A Joint Sequence Translation Model with Integrated Reordering

Nadir Durrani **Helmut Schmid** **Alexander Fraser**
Institute for Natural Language Processing
University of Stuttgart
{durrani,schmid,fraser}@ims.uni-stuttgart.de

Abstract

We present a novel machine translation model which models translation by a linear sequence of *operations*. In contrast to the "N-gram" model, this sequence includes not only translation but also reordering operations. Key ideas of our model are (i) a new reordering approach which better restricts the position to which a word or phrase can be moved, and is able to handle short and long distance reorderings in a unified way, and (ii) a joint sequence model for the translation and reordering probabilities which is more flexible than standard phrase-based MT. We observe statistically significant improvements in BLEU over Moses for German-to-English and Spanish-to-English tasks, and comparable results for a French-to-English task.

1 Introduction

We present a novel generative model that explains the translation process as a linear sequence of operations which generate a source and target sentence in parallel. Possible operations are (i) generation of a sequence of source and target words (ii) insertion of *gaps* as explicit target positions for reordering operations, and (iii) forward and backward jump operations which do the actual reordering. The probability of a sequence of operations is defined according to an N-gram model, i.e., the probability of an operation depends on the $n-1$ preceding operations. Since the translation (generation) and reordering operations are coupled in a single generative story, the reordering decisions may depend on preceding translation decisions and translation decisions may depend on preceding reordering decisions. This provides a natural reordering mechanism which is able to deal with local and long-distance reorderings in a consistent way. Our approach can be viewed as an extension of the N-gram SMT approach (Mariño et al., 2006) but our model does reordering as an integral part of a generative model.

The paper is organized as follows. Section 2 discusses the relation of our work to phrase-based and the N-gram SMT. Section 3 describes our generative story. Section 4 defines the probability model, which is first presented as a generative model, and then shifted to a discriminative framework. Section 5 provides details on the search strategy. Section 6 explains the training process. Section 7 describes the experimental setup and results. Section 8 gives a few examples illustrating different aspects of our model and Section 9 concludes the paper.

2 Motivation and Previous Work

2.1 Relation of our work to PBSMT

Phrase-based SMT provides a powerful translation mechanism which learns local reorderings, translation of short idioms, and the insertion and deletion of words sensitive to local context. However, PBSMT also has some drawbacks. (i) Dependencies across phrases are not directly represented in the translation model. (ii) Discontinuous phrases cannot be used. (iii) The presence of many different equivalent segmentations increases the search space.

Phrase-based SMT models dependencies between words and their translations inside of a phrase well. However, dependencies across phrase boundaries are largely ignored due to the strong phrasal inde-

German	English
hat er ein buch gelesen	he read a book
hat eine pizza gegessen	has eaten a pizza
er	he
hat	has
ein	a
eine	a
menge	lot of
butterkekse	butter cookies
gegessen	eaten
buch	book
zeitung	newspaper
dann	then

Table 1: Sample Phrase Table

pendence assumption. A phrase-based system using the phrase table[1] shown in Table 1, for example, correctly translates the German sentence "er hat eine pizza gegessen" to "he has eaten a pizza", but fails while translating "er hat eine menge butterkekse gegessen" (see Table 1 for a gloss) which is translated as "he has a lot of butter cookies eaten" unless the language model provides strong enough evidence for a different ordering. The generation of this sentence in our model starts with generating "er – he", "hat – has". Then a gap is inserted on the German side, followed by the generation of "gegessen – eaten". At this point, the (partial) German and English sentences look as follows:

er hat ☐ gegessen
he has eaten

We jump back to the gap on the German side and fill it by generating "eine – a" and "pizza – pizza", for the first example and generating "eine – a", "menge – lot of", "butterkekse – butter cookies" for the second example, thus handling both short and long distance reordering in a unified manner. Learning the pattern "hat ☐ gegessen – has eaten" helps us to generalize to the second example with unseen context. Notice how the reordering decision is triggered by the translation decision in our model. The probability of a gap insertion operation after the generation of the auxiliaries "hat – has" will be high because reordering is necessary in order to move the second part of the German verb complex ("gegessen") to its correct position at the end of the clause. This mechanism better restricts reordering

[1]The examples given in this section are not taken from the real data/system, but made-up for the sake of argument.

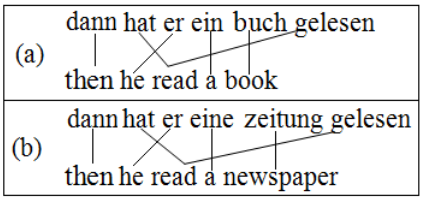

Figure 1: (a) Known Context (b) Unknown Context

than traditional PBSMT and is able to deal with local and long-distance reorderings in a consistent way.

Another weakness of the traditional phrase-based system is that it can only capitalize on continuous phrases. Given the phrase inventory in Table 1, phrasal MT is able to generate example in Figure 1(a). The information "hat...gelesen – read" is internal to the phrase pair "hat er ein buch gelesen – he read a book", and is therefore handled conveniently. On the other hand, the phrase table does not have the entry "hat er eine zeitung gelesen – he read a newspaper" (Figure 1(b)). Hence, there is no option but to translate "hat...gelesen" separately, translating "hat" to "has" which is a common translation for "hat" but wrong in the given context. Context-free hierarchical models (Chiang, 2007; Melamed, 2004) have rules like "hat er X gelesen – he read X" to handle such cases. Galley and Manning (2010) recently solved this problem for phrasal MT by extracting phrase pairs with source and target-side gaps. Our model can also use tuples with source-side discontinuities. The above sentence would be generated by the following sequence of operations: (i) generate "dann – then" (ii) insert a gap (iii) generate "er – he" (iv) backward jump to the gap (v) generate "hat...[gelesen] – read" (only "hat" and "read" are added to the sentences yet) (vi) jump forward to the right-most source word so far generated (vii) insert a gap (viii) continue the source cept ("gelesen" is inserted now) (ix) backward jump to the gap (x) generate "ein – a" (xi) generate "buch – book".

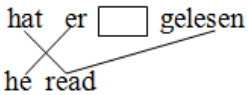

Figure 2: Pattern

From this operation sequence, the model learns a pattern (Figure 2) which allows it to generalize to the example in Figure 1(b). The open gap represented by ☐ serves a similar purpose as the non-terminal categories in a hierarchical phrase-based system such as Hiero. Thus it generalizes to translate "eine zeitung" in exactly the same way as "ein buch".

Another problem of phrasal MT is spurious phrasal segmentation. Given a sentence pair and a corresponding word alignment, phrasal MT can learn an arbitrary number of source segmentations. This is problematic during decoding because different compositions of the same minimal phrasal units are allowed to compete with each other.

2.2 Relation of our work to N-gram SMT

N-gram based SMT is an alternative to hierarchical and non-hierarchical phrase-based systems. The main difference between phrase-based and N-gram SMT is the extraction procedure of translation units and the statistical modeling of translation context (Crego et al., 2005a). The tuples used in N-gram systems are much smaller translation units than phrases and are extracted in such a way that a unique segmentation of each bilingual sentence pair is produced. This helps N-gram systems to avoid the spurious phrasal segmentation problem. Reordering works by linearization of the source side and tuple unfolding (Crego et al., 2005b). The decoder uses word lattices which are built with linguistically motivated re-write rules. This mechanism is further enhanced with an N-gram model of bilingual units built using POS tags (Crego and Yvon, 2010). A drawback of their reordering approach is that search is only performed on a small number of reorderings that are pre-calculated on the source side independently of the target side. Often, the evidence for the correct ordering is provided by the target-side language model (LM). In the N-gram approach, the LM only plays a role in selecting between the pre-calculated orderings.

Our model is based on the N-gram SMT model, but differs from previous N-gram systems in some important aspects. It uses operation n-grams rather than tuple n-grams. The reordering approach is entirely different and considers all possible orderings instead of a small set of pre-calculated orderings. The standard N-gram model heavily relies on POS tags for reordering and is unable to use lexical triggers whereas our model exclusively uses lexical triggers and no POS information. Linearization and unfolding of the source sentence according to the target sentence enables N-gram systems to handle source-side gaps. We deal with this phenomenon more directly by means of tuples with source-side discontinuities. The most notable feature of our work is that it has a complete generative story of translation which combines translation and reordering operations into a single operation sequence model.

Like the N-gram model[2], our model cannot deal with target-side discontinuities. These are eliminated from the training data by a post-editing process on the alignments (see Section 6). Galley and Manning (2010) found that target-side gaps were not useful in their system and not useful in the hierarchical phrase-based system Joshua (Li et al., 2009).

3 Generative Story

Our generative story is motivated by the complex reorderings in the German-to-English translation task. The German and English sentences are jointly generated through a sequence of operations. The English words are generated in linear order[3] while the German words are generated in parallel with their English translations. Occasionally the translator jumps back on the German side to insert some material at an earlier position. After this is done, it jumps forward again and continues the translation. The backward jumps always end at designated landing sites (gaps) which were explicitly inserted before. We use 4 translation and 3 reordering operations. Each is briefly discussed below.

Generate (X,Y): X and Y are German and English cepts[4] respectively, each with one or more words. Words in X (German) may be consecutive or discontinuous, but the words in Y (English) must be consecutive. This operation causes the words in Y and the first word in X to be added to the English and German strings respectively, that were generated so far. Subsequent words in X are added to a queue to be generated later. All the English words in Y are generated immediately because English is generated in linear order. The generation of the second (and subsequent) German word in a multi-word cept can be delayed by gaps, jumps and the Generate Source Only operation defined below.

Continue Source Cept: The German words added

[2]However, Crego and Yvon (2009), in their N-gram system, use split rules to handle target-side gaps and show a slight improvement on a Chinese-English translation task.

[3]Generating the English words in order is also what the decoder does when translating from German to English.

[4]A cept is a group of words in one language translated as a minimal unit in one specific context (Brown et al., 1993).

to the queue by the Generate (X,Y) operation are generated by the Continue Source Cept operation. Each Continue Source Cept operation removes one German word from the queue and copies it to the German string. If X contains more than one German word, say n many, then it requires n translation operations, an initial Generate $(X_1...X_n, Y)$ operation and $n-1$ Continue Source Cept operations. For example "hat...gelesen – read" is generated by the operation Generate (hat gelesen, read), which adds "hat" and "read" to the German and English strings and "gelesen" to a queue. A Continue Source Cept operation later removes "gelesen" from the queue and adds it to the German string.

Generate Source Only (X): The string X is added at the current position in the German string. This operation is used to generate a German word X with no corresponding English word. It is performed immediately after its preceding German word is covered. This is because there is no evidence on the English-side which indicates when to generate X. Generate Source Only (X) helps us learn a source word deletion model. It is used during decoding, where a German word (X) is either translated to some English word(s) by a Generate (X,Y) operation or deleted with a Generate Source Only (X) operation.

Generate Identical: The same word is added at the current position in both the German and English strings. The Generate Identical operation is used during decoding for the translation of unknown words. The probability of this operation is estimated from singleton German words that are translated to an identical string. For example, for a tuple "Portland – Portland", where German "Portland" was observed exactly once during training, we use a Generate Identical operation rather than Generate (Portland, Portland).

We now discuss the set of reordering operations used by the generative story. Reordering has to be performed whenever the German word to be generated next does not immediately follow the previously generated German word. During the generation process, the translator maintains an index which specifies the position after the previously covered German word (j), an index (Z) which specifies the index after the right-most German word covered so far, and an index of the next German word to be covered (j'). The set of reordering operations used in generation depends upon these indexes.

Operations	Generation	States
Generate (dann , then)	dann ↓ then	i=1 j=1 j'=2 k=0 Z=1
Insert Gap – Generate (er , he)	dann □ er ↓ then he	i=2 j=3 j'=1 k=0 Z=3 S=1
Jump Back(1) – Generate (hat gelesen, read)	dann hat ↓ er then he read	i=3 j=2 j'=5 k=1 Z=3
Jump Forward – Insert Gap – Continue Source Cept	dann hat er □ gelesen ↓ then he read	i=3 j=6 j'=3 k=0 Z=6 S=3
Jump Back(1) – Generate (ein , a)	dann hat er ein ↓ gelesen then he read a	i=4 j =4 j'=4 k=0 Z=6
Generate (buch , book)	dann hat er ein buch ↓ gelesen then he read a book	i=5 j=5 j'=6 k=0 Z=6

Table 2: Step-wise Generation of Example 1 (a). The arrow indicates position j.

Insert Gap: This operation inserts a gap which acts as a place-holder for the skipped words. There can be more than one open gap at a time.

Jump Back (W): This operation lets the translator jump back to an open gap. It takes a parameter W specifying which gap to jump to. Jump Back (1) jumps to the closest gap to Z, Jump Back (2) jumps to the second closest gap to Z, etc. After the backward jump the target gap is closed.

Jump Forward: This operation makes the translator jump to Z. It is performed if some already generated German word is between the previously generated word and the word to be generated next. A Jump Back (W) operation is only allowed at position Z. Therefore, if $j \neq Z$, a Jump Forward operation has to be performed prior to a Jump Back operation.

Table 2 shows step by step the generation of a German/English sentence pair, the corresponding translation operations, and the respective values of the index variables. A formal algorithm for converting a word-aligned bilingual corpus into an operation sequence is presented in Algorithm 1.

4 Model

Our translation model $p(F, E)$ is based on operation N-gram model which integrates translation and reordering operations. Given a source string F, a sequence of tuples $T = (t_1, \ldots, t_n)$ as hypothesized by the decoder to generate a target string E, the translation model estimates the probability of a

Algorithm 1 Corpus Conversion Algorithm

i	Position of current English cept
j	Position of current German word
j'	Position of next German word
N	Total number of English cepts
f_j	German word at position j
E_i	English cept at position i
F_i	Sequence of German words linked to E_i
L_i	Number of German words linked with E_i
k	Number of already generated German words for E_i
a_{ik}	Position of k^{th} German translation of E_i
Z	Position after right-most generated German word
S	Position of the first word of a target gap

$i := 0; j := 0; k := 0$

while f_j is an unaligned word **do**
  Generate Source Only (f_j)
  $j := j + 1$
$Z := j$

while $i < N$ **do**
  $j' := a_{ik}$
  if $j < j'$ **then**
    if f_j was not generated yet **then**
      Insert Gap
    if $j = Z$ **then**
      $j := j'$
    else
      Jump Forward
  if $j' < j$ **then**
    if $j < Z$ and f_j was not generated yet **then**
      Insert Gap
    $W :=$ relative position of target gap
    Jump Back (W)
    $j := S$
  if $j < j'$ **then**
    Insert Gap
    $j := j'$
  if k = 0 **then**
    Generate (F_i, E_i) {or **Generate Identical**}
  else
    Continue Source Cept
  $j := j + 1; k := k + 1$
  while f_j is an unaligned word **do**
    Generate Source Only (f_j)
    $j := j + 1$
  if $Z < j$ **then**
    $Z := j$
  if $k = L_i$ **then**
    $i := i + 1; k := 0$

Remarks:
We use cept positions for English (not word positions) because English cepts are composed of consecutive words. German positions are word-based.
The relative position of the target gap is 1 if it is closest to Z, 2 if it is the second closest gap etc.
The operation **Generate Identical** is chosen if $F_i = E_i$ and the overall frequency of the German cept F_i is 1.

generated operation sequence $O = (o_1, \ldots, o_J)$ as:

$$p(F, E) \approx \prod_{j=1}^{J} p(o_j | o_{j-m+1} ... o_{j-1})$$

where m indicates the amount of context used. Our translation model is implemented as an N-gram model of operations using SRILM-Toolkit (Stolcke, 2002) with Kneser-Ney smoothing. We use a 9-gram model ($m = 8$).

Integrating the language model the search is defined as:

$$\hat{E} = \arg\max_{E} p_{LM}(E) p(F, E)$$

where $p_{LM}(E)$ is the monolingual language model and $p(F, E)$ is the translation model. But our translation model is a joint probability model, because of which E is generated twice in the numerator. We add a factor, prior probability $p_{pr}(E)$, in the denominator, to negate this effect. It is used to marginalize the joint-probability model $p(F, E)$. The search is then redefined as:

$$\hat{E} = \arg\max_{E} p_{LM}(E) \frac{p(F, E)}{p_{pr}(E)}$$

Both, the monolingual language and the prior probability model are implemented as standard word-based n-gram models:

$$p_x(E) \approx \prod_{j=1}^{J} p(w_j | w_{j-m+1}, \ldots, w_{j-1})$$

where $m = 4$ (5-gram model) for the standard monolingual model ($x = LM$) and $m = 8$ (same as the operation model[5]) for the prior probability model ($x = pr$).

In order to improve end-to-end accuracy, we introduce new features for our model and shift from the generative[6] model to the standard log-linear approach (Och and Ney, 2004) to tune[7] them. We search for a target string E which maximizes a linear combination of feature functions:

[5] In decoding, the amount of context used for the prior probability is synchronized with the position of back-off in the operation model.

[6] Our generative model is about 3 BLEU points worse than the best discriminative results.

[7] We tune the operation, monolingual and prior probability models as separate features. We expect the prior probability model to get a negative weight but we do not force MERT to assign a negative weight to this feature.

$$\hat{E} = \arg\max_{E} \left\{ \sum_{j=1}^{J} \lambda_j h_j(F, E) \right\}$$

where λ_j is the weight associated with the feature $h_j(F, E)$. Other than the 3 features discussed above (log probabilities of the operation model, monolingual language model and prior probability model), we train 8 additional features discussed below:

Length Bonus The length bonus feature counts the length of the target sentence in words.

Deletion Penalty Another feature for avoiding too short translations is the deletion penalty. Deleting a source word (Generate Source Only (X)) is a common operation in the generative story. Because there is no corresponding target-side word, the monolingual language model score tends to favor this operation. The deletion penalty counts the number of deleted source words.

Gap Bonus and Open Gap Penalty These features are introduced to guide the reordering decisions. We observe a large amount of reordering in the automatically word aligned training text. However, given only the source sentence (and little world knowledge), it is not realistic to try to model the reasons for all of this reordering. Therefore we can use a more robust model that reorders less than humans. The gap bonus feature sums to the total number of gaps inserted to produce a target sentence. The open gap penalty feature is a penalty (paid once for each translation operation (Generate, Generate Identical, Generate Source Only) performed) whose value is the number of open gaps. This penalty controls how quickly gaps are closed.

Distortion and Gap Distance Penalty We have two additional features to control the reordering decisions. One of them is similar[8] to the distance-based reordering model used by phrasal MT. The other feature is the gap distance penalty which calculates the distance between the first word of a source cept X and the start of the left-most gap. This cost is paid once for each Generate, Generate Identical and Generate Source Only. For a source cept coverd by indexes $X_1, \ldots, X_n$, we get the feature value $g_j = X_1 - S$, where S is the index of the left-most source word where a gap starts.

Lexical Features We also use source-to-target $p(e|f)$ and target-to-source $p(f|e)$ lexical translation probabilities. Our lexical features are standard (Koehn et al., 2003). The estimation is motivated by IBM Model-1. Given a tuple t_i with source words $f = f_1, f_2, \ldots, f_n$, target words $e = e_1, e_2, \ldots, e_m$ and an alignment a between the source word positions $x = 1, \ldots, n$ and the target word positions $y = 1, \ldots, m$, the lexical feature $p_w(f|e)$ is computed as follows:

$$p_w(f|e, a) = \prod_{x=1}^{n} \frac{1}{|\{y : (x, y) \in a\}|} \sum_{\forall (x,y) \in a} w(f_x|e_y)$$

$p_w(e|f, a)$ is computed in the same way.

5 Decoding

Our decoder for the new model performs a stack-based search with a beam-search algorithm similar to that used in Pharoah (Koehn, 2004a). Given an input sentence F, it first extracts a set of matching source-side cepts along with their n-best translations to form a tuple inventory. During hypothesis expansion, the decoder picks a tuple from the inventory and generates the sequence of operations required for the translation with this tuple in light of the previous hypothesis.[9] The sequence of operations may include translation (generate, continue source cept etc.) and reordering (gap insertions, jumps) operations. The decoder also calculates the overall cost of the new hypothesis. Recombination is performed on hypotheses having the same coverage vector, monolingual language model context, and operation model context. We do histogram-based pruning, maintaining the 500 best hypotheses for each stack.[10]

[8] Let $X_1, \ldots, X_n$ and $Y_1, \ldots, Y_m$ represent indexes of the source words covered by the tuples t_j and t_{j-1} respectively. The distance between t_j and t_{j-1} is given as $d_j = min(|X_k - Y_l| - 1) \; \forall \; X_k \in \{X_1, \ldots, X_n\}$ and $\forall \; Y_l \in \{Y_1, \ldots, Y_m\}$

[9] A hypothesis maintains the index of the last source word covered (j), the position of the right-most source word covered so far (Z), the number of open gaps, the number of gaps so far inserted, the previously generated operations, the generated target string, and the accumulated values of all the features discussed in Section 4.

[10] We need a higher beam size to produce translation units similar to the phrase-based systems. For example, the phrase-based system can learn the phrase pair "zum Beispiel – for example" and generate it in a single step placing it directly into the stack two words to the right. Our system generates this example with two separate tuple translations "zum – for" and "Beispiel – example" in two adjacent stacks. Because "zum – for" is not a frequent translation unit, it will be ranked quite low in the first

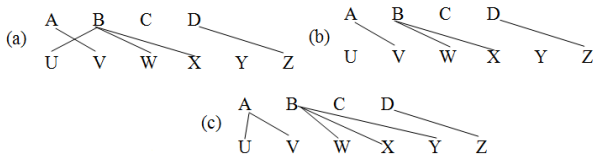

Figure 3: Post-editing of Alignments (a) Initial (b) No Target-Discontinuities (c) Final Alignments

6 Training

Training includes: (i) post-editing of the alignments, (ii) generation of the operation sequence (iii) estimation of the n-gram language models.

Our generative story does not handle target-side discontinuities and unaligned target words. Therefore we eliminate them from the training corpus in a 3-step process: If a source word is aligned with multiple target words which are not consecutive, first the link to the least frequent target word is identified, and the group of links containing this word is retained while the others are deleted. The intuition here is to keep the alignments containing content words (which are less frequent than functional words). The new alignment has no target-side discontinuities anymore, but might still contain unaligned target words. For each unaligned target word, we determine the (left or right) neighbour that it appears more frequently with and align it with the same source word as the neighbour. The result is an alignment without target-side discontinuities and unaligned target words. Figure 3 shows an illustrative example of the process. The tuples in Figure 3c are "A – U V", "B – W X Y", "C – NULL", "D – Z".

We apply Algorithm 1 to convert the preprocessed aligned corpus into a sequence of translation operations. The resulting operation corpus contains one sequence of operations per sentence pair.

In the final training step, the three language models are trained using the SRILM Toolkit. The operation model is estimated from the operation corpus. The prior probability model is estimated from the target side part of the bilingual corpus. The monolingual language model is estimated from the target side of the bilingual corpus and additional monolingual data.

stack until the tuple "Beispiel – example" appears in the second stack. Koehn and his colleagues have repeatedly shown that increasing the Moses stack size from 200 to 1000 does not have a significant effect on translation into English, see (Koehn and Haddow, 2009) and other shared task papers.

7 Experimental Setup

7.1 Data

We evaluated the system on three data sets with German-to-English, Spanish-to-English and French-to-English news translations, respectively. We used data from the 4^{th} version of the *Europarl Corpus* and the *News Commentary* which was made available for the translation task of the *Fourth Workshop on Statistical Machine Translation*.[11] We use 200K bilingual sentences, composed by concatenating the entire news commentary ($\approx$ 74K sentences) and Europarl ($\approx$ 126K sentence), for the estimation of the translation model. Word alignments were generated with GIZA++ (Och and Ney, 2003), using the grow-diag-final-and heuristic (Koehn et al., 2005). In order to obtain the best alignment quality, the alignment task is performed on the entire parallel data and not just on the training data we use. All data is lowercased, and we use the Moses tokenizer and recapitalizer. Our monolingual language model is trained on 500K sentences. These comprise 300K sentences from the monolingual corpus (news commentary) and 200K sentences from the target-side part of the bilingual corpus. The latter part is also used to train the prior probability model. The dev and test sets are news-dev2009a and news-dev2009b which contain 1025 and 1026 parallel sentences. The feature weights are tuned with Z-MERT (Zaidan, 2009).

7.2 Results

Baseline: We compare our model to a recent version of Moses (Koehn et al., 2007) using Koehn's training scripts and evaluate with BLEU (Papineni et al., 2002). We provide Moses with the same initial alignments as we are using to train our system.[12] We use the default parameters for Moses, and a 5-gram English language model (the same as in our system).

We compare two variants of our system. The first system (Tw_{no-rl}) applies no hard reordering limit and uses the distortion and gap distance penalty features as soft constraints, allowing all possible reorderings. The second system (Tw_{rl-6}) uses no distortion and gap distance features, but applies a hard constraint which limits reordering to no more than 6

[11] http://www.statmt.org/wmt09/translation-task.html

[12] We tried applying our post-processing to the alignments provided to Moses and found that this made little difference.

Source	German	Spanish	French
$\mathbf{Bl_{no-rl}}$	17.41	19.85	19.39
$\mathbf{Bl_{rl-6}}$	18.57	21.67	20.84
$\mathbf{Tw_{no-rl}}$	**18.97**	**22.17**	20.94
$\mathbf{Tw_{rl-6}}$	**19.03**	21.88	20.72

Table 3: This Work(Tw) vs Moses (Bl), no-rl = No Reordering Limit, rl-6 = Reordering limit 6

positions. Specifically, we do not extend hypotheses that are more than 6 words apart from the first word of the left-most gap during decoding. In this experiment, we disallowed tuples which were discontinuous on the source side. We compare our systems with two Moses systems as baseline, one using no reordering limit (Bl_{no-rl}) and one using the default distortion limit of 6 (Bl_{rl-6}).

Both of our systems (see Table 3) outperform Moses on the German-to-English and Spanish-to-English tasks and get comparable results for French-to-English. Our best system (Tw_{no-rl}), which uses no hard reordering limit, gives statistically significant ($p < 0.05$)[13] improvements over Moses (both baselines) for the German-to-English and Spanish-to-English translation task. The results for Moses drop by more than a BLEU point without the reordering limit (see Bl_{no-rl} in Table 3). All our results are statistically significant over the baseline Bl_{no-rl} for all the language pairs.

In another experiment, we tested our system also with tuples which were discontinuous on the source side. These gappy translation units neither improved the performance of the system with hard reordering limit ($Tw_{rl-6-asg}$) nor that of the system without reordering limit ($Tw_{no-rl-asg}$) as Table 4 shows. In an analysis of the output we found two reasons for this result: (i) Using tuples with source gaps increases the list of extracted n-best translation tuples exponentially which makes the search problem even more difficult. Table 5 shows the number of tuples (with and without gaps) extracted when decoding the test file with 10-best translations. (ii) The future cost[14] is poorly estimated in case of tuples with gappy source cepts, causing search errors.

In an experiment, we deleted gappy tuples with

[13]We used Kevin Gimpel's implementation of pairwise bootstrap resampling (Koehn, 2004b), 1000 samples.

[14]The dynamic programming approach of calculating future cost for bigger spans gives erroneous results when gappy cepts can interleave. Details omitted due to space limitations.

Source	German	Spanish	French
$\mathbf{Tw_{no-rl-asg}}$	18.61	21.60	20.59
$\mathbf{Tw_{rl-6-asg}}$	18.65	21.40	20.47
$\mathbf{Tw_{no-rl-hsg}}$	18.91	21.93	20.87
$\mathbf{Tw_{rl-6-hsg}}$	**19.23**	21.79	20.85

Table 4: Our Systems with Gappy Units, asg = All Gappy Units, hsg = Heuristic for pruning Gappy Units

Source	German	Spanish	French
Gaps	965515	1705156	1473798
No-Gaps	256992	313690	343220
Heuristic (hsg)	281618	346993	385869

Table 5: 10-best Translation Options With & Without Gaps and using our Heuristic

a score (future cost estimate) lower than the sum of the best scores of the parts. This heuristic removes many useless discontinuous tuples. We found that results improved ($Tw_{no-rl-hsg}$ and $Tw_{rl-6-hsg}$ in Table 4) compared to the version using all gaps ($Tw_{no-rl-asg}$, $Tw_{rl-6-asg}$), and are closer to the results without discontinuous tuples (Tw_{no-rl} and Tw_{rl-6} in Table 3).

8 Sample Output

In this section we compare the output of our systems and Moses. Example 1 in Figure 4 shows the powerful reordering mechanism of our model which moves the English verb phrase "do not want to negotiate" to its correct position between the subject "they" and the prepositional phrase "about concrete figures". Moses failed to produce the correct word order in this example. Notice that although our model is using smaller translation units "nicht – do not", "verhandlen – negotiate" and "wollen – want to", it is able to memorize the phrase translation "nicht verhandlen wollen – do not want to negotiate" as a sequence of translation and reordering operations. It learns the reordering of "verhandlen – negotiate" and "wollen – want to" and also captures dependencies across phrase boundaries.

Example 2 shows how our system without a reordering limit moves the English translation "vote" of the German clause-final verb "stimmen" across about 20 English tokens to its correct position behind the auxiliary "would".

Example 3 shows how the system with gappy tuples translates a German sentence with the particle verb "kehrten...zurück" using a single tuple (dashed lines). Handling phenomena like particle verbs

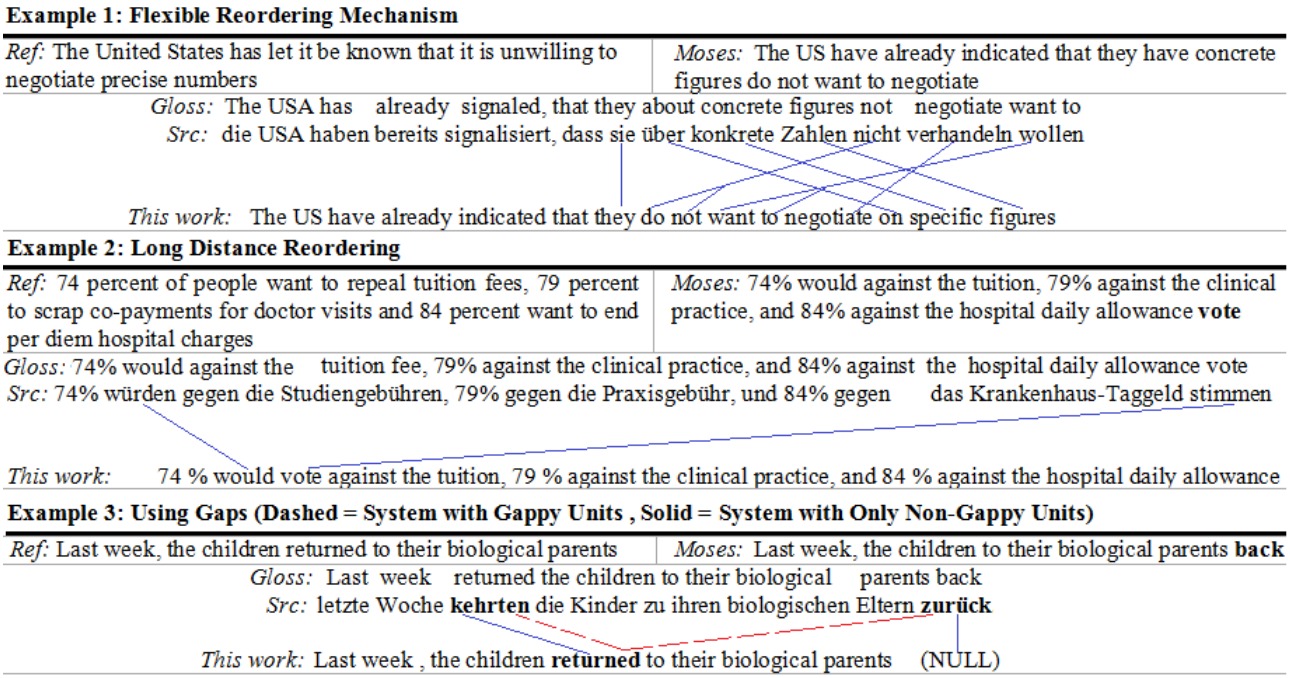

Figure 4: Sample Output Sentences

strongly motivates our treatment of source side gaps. The system without gappy units happens to produce the same translation by translating "kehrten" to "returned" and deleting the particle "zurück" (solid lines). This is surprising because the operation for translating "kehrten" to "returned" and for deleting the particle are too far apart to influence each other in an n-gram model. Moses run on the same example deletes the main verb ("kehrten"), an error that we frequently observed in the output of Moses.

Our last example (Figure 5) shows that our model learns idioms like "meiner Meinung nach – In my opinion ," and short phrases like "gibt es – there are" showing its ability to memorize these "phrasal" translations, just like Moses.

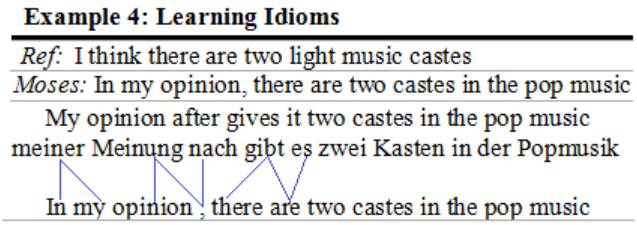

Figure 5: Learning Idioms

9 Conclusion

We have presented a new model for statistical MT which can be used as an alternative to phrase-based translation. Similar to N-gram based MT, it addresses three drawbacks of traditional phrasal MT by better handling dependencies across phrase boundaries, using source-side gaps, and solving the phrasal segmentation problem. In contrast to N-gram based MT, our model has a generative story which tightly couples translation and reordering. Furthermore it considers all possible reorderings unlike N-gram systems that perform search only on a limited number of pre-calculated orderings. Our model is able to correctly reorder words across large distances, and it memorizes frequent phrasal translations including their reordering as probable operations sequences. Our system outperformed Moses on standard Spanish-to-English and German-to-English tasks and achieved comparable results for French-to-English. A binary version of the corpus conversion algorithm and the decoder is available.[15]

Acknowledgments

The authors thank Fabienne Braune and the reviewers for their comments. Nadir Durrani was funded by the Higher Education Commission (HEC) of Pakistan. Alexander Fraser was funded by Deutsche Forschungsgemeinschaft grant Models of Morphosyntax for Statistical Machine Translation. Helmut Schmid was supported by Deutsche Forschungsgemeinschaft grant SFB 732.

[15] http://www.ims.uni-stuttgart.de/∼durrani/resources.html

References

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

David Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics*, 33(2):201–228.

Josep Maria Crego and François Yvon. 2009. Gappy translation units under left-to-right SMT decoding. In *Proceedings of the Meeting of the European Association for Machine Translation (EAMT)*, pages 66–73, Barcelona, Spain.

Josep Maria Crego and François Yvon. 2010. Improving reordering with linguistically informed bilingual n-grams. In *Coling 2010: Posters*, pages 197–205, Beijing, China, August. Coling 2010 Organizing Committee.

Josep M. Crego, Marta R. Costa-Jussà, José B. Mariño, and José A. R. Fonollosa. 2005a. Ngram-based versus phrase-based statistical machine translation. In *Proceedings of the International Workshop on Spoken Language Technology (IWSLT05)*, pages 177–184.

Josep M. Crego, José B. Mariño, and Adrià de Gispert. 2005b. Reordered search and unfolding tuples for ngram-based SMT. In *Proceedings of the 10th Machine Translation Summit (MT Summit X)*, pages 283–289, Phuket, Thailand.

Michel Galley and Christopher D. Manning. 2010. Accurate non-hierarchical phrase-based translation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 966–974, Los Angeles, California, June. Association for Computational Linguistics.

Philipp Koehn and Barry Haddow. 2009. Edinburgh's submission to all tracks of the WMT 2009 shared task with reordering and speed improvements to Moses. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 160–164, Athens, Greece, March. Association for Computational Linguistics.

Philipp Koehn, Franz J. Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the Human Language Technology and North American Association for Computational Linguistics Conference*, pages 127–133, Edmonton, Canada.

Philipp Koehn, Amittai Axelrod, Alexandra Birch Mayne, Chris Callison-Burch, Miles Osborne, and David Talbot. 2005. Edinburgh system description for the 2005 iwslt speech translation evaluation. In *International Workshop on Spoken Language Translation 2005*.

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 *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics, Demonstration Program*, Prague, Czech Republic.

Philipp Koehn. 2004a. Pharaoh: A beam search decoder for phrase-based statistical machine translation models. In *AMTA*, pages 115–124.

Philipp Koehn. 2004b. Statistical significance tests for machine translation evaluation. In Dekang Lin and Dekai Wu, editors, *Proceedings of EMNLP 2004*, pages 388–395, Barcelona, Spain, July. Association for Computational Linguistics.

Zhifei Li, Chris Callison-Burch, Chris Dyer, Juri Ganitkevitch, Sanjeev Khudanpur, Lane Schwartz, Wren N. G. Thornton, Jonathan Weese, and Omar F. Zaidan. 2009. Joshua: An open source toolkit for parsing-based machine translation.

J.B. Mariño, R.E. Banchs, J.M. Crego, A. de Gispert, P. Lambert, J.A.R. Fonollosa, and M.R. Costa-jussà. 2006. N-gram-based machine translation. *Computational Linguistics*, 32(4):527–549.

I. Dan Melamed. 2004. Statistical machine translation by parsing. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, Barcelona, Spain.

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

Franz J. Och and Hermann Ney. 2004. The alignment template approach to statistical machine translation. *Computational Linguistics*, 30(1):417–449.

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 Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, Morristown, NJ, USA. Association for Computational Linguistics.

Andreas Stolcke. 2002. SRILM - an extensible language modeling toolkit. In *Intl. Conf. Spoken Language Processing*, Denver, Colorado.

Omar F. Zaidan. 2009. Z-MERT: A fully configurable open source tool for minimum error rate training of machine translation systems. *The Prague Bulletin of Mathematical Linguistics*, 91:79–88.