.

Nan Yang, Mu Li, Dongdong Zhang, and Nenghai Yu. 2012. A ranking-based approach to word reordering for statistical machine translation. In *Proceedings of ACL*, pages 912–920, Jeju Island, Korea.