The Operation Sequence Model — Combining N-Gram-Based and Phrase-Based Statistical Machine Translation

Nadir Durrani*
QCRI Qatar

Helmut Schmid**
LMU Munich

Alexander Fraser†
LMU Munich

Philipp Koehn‡
University of Edinburgh

Hinrich Schütze§
LMU Munich

In this article, we present a novel machine translation model, the Operation Sequence Model (OSM), which combines the benefits of phrase-based and N-gram-based statistical machine translation (SMT) and remedies their drawbacks. The model represents the translation process as a linear sequence of operations. The sequence includes not only translation operations but also reordering operations. As in N-gram-based SMT, the model is: (i) based on minimal translation units, (ii) takes both source and target information into account, (iii) does not make a phrasal independence assumption, and (iv) avoids the spurious phrasal segmentation problem. As in phrase-based SMT, the model (i) has the ability to memorize lexical reordering triggers, (ii) builds the search graph dynamically, and (iii) decodes with large translation units during search. The unique properties of the model are (i) its strong coupling of reordering and translation where translation and reordering decisions are conditioned on n *previous translation and reordering decisions, and (ii) the ability to model local and long-range reorderings consistently. Using BLEU as a metric of translation accuracy, we found that our system performs significantly*

* Qatar Computing Research Institute, Qatar Foundation. E-mail: `ndurrani@qf.org.qa`.
** CIS, Ludwig Maximilian University Munich. E-mail: `schmid@cis.uni-muenchen.de`.
† CIS, Ludwig Maximilian University Munich. E-mail: `fraser@cis.uni-muenchen.de`.
‡ University of Edinburgh, Edinburgh. E-mail: `pkoehn@inf.ed.ac.uk@inf.ed.ac.uk`.
§ CIS, Ludwig Maximilian University Munich. E-mail: `hs2014@cislmu.org`. Some of the research presented here was carried out while the authors were at the University of Stuttgart and the University of Edinburgh.

Submission received: 5 October 2013; revised version received: 23 October 2014; Accepted for publication: 25 November 2014.

doi:10.1162/COLI_a_00218

better than state-of-the-art phrase-based systems (Moses and Phrasal) and N-gram-based systems (Ncode) on standard translation tasks. We compare the reordering component of the OSM to the Moses lexical reordering model by integrating it into Moses. Our results show that OSM outperforms lexicalized reordering on all translation tasks. The translation quality is shown to be improved further by learning generalized representations with a POS-based OSM.

1. Introduction

Statistical Machine Translation (SMT) advanced near the beginning of the century from word-based models (Brown et al. 1993) towards more advanced models that take contextual information into account. Phrase-based (Koehn, Och, and Marcu 2003; Och and Ney 2004) and N-gram-based (Casacuberta and Vidal 2004; Mariño et al. 2006) models are two instances of such frameworks. Although the two models have some common properties, they are substantially different. The present work is a step towards combining the benefits and remedying the flaws of these two frameworks.

Phrase-based systems have a simple but effective mechanism that learns larger chunks of translation called bilingual phrases.[1] Memorizing larger units enables the phrase-based model to learn local dependencies such as short-distance reorderings, idiomatic collocations, and insertions and deletions that are internal to the phrase pair. The model, however, has the following drawbacks: (i) it makes independence assumptions over phrases, ignoring the contextual information outside of phrases, (ii) the reordering model has difficulties in dealing with long-range reorderings, (iii) problems in both search and modeling require the use of a hard reordering limit, and (iv) it has the spurious phrasal segmentation problem, which allows multiple derivations of a bilingual sentence pair that have the same word alignment but different model scores.

N-gram-based models are Markov models over sequences of tuples that are generated monotonically. Tuples are minimal translation units (MTUs) composed of source and target cepts.[2] The N-gram-based model has the following drawbacks: (i) only precalculated orderings are hypothesized during decoding, (ii) it cannot memorize and use lexical reordering triggers, (iii) it cannot perform long distance reorderings, and (iv) using tuples presents a more difficult search problem than in phrase-based SMT.

The Operation Sequence Model. In this article we present a novel model that tightly integrates translation and reordering into a single generative process. Our model explains the translation process as a linear sequence of operations that generates a source and target sentence in parallel, in a target left-to-right order. 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 that do the actual reordering. The probability of a sequence of operations is defined according to an N-gram model, that is, the probability of an operation depends on the $n-1$ preceding operations. Because the translation (lexical 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

1 A **Phrase pair** in phrase-based SMT is a pair of sequences of words. The sequences are not necessarily linguistic constituents. Phrase pairs are built by combining minimal translation units and ordering information. As is customary we use the term *phrase* to refer to phrase pairs if there is no ambiguity.

2 A **cept** is a group of source (or target) words connected to a group of target (or source) words in a particular alignment (Brown et al. 1993).

on preceding reordering decisions. This provides a natural reordering mechanism that is able to deal with local and long-distance reorderings in a consistent way.

Like the N-gram-based SMT model, the operation sequence model (OSM) is based on minimal translation units and takes both source and target information into account. This mechanism has several useful properties. Firstly, no phrasal independence assumption is made. The model has access to both source and target context outside of phrases. Secondly the model learns a unique derivation of a bilingual sentence given its alignments, thus avoiding the spurious phrasal segmentation problem. The OSM, however, uses operation N-grams (rather than tuple N-grams), which encapsulate both translation and reordering information. This allows the OSM to use lexical triggers for reordering like phrase-based SMT. Our reordering approach is entirely different from the tuple N-gram model. We consider all possible orderings instead of a small set of POS-based pre-calculated orderings, as is used in N-gram-based SMT, which makes their approach dependent on the availability of a source and target POS-tagger. We show that despite using POS tags the reordering patterns learned by N-gram-based SMT are not as general as those learned by our model.

Combining MTU-model with Phrase-Based Decoding. Using minimal translation units makes the search much more difficult because of the poor translation coverage, inaccurate future cost estimates, and pruning of correct hypotheses because of insufficient context. The ability to memorize and produce larger translation units gives an edge to the phrase-based systems during decoding, in terms of better search performance and superior selection of translation units. In this article, we combine N-gram-based modeling with phrase-based decoding to benefit from both approaches. Our model is based on minimal translation units, but we use phrases during decoding. Through an extensive evaluation we found that this combination not only improves the search accuracy but also the BLEU scores. Our in-house phrase-based decoder outperformed state-of-the-art phrase-based (Moses and Phrasal) and N-gram-based (NCode) systems on three translation tasks.

Comparative Experiments. Motivated by these results, we integrated the OSM into the state-of-the-art phrase-based system Moses (Koehn et al. 2007). Our aim was to directly compare the performance of the lexicalized reordering model to the OSM and to see whether we can improve the performance further by using both models together. Our integration of the OSM into Moses gave a statistically significant improvement over a competitive baseline system in most cases.

In order to assess the contribution of improved reordering versus the contribution of better modeling with MTUs in the OSM-augmented Moses system, we removed the reordering operations from the stream of operations. This is equivalent to integrating the conventional N-gram tuple sequence model (Mariño et al. 2006) into a phrase-based decoder, as also tried by Niehues et al. (2011). Small gains were observed in most cases, showing that much of the improvement obtained by the OSM is due to better reordering.

Generalized Operation Sequence Model. The primary strength of the OSM over the lexicalized reordering model is its ability to take advantage of the wider contextual information. In an error analysis we found that the lexically driven OSM often falls back to very small context sizes because of data sparsity. We show that this problem can be addressed by learning operation sequences over generalized representations such as POS tags.

The article is organized into seven sections. Section 2 is devoted to a literature review. We discuss the pros and cons of the phrase-based and N-gram-based SMT frameworks in terms of both model and search. Section 3 presents our model. We show how our model combines the benefits of both of the frameworks and removes their drawbacks. Section 4 provides an empirical evaluation of our preliminary system, which uses an MTU-based decoder, against state-of-the-art phrase-based (Moses and Phrasal) and N-gram-based (Ncode) systems on three standard tasks of translating German-to-English, Spanish-to-English, and French-to-English. Our results show improvements over the baseline systems, but we noticed that using minimal translation units during decoding makes the search problem difficult, which suggests using larger units in search. Section 5 presents an extension to our system to combine phrase-based decoding with the operation sequence model to address the problems in search. Section 5.1 empirically shows that information available in phrases can be used to improve the search performance and translation quality. Finally, we probe whether integrating our model into the phrase-based SMT framework addresses the mentioned drawbacks and improves translation quality. Section 6 provides an empirical evaluation of our integration on six standard tasks of translating German–English, French–English, and Spanish–English pairs. Our integration gives statistically significant improvements over submission quality baseline systems. Section 7 concludes.

2. Previous Work

2.1 Phrase-Based SMT

The phrase-based model (Koehn et al. 2003; Och and Ney 2004) segments a bilingual sentence pair into phrases that are continuous sequences of words. These phrases are then reordered through a lexicalized reordering model that takes into account the orientation of a phrase with respect to its previous phrase (Tillmann and Zhang 2005) or block of phrases (Galley and Manning 2008). Phrase-based models memorize local dependencies such as short reorderings, translations of idioms, and the insertion and deletion of words sensitive to local context. Phrase-based systems, however, have the following drawbacks.

Handling of Non-local Dependencies. Phrase-based SMT models dependencies between words and their translations inside of a phrase well. However, dependencies across phrase boundaries are ignored because of the strong phrasal independence assumption. Consider the bilingual sentence pair shown in Figure 1(a).

Reordering of the German word *stimmen* is internal to the phrase-pair *gegen ihre Kampagne stimmen* -'vote against your campaign' and therefore represented by the translation model. However, the model fails to correctly translate the test sentence shown in Figure 1(b), which is translated as 'they would for the legalization of abortion in Canada **vote**', failing to displace the verb. The language model does not provide enough

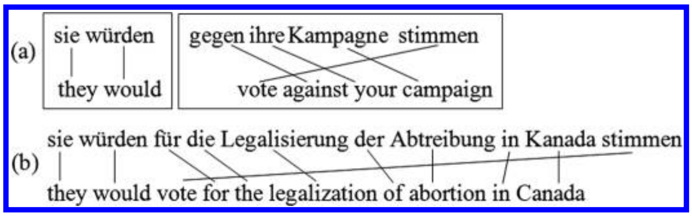

(a)
sie würden
they would
gegen ihre Kampagne stimmen
vote against your campaign
(b)
sie würden für die Legalisierung der Abtreibung in Kanada stimmen
they would vote for the legalization of abortion in Canada

Figure 1
(a) Training example with learned phrases. (b) Test sentence.

evidence to counter the dispreference of the translation model against jumping over the source words *für die Legalisieurung der Abtreibung in Kanada* and translating *stimmen* - 'vote' at its correct position.

Weak Reordering Model. The lexicalized reordering model is primarily designed to deal with short-distance movement of phrases such as swapping two adjacent phrases and cannot properly handle long-range jumps. The model only learns an orientation of how a phrase was reordered with respect to its previous and next phrase; it makes independence assumptions over previously translated phrases and does not take into account how previous words were translated and reordered. Although such an independence assumption is useful to reduce sparsity, it is overly generalizing and does not help to disambiguate good reorderings from the bad ones.

Moreover, a vast majority of extracted phrases are singletons and the corresponding probability of orientation given phrase-pair estimates are based on a single observation. Due to sparsity, the model falls back to use one-word phrases instead, the orientation of which is ambiguous and can only be judged based on context that is ignored. This drawback has been addressed by Cherry (2013) by using sparse features for reordering models.

Hard Distortion Limit. The lexicalized reordering model fails to filter out bad large-scale reorderings effectively (Koehn 2010). A hard distortion limit is therefore required during decoding in order to produce good translations. A distortion limit beyond eight words lets the translation accuracy drop because of search errors (Koehn et al. 2005). The use of a hard limit is undesirable for German–English and similar language pairs with significantly different syntactic structures. Several researchers have tried to address this problem. Moore and Quirk (2007) proposed improved future cost estimation to enable higher distortion limits in phrasal MT. Green, Galley, and Manning (2010) additionally proposed discriminative distortion models to achieve better translation accuracy than the baseline phrase-based system for a distortion limit of 15 words. Bisazza and Federico (2013) recently proposed a novel method to dynamically select which long-range reorderings to consider during the hypothesis extension process in a phrase-based decoder and showed an improvement in a German–English task by increasing the distortion limit to 18.

Spurious Phrasal Segmentation. A problem with the phrase-based model is that there is no unique correct phrasal segmentation of a sentence. Therefore, all possible ways of segmenting a bilingual sentence consistent with the word alignment are learned and used. This leads to two problems: (i) phrase frequencies are obtained by counting all possible occurrences in the training corpus, and (ii) different segmentations producing the same translation are generated during decoding. The former leads to questionable parameter estimates and the latter may lead to search errors because the probability of a translation is fragmented across different segmentations. Furthermore, the diversity in N-best translation lists is reduced.

2.2 N-Gram-Based SMT

N-gram-based SMT (Mariño et al. 2006) uses an N-gram model that jointly generates the source and target strings as a sequence of bilingual translation units called tuples. **Tuples** are essentially minimal phrases, atomic units that cannot be decomposed any further. The tuples are generated left to right in target word order. Reordering is not

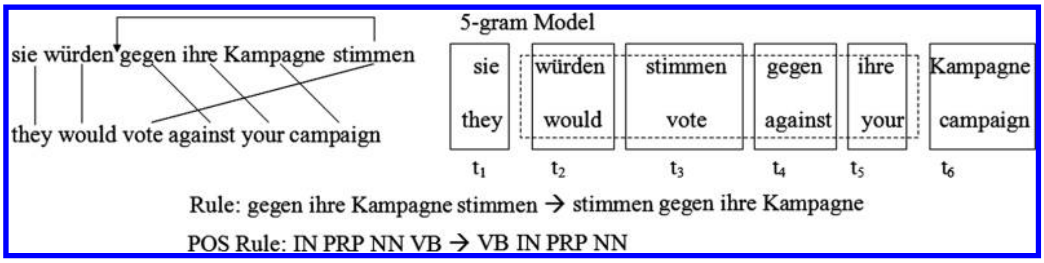

sie würden gegen ihre Kampagne stimmen
they would vote against your campaign
5-gram Model
sie
they
würden
would
stimmen
vote
gegen
against
ihre
your
Kampagne
campaign
t1 t2 t3 t4 t5 t6
Rule: gegen ihre Kampagne stimmen → stimmen gegen ihre Kampagne
POS Rule: IN PRP NN VB → VB IN PRP NN

Figure 2
POS-based reordering in N-gram-based SMT: Learned rules.

part of the statistical model. The parameters of the N-gram model are learned from bilingual data where the tuples have been arranged in target word order (see Figure 2).

Decoders for N-gram-based SMT reorder the source words in a preprocessing step so that the translation can be done monotonically. The reordering is performed with POS-based rewrite rules (see Figure 2 for an example) that have been learned from the training data (Crego and Mariño 2006). Word lattices are used to compactly represent a number of alternative reorderings. Using parts of speech instead of words in the rewrite rules makes them more general and helps to avoid data sparsity problems.

The mechanism has several useful properties. Because it is based on minimal units, there is only one derivation for each aligned bilingual sentence pair. The model therefore avoids spurious ambiguity. The model makes no phrasal independence assumption and generates a tuple monotonically by looking at a context of n previous tuples, thus capturing context across phrasal boundaries. On the other hand, N-gram-based systems have the following drawbacks.

Weak Reordering Model. The main drawback of N-gram-based SMT is its poor reordering mechanism. Firstly, by linearizing the source, N-gram-based SMT throws away useful information about how a particular word is reordered with respect to the previous word. This information is instead stored in the form of rewrite rules, which have no influence on the translation score. The model does not learn lexical reordering triggers and reorders through the learned rules only. Secondly, search is performed only on the precalculated word permutations created based on the source-side words. Often, evidence of the correct reordering is available in the translation model and the target-side language model. All potential reorderings that are not supported by the rewrite rules are pruned in the pre-processing step. To demonstrate this, consider the bilingual sentence pair in Figure 2 again. N-gram-based MT will linearize the word sequence *gegen ihre Kampagne stimmen* to *stimmen gegen ihre Kampagne*, so that it is in the same order as the English words. At the same time, it learns a POS rule: IN PRP NN VB $\rightarrow$ VB IN PRP NN. The POS-based rewrite rules serve to precompute the orderings that will be hypothesized during decoding. However, notice that this rule cannot generalize to the test sentence in Figure 1(b), even though the tuple translation model learned the trigram $<$ *sie* – 'they' *würden* – 'would' *stimmen* – 'vote' $>$ and it is likely that the monolingual language model has seen the trigram *they would vote*.

Hard Reordering Limit. Due to sparsity, only rules with seven or fewer tags are extracted. This subsequently constrains the reordering window to seven or fewer words, preventing the N-gram model from hypothesizing long-range reorderings that require larger jumps. The need to perform long-distance reordering motivated the idea of using syntax trees (Crego and Mariño 2007) to form rewrite rules. However, the rules are still extracted ignoring the target-side, and search is performed only on the precalculated orderings.

Difficult Search Problem. Using MTUs makes the search problem much more difficult because of poor translation option selection. To illustrate this consider the phrase pair *schoss ein Tor* – 'scored a goal', consisting of units *schoss* – 'scored', *ein* – 'a', and *Tor* – 'goal'. It is likely that the N-gram system does not have the tuple *schoss* – 'scored' in its N-best translation options because it is an uncommon translation. Even if *schoss* – 'scored' is hypothesized, it will be ranked quite low in the stack and may be pruned, before *ein* and *Tor* are generated in the next steps. A similar problem is also reported in Costa-jussà et al. (2007): When trying to reproduce the sentences in the N-best translation output of the phrase-based system, the N-gram-based system was able to produce only 37.5% of sentences in the Spanish-to-English and English-to-Spanish translation task, despite having been trained on the same word alignment. A phrase-based system, on the other hand, is likely to have access to the phrasal unit *schoss ein Tor* – 'scored a goal' and can generate it in a single step.

3. Operation Sequence Model

Now we present a novel generative model that explains the translation process as a linear sequence of operations that generate a source and target sentence in parallel. Possible operations are (i) generation of a sequence of source and/or target words, (ii) insertion of gaps as explicit target positions for reordering operations, and (iii) forward and backward jump operations that do the actual reordering. The probability of a sequence of operations is defined according to an N-gram model, that is, the probability of an operation depends on the $n-1$ preceding operations. Because 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 able to deal with local and long-distance reorderings consistently.

3.1 Generative Story

The generative story of the model is motivated by the complex reordering in the German-to-English translation task. The English words are generated in linear order,[3] and the German words are generated in parallel with their English translations. Mostly, the generation is done monotonically. Occasionally the translator inserts a gap on the German side to skip some words to be generated later. Each inserted gap acts as a designated landing site for the translator to jump back to. When the translator needs to cover the skipped words, it jumps back to one of the open gaps. After this is done, the translator jumps forward again and continues the translation. We will now, step by step, present the characteristics of the new model by means of examples.

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

3.1.1 Basic Operations. The generation of the German–English sentence pair *Peter liest* – 'Peter reads' is straightforward because it is a simple 1-to-1 word-based translation without reordering:

Generate (Peter , Peter) Generate (liest , reads)

3.1.2 Insertions and Deletions. The translation *Es ist ja nicht so schlimm* – 'it is not that bad', requires the **insertion** of an additional German word *ja*, which is used as a discourse particle in this construction.

Generate (Es , it) Generate (ist , is) ***Generate Source Only (ja)*** *Generate (nicht , not) Generate (so , that) Generate (schlimm , bad)*

Conversely, the translation *Lies mit* – 'Read with me' requires the **deletion** of an untranslated English word *me*.

Generate (Lies , Read) Generate (mit , with) ***Generate Target Only (me)***

3.1.3 Reordering. Let us now turn to an example that requires reordering, and revisit the example in Figure 1(a). The generation of this sentence in our model starts with generating *sie* – 'they', followed by the generation of *würden* – 'would'. Then a gap is inserted on the German side, followed by the generation of *stimmen* – 'vote'. At this point, the (partial) German and English sentences look as follows:

Operation Sequence	Generation
Generate(sie, they) Generate (würden, would) *Insert Gap Generate(stimmen, vote)*	sie würden $\boxed{}$ stimmen $\downarrow$ 'they would vote'

The arrow sign $\downarrow$ denotes the position after the previously covered German word. The translation proceeds as follows. We jump back to the open gap on the German side and fill it by generating *gegen* – 'against', *Ihre* – 'your' and *Kampagne* – 'campaign'. Let us discuss some useful properties of this mechanism:

1. We have learned a reordering pattern *sie würden* $\boxed{}$ *stimmen* – 'they would vote', which can be used to generalize the test sentence in Figure 1(b). In this case the translator jumps back and generates the tuples *für* – 'for', *die* – 'the', *Legalisierung* – 'legalization', *der* – 'of', *Abtreibung* – 'abortion', *in* – 'in', *Kanada* – 'Canada'.
2. The model handles both local (Figure 1 (a)) and long-range reorderings (Figure 1 (b)) in a unified manner, regardless of how many words separate *würden* and *stimmen*.
3. Learning the operation sequence *Generate(sie, they) Generate(würden, would) Insert Gap Generate(stimmen, vote)* is like learning a phrase pair *sie würden X stimmen* – 'they would vote'. The open gap represented by $\boxed{}$ acts as a placeholder for the skipped phrases and serves a similar purpose as the non-terminal category **X** in a discontinuous phrase-based system.
4. The model couples lexical generation and reordering information. Translation decisions are triggered by reordering decisions and vice

versa. Notice how the reordering decision is triggered by the translation decision in the example. The probability of a gap insertion operation after the generation of the auxiliaries *würden* – 'would' will be high because reordering is necessary in order to move the second part of the German verb complex (*stimmen*) to its correct position at the end of the clause.

Complex reorderings can be achieved by inserting multiple gaps and/or recursively inserting a gap within a gap. Consider the generation of the example in Figure 3 (borrowed from Chiang [2007]). The generation of this bilingual sentence pair proceeds as follows:

Generate(Aozhou, Australia) Generate(shi, is) Insert Gap Generate(zhiyi, one of)

At this point, the (partial) Chinese and English sentences look like this:

Aozhou shi $\boxed{}$ zhiyi ↓

Australia is one of

The translator now jumps back and recursively inserts a gap inside of the gap before continuing translation:

Jump Back (1) Insert Gap Generate(shaoshu, the few) Generate(guojia, countries)

Aozhou shi $\boxed{}$ shaoshu guojia ↓ zhiyi

Australia is one of the few countries

The rest of the sentence pair is generated as follows:

Jump Back (1) Insert Gap Generate(de, that) Jump Back (1) Insert Gap Generate(you, have) Generate(bangjiao, diplomatic relationships) Jump Back (1) Generate(yu, with) Generate(Beihan, North Korea)

Note that the translator jumps back and opens new gaps recursively to exhibit a property similar to the hierarchical model. However, our model uses a deterministic algorithm (see Algorithm 1 later in this article) to convert each bilingual sentence pair given the alignment to a unique derivation, thus avoiding spurious ambiguity unlike hierarchical and phrase-based models.

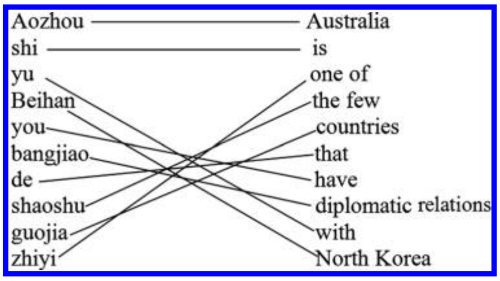


Figure 3
Recursive reordering.

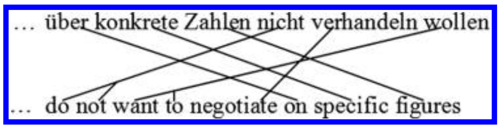

… über konkrete Zahlen nicht verhandeln wollen
… do not want to negotiate on specific figures

Figure 4
Subordinate German–English clause pair.

Multiple gaps can simultaneously exist at any time during generation. The translator decides based on the next English word to be covered which open gap to jump to. Figure 4 shows a German–English subordinate clause pair. The generation of this example is carried out as follows:

Insert Gap Generate(nicht, do not) Insert Gap Generate(wollen, want to)

At this point, the (partial) German and English sentences look as follows:

$\boxed{}$ nicht $\boxed{}$ wollen ↓

do not want to

The inserted gaps act as placeholders for the skipped prepositional phrase *über konkrete Zahlen* – 'on specific figures' and the verb phrase *verhandeln* – 'negotiate'. When the translator decides to generate any of the skipped words, it jumps back to one of the open gaps. The **Jump Back** operation closes the gap that it jumps to. The translator proceeds monotonically from that point until it needs to jump again. The generation proceeds as follows:

Jump Back (1) Generate(verhandeln, negotiate)

$\boxed{}$ nicht verhandeln ↓ wollen

do not want to negotiate

The translation ends by jumping back to the open gap and generating the prepositional phrase as follows:

Jump Back (1) Generate(über, on) Generate(konkrete, specific) Generate(Zahlen, figures)

5. Notice that although our model is based on minimal units, we can nevertheless memorize phrases (along with reordering information) through operation subsequences that are memorized by learning an N-gram model over these operation sequences. Some interesting phrases that our model learns are:

Phrases	Operation Sub-sequence
nicht X wollen – 'do not want to'	*Generate (nicht , do not) Insert Gap Generate (wollen , want to)*
verhandeln wollen – 'want to negotiate'	*Insert Gap Generate (wollen , want to) Jump Back(1) Generate (verhandeln , negotiate)*

X represents $\boxed{}$, the **Insert Gap** operation on the German side in our notation.

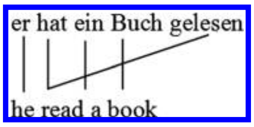


Figure 5
Discontinuous German-side cept.

3.1.4 Generation of Discontinuous Source Units. Now we discuss how discontinuous source cepts can be represented in our generative model. The Insert Gap operation discussed in the previous section can also be used to generate discontinuous source cepts. The generation of any such cept is done in several steps. See the example in Figure 5. The gappy cept *hat...gelesen* – 'read' can be generated as shown.

Operation Sequence	Generation
Generate(er, he) Generate (hat gelesen, read) Insert Gap Continue Source Cept	er hat □ gelesen ↓ he read

After the generation of *er* – 'he', the first part of the German complex verb *hat* is generated as an incomplete translation of 'read'. The second part *gelesen* is added to a queue to be generated later. A gap is then inserted for the skipped words *ein* and *Buch*. Lastly, the second word (*gelesen*) of the unfinished German cept *hat...gelesen* is added to complete the translation of 'read' through a **Continue Source Cept** operation. Discontinuous cepts on the English side cannot be generated analogously because of the fundamental assumption of the model that English (target-side) will be generated from left to right. This is a shortcoming of our approach, which we will discuss later in Section 4.1.

3.2 Definition of Operations

Our model uses five translation and three reordering operations, which are repeatedly applied in a sequence. The following is a definition of each of these operations.

3.3 Translation Operations

Generate (X,Y): X and Y are German and English cepts, 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 (target-side) is generated in linear order as per the assumption of the model.[4] The generation of the second (and subsequent) German words in a multiword cept can be delayed by gaps, jumps, and other operations defined in the following.

4 Note that when we are translating in the opposite direction (i.e., English-to-German), then German becomes target-side and is generated monotonically and gaps and jumps are performed on English (now source-side).

Continue Source Cept: The German words added 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 *kehrten...zurück* – 'returned' is generated by the operation **Generate (kehrten zurück, returned)**, which adds *kehrten* and 'returned' to the German and English strings and *zurück* to a queue. A **Continue Source Cept** operation later removes *zurück* from the queue and adds it to the German string.

Generate Source Only (X): The words in X are added at the current position in the German string. This operation is used to generate a German word 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 that indicates when to generate X.[5] **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 Target Only (Y): The words in Y are added at the current position in the English string. This operation is used to generate an English word with no corresponding German word. We do not utilize this operation in MTU-based decoding where it is hard to predict when to add unaligned target words during decoding. We therefore modified the alignments to remove this, by aligning unaligned target words (see Section 4.1 for details). In phrase-based decoding, however, this is not necessary, as we can easily predict unaligned target words where they are present in a phrase pair.

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 *QCRI* – 'QCRI', where German *QCRI* was observed exactly once during training, we use a **Generate Identical** operation rather than **Generate (QCRI, QCRI)**.

3.4 Reordering Operations

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 that specifies the position after the previously covered German word (j), an index (Z) that 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. Please refer to Algorithm 1 for details.

5 We want to preserve a 1-to-1 relationship between operation sequences and aligned sentence pairs. If we allowed an unaligned source word to be generated at any time, we would obtain several operation sequences that produce the same aligned sentence pair.

Algorithm 1 Corpus Conversion Algorithm

Input		**Output**	
$E_1 \ldots E_n$	English Cepts	<O> = $o_1 \ldots o_n$	Vector of Operations
$F_1 \ldots F_n$	German Cepts		
$a_1 \ldots a_n$	Alignment between E and F		
i	Position of current English cept	F_{a_i}	Sequence of German words linked to E_i
j	Position of current German word	$\lvert F_{a_i} \rvert$	# of German words linked with E_i
j'	Position of next German word	k	# of already generated German words for E_i
N	Total number of English cepts	a_{ik}	Position of k^{th} German translation of E_i
f_j	German word at position j	Z	Position after right-most generated German word
E_i	English cept at position i	$S(W)$	Position of the first word of a target gap W

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

while $f_j$ is an unaligned word do
    O.push(Generate Source Only ($f_j$))
    $j := j + 1$
while $E_i$ is an unaligned cept do
    O.push(Generate Target Only ($E_i$))
    $i := i + 1$
$Z := j$

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

Remarks: 1) We use cept positions for English (not word positions) because English cepts are composed of consecutive words. German positions are word-based. 2) The relative position of the target gap is 1 if it is closest to Z, 2 if it is the second closest gap, etc. 3) The operation **Generate Identical** is chosen if $F_i = E_i$ and count(F_i) is 1.

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

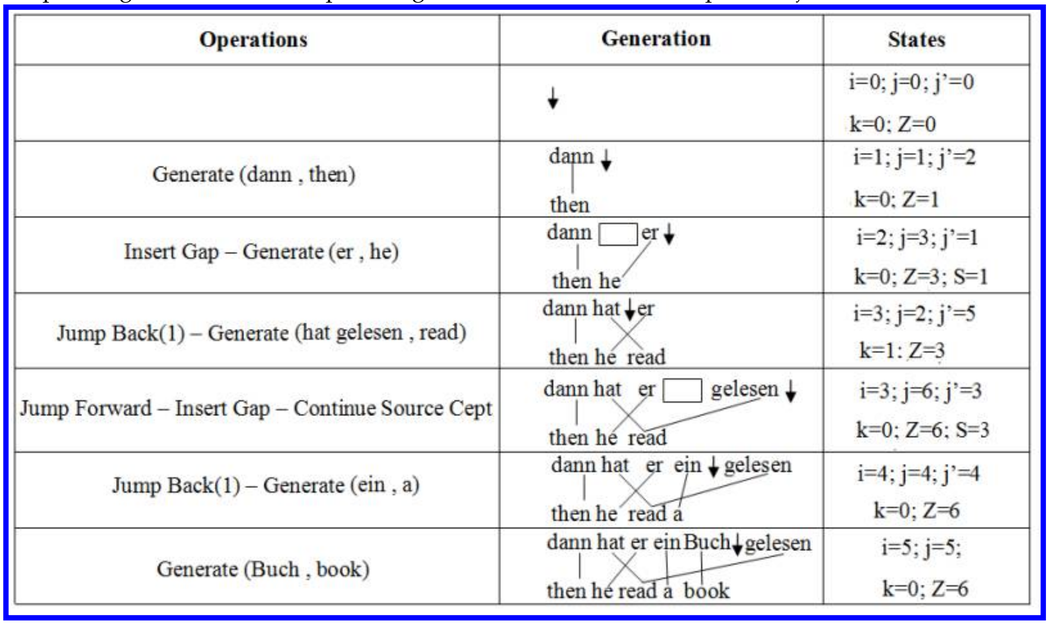
Jump Back (W): This operation lets the translator jump back to an open gap. It takes a parameter *W* specifying which gap to jump to. The **Jump Back (1)** operation jumps to the closest gap to *Z*, **Jump Back (2)** jumps to the second closest gap to *Z*, and so forth. After the backward jump, the target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

Jump Forward: This operation makes the translator jump to *Z*. It is performed when the next German word to be generated is to the right of the last German word generated and does not follow it immediately. It will be followed by an **Insert Gap** or **Jump Back (W)** operation if the next source word is not at position *Z*.

3.5 Conversion Algorithm

We use Algorithm 1 to convert an aligned bilingual sentence pair to a sequence of operations. Table 1 shows step by step by means of an example (Figure 6) how the conversion is done. The values of the index variables are displayed at each point.

Table 1
Step-wise generation of Example in Figure 6. The arrow indicates position *j*.



Operations	Generation	States

	↓	i=0; j=0; j'=0
		k=0; Z=0

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;
		k=0; Z=6

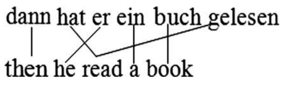


Figure 6
Discontinuous cept translation.

3.6 Model

Our model is estimated from a sequence of operations obtained through the transformation of a word-aligned bilingual corpus. An operation can be to generate source and target words or to perform reordering by inserting gaps and jumping forward and backward. Let $O = o_1, \ldots, o_J$ be a sequence of operations as hypothesized by the translator to generate a word-aligned bilingual sentence pair $< F, E, A >$. The translation model is then defined as:

$$p_T(F, E, A) = p(o_1, .., o_J) = \prod_{j=1}^{J} p(o_j | o_{j-n+1} ... o_{j-1})$$

where n indicates the amount of context used and A defines the word-alignment function between E and F. Our translation model is implemented as an N-gram model of operations using the SRILM toolkit (Stolcke 2002) with Kneser-Ney smoothing (Kneser and Ney 1995). The translate operations in our model (the operations with a name starting with Generate) encapsulate tuples. Tuples are minimal translation units extracted from the word-aligned corpus. The idea is similar to N-gram-based SMT except that the tuples in the N-gram model are generated monotonically. We do not impose the restriction of monotonicity in our model but integrate reordering operations inside the generative model.

As in the tuple N-gram model, there is a 1-to-1 correspondence between aligned sentence pairs and operation sequences, that is, we get exactly one operation sequence per bilingual sentence given its alignments. The corpus conversion algorithm (Algorithm 1) maps each bilingual sentence pair given its alignment into a unique sequence of operations deterministically, thus maintaining a 1-to-1 correspondence. This property of the model is useful because it addresses the spurious phrasal segmentation problem in phrase-based models. A phrase-based model assigns different scores to a derivation based on which phrasal segmentation is chosen. Unlike this, the OSM assigns only one score because the model does not suffer from spurious ambiguity.

3.6.1 Discriminative Model. We use a log-linear approach (Och 2003) to make use of standard features along with several novel features that we introduce to improve end-to-end accuracy. We search for a target string E that maximizes a linear combination of feature functions:

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

where λ_j is the weight associated with the feature $h_j(F, E)$. Apart from the OSM and standard features such as target-side language model, length bonus, distortion limit, and IBM lexical features (Koehn, Och, and Marcu 2003), we used the following new features:

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 and Open Gap Count. 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 do. The gap count feature sums to the total number of gaps inserted while producing a target sentence.

The open gap count feature is a penalty paid once for each translation operation (**Generate(X,Y), Generate Identical, Generate Source Only (X)**) performed whose value is the number of currently open gaps. This penalty controls how quickly gaps are closed.

Distance-Based Features. We have two distance-based features to control the reordering decisions. One of the features is the **Gap Distance,** which calculates the distance between the first word of a source cept X and the start of the leftmost gap. This cost is paid once for each translation operation (**Generate, Generate Identical, Generate Source Only (X)**). For a source cept covering the positions $X_1, \ldots, X_n$, 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. Another distance-based penalty used in our model is the **Source Gap Width**. This feature only applies in the case of a discontinuous translation unit and computes the distance between the words of a gappy cept. Let $f = f_1 \ldots, f_i, \ldots, f_n$ be a gappy source cept where x_i is the index of the i^{th} source word in the cept f. The value of the gap-width penalty is calculated as:

$$w_j = \sum_{i=2}^{n} x_i - x_{i-1} - 1$$

4. MTU-Based Search

We explored two decoding strategies in this work. Our first decoder complements the model and only uses minimal translation units in left-to-right stack-based decoding, similar to that used in Pharaoh (Koehn 2004a). The overall process can be roughly divided into the following steps: (i) extraction of translation units, (ii) future cost estimation, (iii) hypothesis extension, and (iv) recombination and pruning. The last two steps are repeated iteratively until all the words in the source sentence have been translated.

Our hypotheses maintain 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 3.6.1. The sequence of operations may include translation operations (generate, continue source cept, etc.) and reordering operations (gap insertions, jumps). Recombination[6] is performed on hypotheses having the same coverage vector, monolingual language model context, and OSM context. We do histogram-based pruning, maintaining the 500 best hypotheses for each stack. A large beam size is required to cope with the search errors that result from using minimal translation units during decoding. We address this problem in Section 5.

6 Note that although we are using minimal translation units, recombination is still useful as different derivations can arise through different alignments between source and target fragments. Also, recombination can still take place if hypotheses differ slightly in the output (Koehn 2010).

4.1 Handling Unaligned and Discontinuous Target Words

Aligned bilingual training corpora often contain unaligned target words and discontinuous target cepts, both of which pose problems. Unlike discontinuous source cepts, discontinuous target cepts such as *hinunterschüttete* – 'poured ... down' in constructions like *den Drink* ***hinunterschüttete*** – '**poured** the drink **down**' cannot be handled by the operation sequence model because it generates the English words in strict left-to-right order. Therefore they have to be eliminated.

Unaligned target words are only problematic for the MTU-based decoder, which has difficulties predicting where to insert them. Thus, we eliminate unaligned target words in MTU-based decoding.

We use a three-step process (Durrani, Schmid, and Fraser 2011) that modifies the alignments and removes unaligned and discontinuous targets. If a source word is aligned with multiple target words that are not consecutive, first the link to the least frequent target word is identified, and the group (consecutive adjacent words) 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). For example, the alignment link *hinunterschüttete* – 'down' is deleted and only the link *hinunterschüttete* – 'poured' is retained because 'down' occurs more frequently than 'poured'. Crego and Yvon (2009) used split tokens to deal with this phenomenon.

For MTU-based decoding we also need to deal with unaligned target words. For each unaligned target word, we determine the (left or right) neighbor that it appears more frequently with and align it with the same source word as this neighbor. Crego, de Gispert, and Mariño (2005) and Mariño et al. (2006) instead used lexical probabilities $p(f|e)$ obtained from IBM Model 1 (Brown et al. 1993) to decide whether to attach left or right. A more sophisticated strategy based on part-of-speech entropy was proposed by Gispert and Mariño (2006).

4.2 Initial Evaluation

We evaluated our systems on German-to-English, French-to-English, and Spanish-to-English news translation for the purpose of development and evaluation. We used data from the eighth version of the *Europarl Corpus* and the *News Commentary* made available for the translation task of the *Eighth Workshop on Statistical Machine Translation*.[7] The bilingual corpora contained roughly 2M bilingual sentence pairs, which we obtained by concatenating news commentary ($\approx$ 184K sentences) and Europarl for the estimation of the translation model. Word alignments were generated with GIZA++ (Och and Ney 2003), using the grow-diag-final-and heuristic[8] (Koehn et al. 2005). All data are lowercased, and we use the Moses tokenizer. We took news-test-2008 as the dev set for optimization and news-test 2009-2012 for testing. The feature weights are tuned with Z-MERT (Zaidan 2009).

4.2.1 Baseline Systems. We compared our system with (i) Moses[9] (Koehn et al. 2007), (ii) Phrasal[10] (Cer et al. 2010), and (iii) Ncode[11] (Crego, Yvon, and Mariño 2011). We used

7 `http://www.statmt.org/wmt13/translation-task.html`
8 We also tested other symmetrization heuristics such as "Union" and "Intersection" but found the GDFA heuristic gave best results for all language pairs.
9 `http://www.statmt.org/moses/`
10 `http://nlp.stanford.edu/phrasal/`
11 `http://www.limsi.fr/Individu/jmcrego/bincoder/`

all these toolkits with their default settings. Phrasal provides two main extensions to Moses: a hierarchical reordering model (Galley and Manning 2008) and discontinuous source and target phrases (Galley and Manning 2010). We used the default stack sizes of 100 for Moses,[12] 200 for Phrasal, and 25 for Ncode (with 2^n stacks). A 5-gram English language model is used. Both phrase-based systems use the 20 best translation options per source phrase; Ncode uses the 25 best tuple translations and a 4-gram tuple sequence model. A hard distortion limit of 6 is used in the default configuration of both phrase-based systems. Among the other defaults, we retained the hard source gap penalty of 15 and a target gap penalty of 7 in Phrasal. We provide Moses and Ncode with the same post-edited alignments[13] from which we had removed target-side discontinuities. We feed the original alignments to Phrasal because of its ability to learn discontinuous source and target phrases. All the systems use MERT for the optimization of the weight vector.

4.2.2 Training. Training steps include: (i) post-editing of the alignments (Section 4.1), (ii) generation of the operation sequence (Algorithm 1), and (iii) estimation of the N-gram translation (OSM) and language models using the SRILM toolkit (Stolcke 2002) with Kneser-Ney smoothing. We used 5-gram models.

4.2.3 Summary of Developmental Experiments. During the developent of the MTU-based decoder, we performed a number of experiments to obtain optimal settings for the system. We list here a summary of the results from those experiments:

- We found that discontinuous source-side cepts do not improve translation quality in most cases but increase the decoding time by multiple folds. We will therefore only use continuous cepts.
- We performed experiments by varying the distortion limit from the conventional window of 6 words to infinity (= no hard limit). We found that the performance of our system is robust when removing the hard reordering constraint and even saw a slight improvement in results in the case of German-to-English systems. Using no distortion limit, however, significantly increases the decoding time. We will therefore use a window of 16 words, which we found to be optimal on the development set.
- The performance of the MTU-based decoder is sensitive to the stack size. A high limit of 500 is required for decent search accuracy. We will discuss this further in the next section.
- We found using 10 best translation options for each extracted cept during decoding to be optimal.

4.2.4 Comparison with the Baseline Systems. In this section we compare our system ($\mathbf{OSM_{mtu}}$) with the three baseline systems. We used Kevin Gimpel's tester,[14] which uses bootstrap resampling (Koehn 2004b) to test which of our results are significantly better than the baseline results. We mark a baseline result with "*" in order to indicate

12 Using stack sizes from 200–1,000 did not improve results.

13 Using post-processed alignments gave better results than using the original alignments for these baseline systems.

14 `http://www.ark.cs.cmu.edu/MT/`

Table 2
Comparison on five test sets – $\mathbf{OSM_{mtu}}$ = OSM MTU-based decoder.

	Moses	$\mathbf{Phrasal_d}$	**Ncode**	$\mathbf{OSM_{mtu}}$
		German-to-English		
$\mathbf{WMT_{09}}$	*20.47	*20.78	*20.52	21.17
$\mathbf{WMT_{10}}$	*21.37	*21.91	*21.53	22.29
$\mathbf{WMT_{11}}$	*20.40	20.96	*20.21	21.05
$\mathbf{WMT_{12}}$	*20.85	21.06	*20.76	21.37
		French-to-English		
$\mathbf{WMT_{09}}$	*25.78	*25.87	26.15	26.22
$\mathbf{WMT_{10}}$	26.65	*25.87	26.89	26.59
$\mathbf{WMT_{11}}$	*27.37	27.62	27.46	27.75
$\mathbf{WMT_{12}}$	*27.15	27.76	27.55	27.66
		Spanish-to-English		
$\mathbf{WMT_{09}}$	25.90	26.13	25.91	25.90
$\mathbf{WMT_{10}}$	28.91	28.89	29.02	28.82
$\mathbf{WMT_{11}}$	28.84	28.98	28.93	28.95
$\mathbf{WMT_{12}}$	31.28	31.47	31.42	30.86

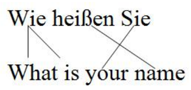
that our model shows a significant improvement over this baseline with a confidence of $p < 0.05$. We use 1,000 samples during bootstrap resampling.

Our German-to-English results (see Table 2) are significantly better than the baseline systems in most cases. Our French-to-English results show a significant improvement over Moses in three out of four cases, and over Phrasal in half of the cases. The N-gram-based system NCode was better or similar to our system on the French task. Our Spanish-to-English system also showed roughly the same translation quality as the baseline systems, but was significantly worse on the $\mathbf{WMT_{12}}$ task.

5. Phrase-Based Search

The MTU-based decoder is the most straightforward implementation of a decoder for the operation sequence model, but it faces search problems that cause a drop in translation accuracy. Although the OSM captures both source and target contexts and provides a better reordering mechanism, the ability to memorize and produce larger translation units gives an edge to the phrase-based model during decoding in terms of better search performance and superior selection of translation units. In this section, we combine N-gram-based modeling with phrase-based decoding. This combination not only improves search accuracy but also increases translation quality in terms of BLEU.

The operation sequence model, although based on minimal translation units, can learn larger translation chunks by memorizing a sequence of operations. However, it often has difficulties to produce the same translations as the phrase-based system because of the following drawbacks of MTU-based decoding: (i) the MTU-based decoder does not have access to all the translation units that a phrase-based decoder uses as part of a larger phrase, (ii) it requires a larger beam size to prevent early pruning of correct hypotheses, and (iii) it uses less-powerful future-cost estimates than the phrase-based decoder. To demonstrate these problems, consider the phrase pair



which the model memorizes through the sequence:

Generate(Wie, What is) Insert Gap Generate (Sie, your) Jump Back (1) Generate (heissen, name)

The MTU-based decoder needs three separate tuple translations to generate the same phrasal translation: *Wie* – 'What is', *Sie* – 'your' and *heißen* – 'name'. Here we are faced with three challenges.

Translation Coverage: The first problem is that the N-gram model does not have the same coverage of translation options. The English cepts 'What is', 'your', and 'name' are not good candidate translations for the German cepts *Wie*, *Sie*, and *heißen*, which are usually translated to 'How', 'you', and 'call', respectively, in isolation. When extracting tuple translations for these cepts from the Europarl data for our system, the tuple *Wie* – 'What is' is ranked 124th, *heißen* – 'name' is ranked 56th, and *Sie* – 'your' is ranked 9th in the list of *n*-best translation candidates. Typically, only the 20 best translation options are used, for the sake of efficiency, and such phrasal units with less frequent translations are never hypothesized in the N-gram-based systems. The phrase-based system, on the other hand, can extract the phrase *Wie heißen Sie* – 'what is your name' even if it is observed only once during training.

Larger Beam Size: Even when we allow a huge number of translation options and therefore hypothesize such units, we are faced with another challenge. A larger beam size is required in MTU-based decoding to prevent uncommon translations from getting pruned. The phrase-based system can generate the phrase pair *Wie heißen Sie* – 'what is your name' in a single step, placing it directly into the stack three words to the right. The MTU-based decoder generates this phrase in three stacks with the tuple translations *Wie* – 'What is', *Sie* – 'your', and *heißen* – 'name'. A very large stack size is required during decoding to prevent the pruning of *Wie* – 'What is', which is ranked quite low in the stack until the tuple *Sie* – 'your' is hypothesized in the next stack. Although the translation quality achieved by phrase-based SMT remains the same when varying the beam size, the performance of our system varies drastically with different beam sizes (especially for the German–English experiments where the search is more difficult due to a higher number of reorderings). Costa-jussà et al. (2007) also report a significant drop in the performance of N-gram-based SMT when a beam size of 10 is used instead of 50 in their experiments.

Future Cost Estimation: A third problem is caused by inaccurate future cost estimation. Using phrases helps phrase-based SMT to better estimate the future language model cost because of the larger context available, and allows the decoder to capture local (phrase-internal) reorderings in the future cost. In comparison, the future cost for tuples is based on unigram probabilities. The future cost estimate for the phrase pair *Wie heißen Sie* – 'What is your name' is estimated by calculating the cost of each feature.

A bigram language model cost, for example, is estimated in the phrase-based system as follows:

$$p_{lm} = p(\text{What}) \times p(\text{is}|\text{What}) \times p(\text{your}|\text{What is}) \times p(\text{name}|\text{What is your})$$

The translation model cost is estimated as:

$$p_{tm} = p(\text{What is your name}|\text{Wie heißen Sie})$$

Phrase-based SMT is aware during the preprocessing step that the words *Wie heißen Sie* may be translated as a phrase. This is helpful for estimating a more accurate future cost because the context is already available. The same is not true for the MTU-based decoder, to which only minimal units are available. The MTU-based decoder does not have the information that *Wie heißen Sie* may be translated as a phrase during decoding. The future cost estimate available to the operation sequence model for the span covering *Wie heißen Sie* will have unigram probabilities for both the translation and language models.

$$p_{lm} = p(\text{What}) \times p(\text{is}|\text{What}) \times p(\text{your}) \times p(\text{name})$$

The translation model cost is estimated as:

$$p_{tm} = p(\text{Generate(Wie, What is)}) \times p(\text{Generate(heißen,name)}) \times p(\text{Generate(Sie, your)})$$

A more accurate future cost estimate for the translation model cost would be:

$$\begin{aligned} p_{tm} = {} & p(\text{Generate(Wie,What is)}) \times p(\text{Insert Gap}|C_2) \times p(\text{Generate(Sie,your)}|C_3) \\ & \times p(\text{Jump Back(1)}|C_4) \times p(\text{Generate(heißen,name)}|C_5) \end{aligned}$$

where C_i is the context for the generation of the ith operation—that is, up to m previous operations. For example C_1 = *Generate(Wie, What is)*, C_2 = *Generate(Wie,What is) Insert Gap,* and so on. The future cost estimates computed in this manner are much more accurate because not only do they consider context, but they also take the reordering operations into account (Durrani, Fraser, and Schmid 2013).

5.1 Evaluating the Phrase-Based Decoder

We extended our in-house OSM decoder to use phrases instead of MTUs during decoding. In order to check whether phrase-based decoding solves the mentioned problems and improves the search accuracy, we evaluated the baseline MTU decoder and the phrase-based decoder with the same model parameters and tuned weights. This allows us to directly compare the model scores. We tuned the feature weights by running MERT with the MTU decoder on the dev set. Table 3 shows results from running both, the MTU-based (**OSM$_{\text{mtu}}$**) and the phrase-based (**OSM$_{\text{phr}}$**) decoder, on the **WMT$_{09}$** test set. Improved search accuracy is the percentage of times each decoder was able to produce a better model score than the other. Our phrase-based decoder uses a stack size of 200. Table 3 shows the percentage of times the MTU-based and phrase-based decoder produce better model scores than their counterpart. It shows that the phrase-based decoder produces better model scores for almost 48% of the hypotheses (on average)

Table 3
Comparing search accuracies of MTU-based (**OSM$_{mtu}$**) and phrase-based (**OSM$_{phr}$**) decoders.

System	**German**	**French**	**Spanish**	**Average**
	Improved Search Accuracy			
OSM$_{mtu}$	8.98%	8.88%	6.73%	8.2%
OSM$_{phr}$	56.20%	37.37%	49.36%	47.64%

across the three language pairs, whereas the MTU-based decoder (using a much higher stack size [500]) produces better hypotheses 8.2% of the time on average.

This improvement in search is also reflected in translation quality. Our phrase-based decoder outperforms the MTU-based decoder in all the cases and gives a significant improvement in 8 out of 12 cases (Table 4).

5.2 Handling of Unaligned and Discontinuous Target Words

In Section 4.1 we discussed the problem of handling unaligned and discontinuous target words in MTU-based decoding. An advantage of phrase-based decoding is that we can use such units during decoding if they appear within the extracted phrases. We use a **Generate Target Only (Y)** operation whenever the unaligned target word Y occurs in

Table 4
Comparison on four test sets – **OSM$_{mtu}$** = MTU-based decoder with stack size 500, **OSM$_{phr}$** = phrase-based decoder with stack size 200.

	Moses	**Phrasal$_d$**	**Ncode**	**OSM$_{mtu}$**	**OSM$_{phr}$**
		German-to-English			
WMT$_{09}$	*20.47	*20.78	*20.52	*21.17	21.47
WMT$_{10}$	*21.37	*21.91	*21.53	*22.29	22.73
WMT$_{11}$	*20.40	*20.96	*20.21	*21.05	21.43
WMT$_{12}$	*20.85	*21.06	*20.76	*21.37	21.98
		French-to-English			
WMT$_{09}$	*25.78	*25.87	*26.15	26.22	26.51
WMT$_{10}$	26.65	*25.87	26.89	26.59	26.88
WMT$_{11}$	*27.37	*27.62	*27.46	27.75	27.91
WMT$_{12}$	*27.15	27.76	*27.55	*27.66	27.98
		Spanish-to-English			
WMT$_{09}$	*25.90	26.13	*25.91	25.90	26.18
WMT$_{10}$	*28.91	*28.89	*29.02	*28.82	29.37
WMT$_{11}$	*28.84	*28.98	*28.93	*28.95	29.66
WMT$_{12}$	31.28	31.47	31.42	*30.86	31.52

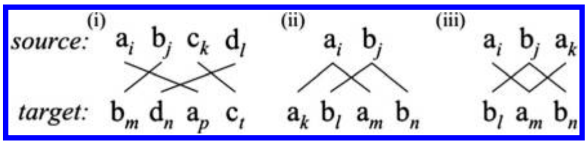


Figure 7
(i) Inside-out, (ii) CDTU (cross-serial discontinuous translation units), (iii) bonbon.

a phrase. Similarly, we use the operation **Generate (hinunterschüttete, poured down)** when the discontinuous tuple *hinunterschüttete* – 'poured ... down' occurs in a phrase. While training the model, we simply ignore the discontinuity and pretend that the word 'down' immediately follows 'poured'. This can be done by linearizing the subsequent parts of discontinuous target cepts to appear after the first word of the cept. During decoding we use phrase-internal alignments to hypothesize such a linearization. This is done only for the estimation of the OSM, and the target for all other purposes is generated in its original order. This heuristic allows us to deal with target discontinuities without extending the operation sequence model in complicated ways. It results in better BLEU accuracy in comparison with the post-editing of the alignments method described in Section 4.1. For details and empirical results refer to Durrani et al. (2013a) (see Table 2 therein, compare Rows 4 and 5).

Note that the OSM, like the discontinuous phrase-based model (Galley and Manning 2010), allows all possible geometries as shown in Figure 7. However, because our decoder only uses continuous phrases, we cannot hypothesize (ii) and (iii) unless they appear inside of a phrase. But our model could be integrated into a discontinuous phrase-based system to overcome this limitation.

6. Further Comparative Experiments

Our model, like the reordering models (Tillmann and Zhang 2005; Galley and Manning 2008) used in phrase-based decoders, is lexicalized. However, our model has richer conditioning as it considers both translation and reordering context across phrasal boundaries. The lexicalized reordering model used in phrase-based SMT only accounts for how a phrase pair was reordered with respect to its previous phrase (or block of phrases). Although such an independence assumption is useful to reduce sparsity, it is overgeneralizing, with only three possible orientations. Moreover, because most of the extracted phrases are observed only once, the corresponding probability of orientation given phrase-pair estimates is very sparse. The model often has to fall back to short one-word phrases. However, most short phrases are observed frequently with all possible orientations during training. This makes it difficult for the decoder to decide which orientation should be picked during decoding. The model therefore overly relies on the language model to break such ties. The OSM may also suffer from data sparsity and the back-off smoothing may fall back to very short contexts. But it might still be able to disambiguate better than the lexicalized reordering models. Also these drawbacks can be addressed by learning an OSM over generalized word representation such as POS tags, as we show in this section.

In an effort to make a comparison of the operation sequence model with the lexicalized reordering model, we incorporate the OSM into the phrase-based Moses decoder. This allows us to exactly compare the two models in identical settings. We integrate the OSM into the hypothesis extension process of the phrase-based decoder. We convert each phrase pair into a sequence of operations by extracting the MTUs within the phrase pair and using phrase internal alignments. The OSM is used as a feature in the log-linear framework. We also use four supportive features: the Gap, Open Gap, Gap-distance, and Deletion counts, as described earlier (see Section 3.6.1).

6.1 Baseline

Our Moses (Koehn et al. 2007) baseline systems are based on the setup described in Durrani et al. (2013b). We trained our systems with the following settings: maximum sentence length 80, grow-diag-final and symmetrization of GIZA++ alignments, an interpolated Kneser-Ney smoothed 5-gram language model with KenLM (Heafield 2011) used at runtime, distortion limit of 6, minimum Bayes-risk decoding (Kumar and Byrne 2004), cube pruning (Huang and Chiang 2007), and the no-reordering-over-punctuation heuristic. We used factored models (Koehn and Hoang 2007), for German–English and English–German. We trained the lexicalized reordering model (Koehn et al. 2005) with msd-bidirectional-fe settings.

6.2 Results

Table 5 shows that the OSM results in higher gains than the lexicalized reordering model on top of a plain phrase-based baseline (**Pb**). The average improvement obtained using the lexicalized reordering model (**Pb**$_{\textbf{lex}}$) over the baseline (**Pb**) is 0.50. In comparison, the average improvement obtained by using the OSM (**Pb**$_{\textbf{osm}}$) over the baseline (**Pb**) is 0.74. The average improvement obtained by the combination (**Pb**$_{\textbf{lex+osm}}$) is 0.97. The average improvement obtained by adding the OSM over the baseline (**Pb**$_{\textbf{lex}}$) is 0.47. We tested for significance and found that in seven out of eight cases adding the OSM on top of **Pb**$_{\textbf{lex}}$ gives a statistically significant improvement with a confidence of $p < 0.05$. Significant differences are marked with an asterisk.

6.3 Comparison with Tuple Sequence Model

In an additional experiment, we studied how much the translation quality decreases when all reordering operations are removed from the operation sequence model during

Table 5
Comparison against the lexicalized reordering model – Pb = baseline without lexical reordering. An asterisk indicates statistical significance over baseline (**Pb**$_{\textbf{lex}}$ = Pb + lexicalized reordering).

	Pb		**Pb**$_{\textbf{lex}}$		**Pb**$_{\textbf{osm}}$		**Pb**$_{\textbf{lex+osm}}$	
	MT$_{\textbf{12}}$	**MT**$_{\textbf{13}}$	**MT**$_{\textbf{12}}$	**MT**$_{\textbf{13}}$	**MT**$_{\textbf{12}}$	**MT**$_{\textbf{13}}$	**MT**$_{\textbf{12}}$	**MT**$_{\textbf{13}}$
FR-EN	30.19	30.73	30.74	30.89	30.77	*31.34	30.97	*31.48
EN-FR	28.45	29.62	28.98	30.06	29.16	30.46	*29.38	*30.54
ES-EN	33.64	29.86	34.07	30.25	34.24	*30.72	*34.43	*31.04
EN-ES	33.57	29.26	34.30	30.03	34.51	30.07	*34.71	*30.53
Avg	30.67		31.17 +0.50		31.41 +0.74		31.64 +0.97	

Table 6
Comparing the operation sequence model versus the tuple sequence model.

	Pb_{lex}		**$Pb_{lex+osm}$**		**$Pb_{lex+tsm}$**	
	MT_{12}	**MT_{13}**	**MT_{12}**	**MT_{13}**	**MT_{12}**	**MT_{13}**
DE-EN	22.95	25.26	*23.54	*26.01	23.18	25.51
EN-DE	17.95	20.16	18.10	20.43	17.90	20.20
FR-EN	30.74	30.89	30.97	*31.48	30.80	31.04
EN-FR	28.98	30.06	*29.38	*30.54	29.12	30.24
ES-EN	34.07	30.25	*34.43	*31.04	34.19	30.44
EN-ES	34.30	30.03	*34.71	*30.53	34.38	30.20
Avg	27.97		28.43 +0.46		28.10 +0.13	

training and decoding. The resulting model is similar to the tuple sequence model (TSM) of Mariño et al. (2006), except that we use phrase-internal reordering rather than POS-based rewrite rules to do the source linearization. Table 6 shows an average improvement of just 0.13 on top of the baseline phrase-based system with lexicalized reordering, which is much lower than the 0.46 points obtained with the full operation sequence model.

Bilingual translation models (without reordering) have been integrated into phrase-based systems before, either inside the decoder (Niehues et al. 2011) or to rerank the N-best candidate translations in the output of a phrase-based system (Zhang et al. 2013). Both groups reported improvements of similar magnitude when using a target-order left-to-right TSM model for German–English and French–English translation with shared task data, but higher gains on other data sets and language pairs. Zhang et al. (2013) showed further gains by combining models with target and source left-to-right and right-to-left orders. The assumption of generating the target in monotonic order is a weakness of our work that can be addressed following Zhang et al. (2013). By generating MTUs in source order and allowing gaps and jumps on the target side, the model will be able to learn other reordering patterns that are ignored by the standard OSM.

6.4 OSM over Generalized Representations

Because of data sparsity, it is impossible to observe all possible reordering patterns with all possible lexical choices in translation operations. The lexically driven OSM therefore often backs off to very small context sizes. Consider the example shown in Figure 1. The learned pattern *sie würden* $\square$ *stimmen* – 'they would vote' cannot be generalized to *er würde* $\square$ *wählen* – 'he would vote'. We found that the OSM uses only two preceding operations as context on average. This problem can be addressed by replacing words with POS tags (or any other generalized representation such as Morph tags, word clusters) to allow the model to consider a wider syntactic context where this is appropriate, thus improving lexical decisions and the reordering capability of the model. Crego and Yvon (2010) and Niehues et al. (2011) have shown improvements in

Table 7
Using generalized OSMs. s = surface; p = pos.

	$\mathbf{Pb}_{\mathbf{lex}}$		$\mathbf{Pb}_{\mathbf{lex+osm_{(s)}}}$		$\mathbf{Pb}_{\mathbf{lex+osm(s)+osm(p)}}$	
	$\mathbf{MT}_{12}$	$\mathbf{MT}_{13}$	$\mathbf{MT}_{12}$	$\mathbf{MT}_{13}$	$\mathbf{MT}_{12}$	$\mathbf{MT}_{13}$
DE-EN	22.95	25.26	23.54	26.01	23.78	26.30
EN-DE	17.95	20.16	18.10	20.43	18.33	20.70
Avg	21.58		22.02 +0.44		22.28 +0.70	

translation quality when using a TSM model over POS units. We estimate OSMs over generalized tags and add these as separate features to the loglinear framework.[15]

Experiments. We enabled factored sequence models (Koehn and Hoang 2007) in German–English language pairs as these have been shown to be useful previously. We used LoPar (Schmid 2000) to obtain morphological analysis and POS annotation of German and MXPOST (Ratnaparkhi 1998), a maximum entropy model for English POS tags. We simply estimate OSMs over POS tags[16] by replacing the words by the corresponding tags during training.

Table 7 shows that a system with an additional POS-based OSM ($\mathbf{Pb}_{\mathbf{lex+osm(s)+osm(p)}}$) gives an average improvement of +0.26 over the baseline ($\mathbf{Pb}_{\mathbf{lex+osm(s)}}$) system that uses an OSM over surface forms only. The overall gain by using OSMs over the baseline system is +0.70. OSM over surface tags considers 3-gram on average, and OSM over POS tags considers 4.5-grams on average, thus considering wider contextual information when making translation and reordering decisions.

6.5 Time Complexities and Memory Usage

Table 8 shows the wall-clock decoding time (in minutes) from running the Moses decoder (on news-test2013) with and without the OSMs. Each decoder is run with 24 threads on a machine with 140GB RAM and 24 processors. Timings vary between experiments because of the fact that machines were somewhat busy in some cases. But generally, the OSM increases decoding time by more than half an hour.[17]

Table 9 shows the overall sizes of phrase-based translation and reordering models along with the OSMs. It also shows the model sizes when filtered on news-test2013. A similar amount of reduction could be achieved by applying filtering to the OSMs following the language model filtering described by Heafield and Lavie (2010).

15 We also tried to amalgamate lexically driven OSM and generalized OSMs into a single model rather than using these as separate features. However, this attempt was unsuccessful (See Durrani et al. [2014] for details).

16 We also found using morphological tags and automatic word clusters to be useful in our recent IWSLT evaluation campaign (Birch, Durrani, and Koehn 2013; Durrani et al. 2014).

17 The code for the OSM in Moses can be greatly optimized but requires major modifications to source and target phrase classes in Moses.

Table 8
Wall-clock decoding times (in minutes) on WMT-13.

	Into English		From English	
	$\mathbf{Pb}_{\mathbf{lex}}$	$\mathbf{Pb}_{\mathbf{lex+osm}}$	$\mathbf{Pb}_{\mathbf{lex}}$	$\mathbf{Pb}_{\mathbf{lex+osm}}$
DE	61	88 Δ 27	143	158 Δ 15
FR	108	163 Δ 55	113	154 Δ 41
ES	111	142 Δ 31	74	109 Δ 35
Avg	93	131 Δ 38	110	140 Δ 30

Table 9
Data sizes (in number of sentences) and memory usage (in giga-bytes). Columns: Phrase translation and lexicalized reordering tables give overall model sizes/sizes when filtered on WMT-2013.

		Into English			From English		
	Data Sizes	**Phrase-Table**	**Lex. Reo**	**OSM**	**Phrase-Table**	**Lex. Reo**	**OSM**
DE	5.5M	5.8/0.59	1.5/0.14	2.0	4.9/0.14	1.6/0.24	2.1
FR	39M	28/0.35	9.4/0.99	14	28/0.33	9.7/1.2	14
ES	15.2M	9.2/0.68	3.2/0.25	4.4	9.0/0.76	3.2/0.28	4.4

7. Conclusion

In this article we presented a new model for statistical MT that combines the benefits of two state-of-the-art SMT frameworks, namely, N-gram-based and phrase-based SMT. Like the N-gram-based model, it addresses two drawbacks of phrasal MT by better handling dependencies across phrase boundaries, and solving the phrasal segmentation problem. In contrast to N-gram-based MT, our model has a generative story that tightly couples translation and reordering. Furthermore, it is able to consider 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 operation sequences.

We tested a version of our system that decodes based on minimal translation units (MTUs) against the state-of-the-art phrase-based systems Moses and Phrasal and the N-gram-based system Ncode for German-to-English, French-to-English, and Spanish-to-English on three standard test sets. Our system shows statistically significant improvements in 9 out of 12 cases in the German-to-English translation task, and 10 out of 12 cases in the French-to-English translation task. Our Spanish-to-English results are similar to the baseline systems in most of the cases but consistently worse than Ncode.

MTU-based decoding suffers from poor translation coverage, inaccurate future cost estimates, and pruning of correct hypotheses. Phrase-based SMT, on the other hand, avoids these drawbacks by using larger translation chunks during search. We therefore extended our decoder to use phrases instead of cepts while keeping the statistical model unchanged. We found that combining a model based on minimal units with phrase-based decoding improves both search accuracy and translation quality. Our system extended with phrase-based decoding showed improvements over all the baseline systems, including our MTU-based decoder. In most of the cases, the difference was significant.

Our results show that OSM consistently outperforms the Moses lexicalized reordering model and gives statistically significant gains over a very competitive Moses baseline system. We showed that considering both translation and reordering context is important and ignoring reordering context results in a significant reduction in the performance. We also showed that an OSM based on surface forms suffers from data sparsity and that an OSM based on a generalized representation with part-of-speech tags improves the translation quality by considering a larger context. In the future we would like to study whether the insight of using minimal units for modeling and search based on composed rules would hold for hierarchical SMT. Vaswani et al. (2011) recently showed that a Markov model over the derivation history of minimal rules can obtain the same translation quality as using grammars formed with composed rules, which we believe is quite promising.

Acknowledgments
We would like to thank the anonymous reviewers and Andreas Maletti and François Yvon for their helpful feedback and suggestions. The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreements 287658 (EU-Bridge) and 287688 (MateCat). 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. This publication only reflects the authors' views.

References

Birch, Alexandra, Nadir Durrani, and Philipp Koehn. 2013. Edinburgh SLT and MT System Description for the IWSLT 2013 Evaluation. In *Proceedings of the 10th International Workshop on Spoken Language Translation*, pages 40–48, Heidelberg.

Bisazza, Arianna and Marcello Federico. 2013. Efficient Solutions for Word Reordering in German-English Phrase-Based Statistical Machine Translation. In *Proceedings of the Eighth Workshop on Statistical Machine Translation*, pages 440–451, Sofia.

Brown, Peter F., 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.

Casacuberta, Francisco and Enrique Vidal. 2004. Machine Translation with Inferred Stochastic Finite-State Transducers. *Computational Linguistics*, 30:205–225.

Cer, Daniel, Michel Galley, Daniel Jurafsky, and Christopher D. Manning. 2010. Phrasal: A Statistical Machine Translation Toolkit for Exploring New Model Features. In *Proceedings of the North American Chapter of ACL 2010 Demonstration Session*, pages 9–12, Los Angeles, CA.

Cherry, Colin. 2013. Improved Reordering for Phrase-Based Translation Using Sparse Features. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 22–31, Atlanta, GA.

Chiang, David. 2007. Hierarchical Phrase-Based Translation. *Computational Linguistics*, 33(2):201–228.

Costa-jussà, Marta R., Josep M. Crego, David Vilar, José A. R. Fonollosa, José B. Mariño, and Hermann Ney. 2007. Analysis and System Combination of Phrase- and N-Gram-Based Statistical Machine Translation Systems. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Companion Volume, Short Papers*, pages 137–140, Rochester, NY.

Crego, Josep M., Adrià de Gispert, and José B. Mariño. 2005. The TALP

N-gram-based SMT System for IWSLT'05. In *Proceedings of the International Workshop on Spoken Language Translation*, pages 116–122, Phuket.

Crego, Josep M. and José B. Mariño. 2006. Improving Statistical MT by Coupling Reordering and Decoding. *Machine Translation*, 20(3):199–215.

Crego, Josep M. and José B. Mariño. 2007. Syntax-Enhanced N-gram-Based SMT. In *Proceedings of the 11th Machine Translation Summit*, pages 111–118, Copenhagen.

Crego, Josep M. 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*, pages 66–73, Barcelona.

Crego, Josep M. and François Yvon. 2010. Improving Reordering with Linguistically Informed Bilingual N-Grams. In *COLING 2010: Posters*, pages 197–205, Beijing.

Crego, Josep M., François Yvon, and José B. Mariño. 2011. Ncode: an Open Source Bilingual N-gram SMT Toolkit. *The Prague Bulletin of Mathematical Linguistics*, 96:49–58.

Durrani, Nadir, Alexander Fraser, and Helmut Schmid. 2013. Model With Minimal Translation Units, But Decode With Phrases. In the *2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1–11, Atlanta, GA.

Durrani, Nadir, Alexander Fraser, Helmut Schmid, Hieu Hoang, and Philipp Koehn. 2013a. Can Markov Models Over Minimal Translation Units Help Phrase-Based SMT? In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 399–405, Sofia.

Durrani, Nadir, Barry Haddow, Kenneth Heafield, and Philipp Koehn. 2013b. Edinburgh's Machine Translation Systems for European Language Pairs. In *Proceedings of the Eighth Workshop on Statistical Machine Translation*, pages 114–121, Sofia.

Durrani, Nadir, Philipp Koehn, Helmut Schmid, and Alexander Fraser. 2014. Investigating the Usefulness of Generalized Word Representations in SMT. In *Proceedings of the 25th Annual Conference on Computational Linguistics (COLING)*, pages 421–432, Dublin.

Durrani, Nadir, Helmut Schmid, and Alexander Fraser. 2011. A Joint Sequence Translation Model with Integrated Reordering. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1,045–1,054, Portland, OR.

Galley, Michel and Christopher D. Manning. 2008. A Simple and Effective Hierarchical Phrase Reordering Model. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 848–856, Honolulu, HI.

Galley, Michel 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, CA.

Gispert, Adrià and José B. Mariño. 2006. Linguistic Tuple Segmentation in N-Gram-Based Statistical Machine Translation. In *INTERSPEECH*, pages 1,149–1,152, Pittsburgh, PA.

Green, Spence, Michel Galley, and Christopher D. Manning. 2010. Improved Models of Distortion Cost for Statistical Machine Translation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 867–875, Los Angeles, CA.

Heafield, Kenneth. 2011. KenLM: Faster and Smaller Language Model Queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 187–197, Edinburgh.

Heafield, Kenneth and Alon Lavie. 2010. Combining Machine Translation Output with Open Source: The Carnegie Mellon Multi-Engine Machine Translation Scheme. *The Prague Bulletin of Mathematical Linguistics*, 93:27–36.

Huang, Liang and David Chiang. 2007. Forest Rescoring: Faster Decoding with Integrated Language Models. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 144–151, Prague.

Kneser, Reinhard and Hermann Ney. 1995. Improved Backing-off for M-gram Language Modeling. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 181–184.

Koehn, Philipp. 2004a. Pharaoh: A Beam Search Decoder for Phrase-Based Statistical Machine Translation Models. In *Association for Machine Translation in the Americas*, pages 115–124, Washington, DC.

Koehn, Philipp. 2004b. Statistical Significance Tests for Machine Translation Evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 388–395, Barcelona.

Koehn, Philipp. 2010. *Statistical Machine Translation*. Cambridge University Press.

Koehn, Philipp, Amittai Axelrod, Alexandra Birch, 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*, pages 68–75, Pittsburgh, PA.

Koehn, Philipp and Hieu Hoang. 2007. Factored Translation Models. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 868–876, Prague.

Koehn, Philipp, 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: Demonstrations*. pages 117–180, Prague.

Koehn, Philipp, Franz J. Och, and Daniel Marcu. 2003. Statistical Phrase-Based Translation. In *2003 Meeting of the North American Chapter of the Association for Computational Linguistics*, pages 127–133, Edmonton.

Kumar, Shankar and William J. Byrne. 2004. Minimum Bayes-Risk Decoding for Statistical Machine Translation. In *Human Language Technologies: The 2004 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 169–176, Boston, MA.

Mariño, José B., Rafael E. Banchs, Josep M. Crego, Adrià de Gispert, Patrik Lambert, José A. R. Fonollosa, and Marta R. Costa-jussà. 2006. N-gram-Based Machine Translation. *Computational Linguistics*, 32(4):527–549.

Moore, Robert and Chris Quirk. 2007. Faster Beam Search Decoding for Phrasal Statistical Machine Translation. In *Proceedings of the 11th Machine Translation Summit*, Copenhagen.

Niehues, Jan, Teresa Herrmann, Stephan Vogel, and Alex Waibel. 2011. Wider Context by Using Bilingual Language Models in Machine Translation. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 198–206, Edinburgh.

Och, Franz J. 2003. Minimum Error Rate Training in Statistical Machine Translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 160–167, Sapporo.

Och, Franz J. and Hermann Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1):19–51.

Och, Franz J. and Hermann Ney. 2004. The Alignment Template Approach to Statistical Machine Translation. *Computational Linguistics*, 30(1):417–449.

Ratnaparkhi, Adwait. 1998. *Maximum Entropy Models for Natural Language Ambiguity Resolution*. Ph.D. thesis, University of Pennsylvania, Philadelphia, PA.

Schmid, Helmut. 2000. Lopar: Design and implementation. Bericht des sonderforschungsbereiches "sprachtheoretische grundlagen für die computerlinguistik." Technical report, available at `www.cis.uni-muenchen.de/~schmid/papers/lopar.pdf`. Institute for Computational Linguistics, University of Stuttgart.

Stolcke, Andreas. 2002. SRILM - An Extensible Language Modeling Toolkit. In *International Conference on Spoken Language Processing*, Denver, CO.

Tillmann, Christoph and Tong Zhang. 2005. A Localized Prediction Model for Statistical Machine Translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 557–564, Ann Arbor, MI.

Vaswani, Ashish, Haitao Mi, Liang Huang, and David Chiang. 2011. Rule Markov Models for Fast Tree-to-String Translation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 856–864, Portland, OR.

Zaidan, Omar F. 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.

Zhang, Hui, Kristina Toutanova, Chris Quirk, and Jianfeng Gao. 2013. Beyond Left-to-Right: Multiple Decomposition Structures for SMT. In the *2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 12–21, Atlanta, GA.