Feature Decay Algorithms for Fast Deployment of Accurate Statistical Machine Translation Systems

Ergun Biçici
Centre for Next Generation Localisation,
Dublin City University, Dublin, Ireland.
ergun.bicici@computing.dcu.ie

Abstract

We use feature decay algorithms (FDA) for fast deployment of accurate statistical machine translation systems taking only about half a day for each translation direction. We develop parallel FDA for solving computational scalability problems caused by the abundance of training data for SMT models and language models and still achieve SMT performance that is on par with using all of the training data or better. Parallel FDA runs separate FDA models on randomized subsets of the training data and combines the instance selections later. Parallel FDA can also be used for selecting the LM corpus based on the training set selected by parallel FDA. The high quality of the selected training data allows us to obtain very accurate translation outputs close to the top performing SMT systems. The relevancy of the selected LM corpus can reach up to 86% reduction in the number of OOV tokens and up to 74% reduction in the perplexity. We perform SMT experiments in all language pairs in the WMT13 translation task and obtain SMT performance close to the top systems using significantly less resources for training and development.

1 Introduction

Statistical machine translation (SMT) is a data intensive problem. If you have the translations for the source sentences you are translating in your training set or even portions of it, then the translation task becomes easier. If some tokens are not found in your training data then you cannot translate them and if some translated word do not appear in your language model (LM) corpus, then it becomes harder for the SMT engine to find their correct position in the translation.

Current SMT systems also face problems caused by the proliferation of various parallel corpora available for building SMT systems. The training data for many of the language pairs in the translation task, part of the Workshop on Machine translation (WMT13) (Callison-Burch et al., 2013), have increased the size of the available parallel corpora for instance by web crawled corpora over the years. The increased size of the training material creates computational scalability problems when training SMT models and can increase the amount of noisy parallel sentences found. As the training set sizes increase, proper training set selection becomes more important.

At the same time, when we are going to translate just a couple of thousand sentences, possibly belonging to the same target domain, it does not make sense to invest resources for training SMT models over tens of millions of sentences or even more. SMT models like Moses already have filtering mechanisms to create smaller parts of the built models that are relevant to the test set.

In this paper, we develop parallel feature decay algorithms (FDA) for solving computational scalability problems caused by the abundance of training data for SMT models and LMs and still achieve SMT performance that is on par with using all of the training data or better. Parallel FDA runs separate FDA models on randomized subsets of the training data and combines the instance selections later. We perform SMT experiments in all language pairs of the WMT13 (Callison-Burch et al., 2013) and obtain SMT performance close to the baseline Moses (Koehn et al., 2007) system using less resources for training. With parallel FDA, we can solve not only the instance selection problem for training data but also instance selection for the LM training corpus, which allows us to train higher order n-gram language models and model the dependencies better.

Parallel FDA improves the scalability of FDA and allows rapid prototyping of SMT systems for a given target domain or task. Parallel FDA can be very useful for MT in target domains with limited resources or in disaster and crisis situations (Lewis et al., 2011) where parallel corpora can be gathered by crawling and selected by parallel FDA. Parallel FDA also improves the computational requirements of FDA by selecting from smaller corpora and distributing the work load. The high quality of the selected training data allows us to obtain very accurate translation outputs close to the top performing SMT systems. The relevancy of the LM corpus selected can reach up to 86% reduction in the number of OOV tokens and up to 74% reduction in the perplexity.

We organize our work as follows. We describe FDA and parallel FDA models in the next section. We also describe how we extend the FDA model for LM corpus selection. In section 3, we present our experimental results and in the last section, we summarize our contributions.

2 Feature Decay Algorithms for Instance Selection

In this section, we describe the FDA algorithm, the parallel FDA model, and how FDA training instance selection algorithms can be used also for instance selection for language model corpora.

2.1 Feature Decay Algorithm (FDA)

Feature decay algorithms (Biçici and Yuret, 2011a; Biçici, 2011) increase the diversity of the training set by decaying the weights of n-gram features that have already been included. FDAs try to maximize the coverage of the target language features for the test set. Translation performance can improve as we include multiple possible translations for a given word, which increases the diversity of the training set. A target language feature that does not appear in the selected training instances will be difficult to produce regardless of the decoding algorithm (impossible for unigram features). FDA tries to find as many training instances as possible to increase the chances of covering the correct target language feature by reducing the weight of the included features after selecting each training instance.

Algorithm 1 gives the pseudo-code for FDA. We improve FDA with improved scaling, where the score for each sentence is scaled proportional to the length of the sentence, which reduces the average length of the training instances.

Algorithm 1: The Feature Decay Algorithm

```
Input: Parallel training sentences $\mathcal{U}$, test set
       features $\mathcal{F}$, and desired number of
       training instances $N$.
Data: A priority queue $\mathcal{Q}$, sentence scores
      score, feature values fval.
Output: Subset of the parallel sentences to be
        used as the training data $\mathcal{L} \subseteq \mathcal{U}$.
1  foreach $f \in \mathcal{F}$ do
2      $\texttt{fval}(f) \leftarrow \texttt{init}(f,\mathcal{U})$
3  foreach $S \in \mathcal{U}$ do
4      $\texttt{score}(S) \leftarrow \frac{1}{|S|^s} \sum_{f \in \texttt{features}(S)} \texttt{fval}(f)$
5      $\texttt{enqueue}(\mathcal{Q}, S, \texttt{score}(S))$
6  while $|\mathcal{L}| < N$ do
7      $S \leftarrow \texttt{dequeue}(\mathcal{Q})$
8      $\texttt{score}(S) \leftarrow \frac{1}{|S|^s} \sum_{f \in \texttt{features}(S)} \texttt{fval}(f)$
9      if $\texttt{score}(S) \geq \texttt{topval}(\mathcal{Q})$ then
10         $\mathcal{L} \leftarrow \mathcal{L} \cup \{S\}$
11         foreach $f \in \texttt{features}(S)$ do
12             $\texttt{fval}(f) \leftarrow \texttt{decay}(f,\mathcal{U},\mathcal{L})$
13     else
14         $\texttt{enqueue}(\mathcal{Q}, S, \texttt{score}(S))$
```

The input to the algorithm consists of parallel training sentences, the number of desired training instances, and the source language features of the test set. The feature decay function (decay) is the most important part of the algorithm where feature weights are multiplied by $1/n$ where n is the count of the feature in the current training set. The initialization function (init) calculates the log of inverse document frequency (idf): $\texttt{init}(f,\mathcal{U}) = \log(|\mathcal{U}|/(1 + C(f,\mathcal{U})))$, where $|\mathcal{U}|$ is the sum of the number of features appearing in the training corpus and $C(f,\mathcal{U})$ is the number of times feature f appear in $\mathcal{U}$. Further experiments with the algorithm are given in (Biçici and Yuret, 2011a). We improve FDA with a scaling factor that prefers shorter sentences defined as: $|S|^s$, where s is the power of the source sentence length and we set it to 0.9 after optimizing it over the perplexity of the LM built over the selected corpus (further discussed in Section 2.3).

2.2 Parallel FDA Model

FDA model obtains a sorting over all of the available training corpus based on the weights of the

Algorithm 2: Parallel FDA

```
  Input: $\mathcal{U}$, $\mathcal{F}$, and $N$.
  Output: $\mathcal{L} \subseteq \mathcal{U}$.
1 $\mathcal{U} \leftarrow$ shuffle($\mathcal{U}$)
2 $\boldsymbol{\mathcal{U}}, M \leftarrow$ split($\mathcal{U}, N$)
3 $\boldsymbol{\mathcal{L}} \leftarrow \{\}$
4 $\boldsymbol{\mathcal{S}} \leftarrow \{\}$
5 foreach $\mathcal{U}_i \in \boldsymbol{\mathcal{U}}$ do
6     $\mathcal{L}_i, \mathcal{S}_i \leftarrow$ FDA($\mathcal{U}_i, \mathcal{F}, M$)
7     add($\boldsymbol{\mathcal{L}}, \mathcal{L}_i$)
8     add($\boldsymbol{\mathcal{S}}, \mathcal{S}_i$)
9 $\mathcal{L} \leftarrow$ merge($\boldsymbol{\mathcal{L}}, \boldsymbol{\mathcal{S}}$)
```

features found on the test set. Each selected training instance effects which feature weights will be decayed and therefore can result in a different ordering of the instances if previous instance selections are altered. This makes it difficult to parallelize the FDA algorithm fully. Parallel FDA model first shuffles the parallel training sentences, $\mathcal{U}$, and distributes them to multiple splits for running individual FDA models on them.

The input to parallel FDA also consists of parallel training sentences, the number of desired training instances, and the source language features of the test set. The first step shuffles the parallel training sentences and the next step splits into equal parts and outputs the split files and the adjusted number of instances to select from each, M. Since we split into equal parts, we select equal number of sentences, M, from each split. Then we run FDA on each file to obtain sorted files, $\boldsymbol{\mathcal{L}}$, together with their scores, $\boldsymbol{\mathcal{S}}$. merge combines k sorted lists into one sorted list in $O(Mk \log k)$ where Mk is the total number of elements in all of the input lists. [1] The obtained $\mathcal{L}$ is the new training set to be used for SMT experiments. We compared the target 2-gram feature coverage of the training sets obtained with FDA and parallel FDA and found that parallel FDA achieves close performance.

Parallel FDA improves the scalability of FDA and allows rapid prototyping of SMT systems for a given target domain or task. Parallel FDA also improves the computational requirements of FDA by selecting from smaller corpora and distributing the work load, which can be very useful for MT in disaster scenarios.

[1] (Cormen et al., 2009), question 6.5-9. Merging k sorted lists into one sorted list using a min-heap for k-way merging.

2.3 Instance Selection for the Language Model Corpus

The language model corpus is very important for improving the SMT performance since it helps finding the correct ordering among the translated tokens or phrases. Increased LM corpus size can increase the SMT performance where doubling the LM corpus can improve the BLEU (Papineni et al., 2002) by 0.5 (Koehn, 2006). However, although LM corpora resources are more abundant, training on large LM corpora also poses computational scalability problems and until 2012, LM corpora such as LDC Gigaword corpora were not fully utilized due to memory limitations of computers and even with large memory machines, the LM corpora is split into pieces, interpolated, and merged (Koehn and Haddow, 2012) or the LM order is decreased to use up to 4-grams (Markus et al., 2012) or low frequency n-gram counts are omitted and better smoothing techniques are developed (Yuret, 2008). Using only the given training data for building the LM is another option used for limiting the size of the corpus, which can also obtain the second best performance in Spanish-English translation task and in the top tier for German-English (Guzman et al., 2012; Callison-Burch et al., 2012). This can also indicate that prior knowledge of the test set domain and its similarity to the available parallel training data may be diminishing the gains in SMT performance through better language modeling or better domain adaptation.

For solving the computational scalability problems, there is a need for properly selecting LM training data as well. We select LM corpus with parallel FDA based on this observation:

> No word not appearing in the training set can appear in the translation.

It is impossible for an SMT system to translate a word unseen in the training corpus nor can it translate it with a word not found in the target side of the training set [2]. Thus we are only interested in correctly ordering the words appearing in the training corpus and collecting the sentences that contain them for building the LM. At the same time, we want to be able to model longer range dependencies more efficiently especially for morphologically rich languages (Yuret and Biçici,

[2] Unless the translation is a verbatim copy of the source.

2009). Therefore, a compact and more relevant LM corpus can be useful.

Selecting the LM corpus is harder. First of all, we know which words should appear in the LM corpus but we do not know which phrases should be there since the translation model may reorder the translated words, find different translations, and generate different phrases. Thus, we use 1-gram features for LM corpus selection. At the same time, in contrast with selecting instances for the training set, we are less motivated to increase the diversity since we want predictive power on the most commonly observed patterns. Thus, we do not initialize feature weights with the idf score and instead, we use the inverse of the idf score for initialization, which is giving more importance to frequently occurring words in the training set. This way of LM corpus selection also allows us to obtain a more controlled language and helps us create translation outputs within the scope of the training corpus and the closely related LM corpus.

We shuffle the LM corpus available before splitting and select from individual splits, to prevent extreme cases. We add the training set directly into the LM and also add the training set not selected into the pool of sentences that can be selected for the LM. The scaling parameter s is optimized over the perplexity of the training data with the LM built over the selected LM corpus.

3 Experiments

We experiment with all language pairs in both directions in the WMT13 translation task (Callison-Burch et al., 2013), which include English-German (en-de), English-Spanish (en-es), English-French (en-fr), English-Czech (en-cs), and English-Russian (en-ru). We develop translation models using the phrase-based Moses (Koehn et al., 2007) SMT system. We true-case all of the corpora, use 150-best lists during tuning, set the max-fertility of GIZA++ (Och and Ney, 2003) to a value between 8-10, use 70 word classes learned over 3 iterations with the mkcls tool during GIZA++ training, vary the LM order between 5 to 9 for all language pairs, and train the LM using SRILM (Stolcke, 2002). The development set contains 3000 sentences randomly sampled from among all of the development sentences provided.

Since we do not know the best training set size that will maximize the performance, we rely on previous SMT experiments (Biçici and Yuret, 2011a; Biçici and Yuret, 2011b) to select the proper training set size. We choose close to 15 million words and its corresponding number of sentences for each training corpus and 10 million sentences for each LM corpus not including the selected training set, which is added later. This corresponds to selecting roughly 15% of the training corpus for en-de and 35% for ru-en, and due to their larger size, 5% for en-es, 6% for cs-en, 2% for en-fr language pairs. The size of the LM corpus allows us to build higher order models. The statistics of the training data selected by the parallel FDA is given in Table 1. Note that the training set size for different translation directions differ slightly since we run a parallel FDA for each.

	cs / en	de / en	es / en	fr / en	ru / en
words (#M)	186 / 215	92 / 99	409 / 359	1010 / 886	41 / 44
sents (#K)	867	631	841	998	709
words (#M)	13 / 15	16 / 17	23 / 21	26 / 22	16 / 18

Table 1: Comparison of the training data available and the selected training set by parallel FDA for each language pair. The size of the parallel corpora is given in millions (M) of words or thousands (K) of sentences.

After selecting the training set, we select the LM corpora using the words in the target side of the training set as the features. For en, es, and fr, we have access to the LDC Gigaword corpora, from which we extract only the story type news and for en, we exclude the corpora from Xinhua News Agency (xin_eng). The size of the LM corpora from LDC and the monolingual LM corpora provided by WMT13 are given in Table 2. For all target languages, we select 10M sentences with parallel FDA from the LM corpora and the remaining training sentences and add the selected training data to obtain the LM corpus. Thus the size of the LM corpora is 10M plus the number of sentences in the training set as given in Table 1.

#M	cs	de	en	es	fr	ru
LDC	-	-	3402	949	773	-
Mono	388	842	1389	341	434	289

Table 2: The size of the LM corpora from LDC and the monolingual language model corpora provided in millions (M) of words.

With FDA, we can solve not only the instance selection problem for the training data but also the instance selection problem for the LM training corpus and achieve close target 2-gram cover-

	$S \rightarrow en$					$en \rightarrow T$				
	cs-en	de-en	es-en	fr-en	ru-en	en-cs	en-de	en-es	en-fr	en-ru
WMT13	.2620	.2680	.3060	.3150	.2430	.1860	.2030	.3040	.3060	.1880
BLEUc	.2430	.2414	.2909	.2539	.2226	.1708	.1792	.2799	.2379	.1732
BLEUc diff	.0190	.0266	.0151	.0611	.0204	.0152	.0238	.0241	.0681	.0148
LM order	7	9	7	9	6	5	5	5	7	5
BLEUc, n	.2407, 5	.2396, 5	.2886, 8	.2532, 6	.2215, 9	.1698, 9	.1784, 9	.2794, 9	.2374, 9	.1719, 9

Table 3: Best BLEUc results obtained on the translation task together with the LM order used when obtaining the result compared with the best constrained Moses results in WMT12 and WMT13. The last row compares the BLEUc result with respect to using a different LM order.

age using about 5% of the available training data and 5% of the available LM corpus for instance for en. A smaller LM training corpus also allows us to train higher order n-gram language models and model the dependencies better and achieve lower perplexity as given in Table 5.

3.1 WMT13 Translation Task Results

We run a number of SMT experiments for each language pair varying the LM order used and obtain different results and sorted these based on the tokenized BLEU performance, BLEUc. The best BLEUc results obtained on the translation task together with the LM order used when obtaining the results are given in Table 3. We also list the top results from WMT13 (Callison-Burch et al., 2013) [3], which use phrase-based Moses for comparison [4] and the BLEUc difference we obtain. For translation tasks with en as the target, higher order n-gram LM perform better whereas for translation tasks with en as the source, mostly 5-gram LM perform the best. We can obtain significant gains in BLEU (+0.0023) using higher order LMs.

For all translation tasks except fr-en and en-fr, we are able to obtain very close results to the top Moses system output (0.0148 to 0.0266 BLEUc difference). This shows that we can obtain very accurate translation outputs yet use only a small portion of the training corpus available, significantly reducing the time required for training, development, and deployment of an SMT system for a given translation task.

We are surprised by the lower performance in en-fr or fr-en translation tasks and the reason is, we believe, due to the inherent noise in the GigaFrEn training corpus [5]. FDA is an instance selection tool and it does not filter out target sentences that are noisy since FDA only looks at the source sentences when selecting training instance pairs. Noisy instances may be caused by a sentence alignment problem and one way to fix them is to measure the sentence alignment accuracy by using a similarity score over word distributions such as the Zipfian Word Vectors (Biçici, 2008). Since noisy parallel corpora can decrease the performance, we also experimented with discarding the GigaFrEn corpus in the experiments. However, this decreased the results by 0.0003 BLEU in contrast to 0.004-0.01 BLEU gains reported in (Koehn and Haddow, 2012). Also, note that the BLEU results we obtained are lower than in (Koehn and Haddow, 2012), which may be an indication that our training set size was small for this task.

3.2 Training Corpus Quality

We measure the quality of the training corpus by the coverage of the target 2-gram features of the test set, which is found to correlate well with the BLEU performance achievable (Biçici and Yuret, 2011a). Table 4 presents the source (scov) and target (tcov) 2-gram feature coverage of both the parallel training corpora (train) that we select from and the training sets obtained with parallel FDA. We show that we can obtain coverages close to using all of the available training corpora.

3.3 LM Corpus Quality

We compare the perplexity of the LM trained on all of the available training corpora for the de-en language pair versus the LM trained on the parallel FDA training corpus and the parallel FDA LM corpus. The number of OOV tokens become 2098, 2255, and 291 respectively for English and 2143, 2555, and 666 for German. To be able to compare the perplexities, we take the OOV tokens into consideration during calculations. Tokenized LM

[3]We use the results from matrix.statmt.org.

[4]Phrase-based Moses systems usually rank in the top 3.

[5]We even found control characters in the corpora.

		cs-en	de-en	es-en	fr-en	ru-en	en-cs	en-de	en-es	en-fr	en-ru
train	scov	.70	.74	.85	.83	.66	.82	.82	.84	.87	.78
	tcov	.82	.82	.84	.87	.78	.70	.74	.85	.83	.66
FDA	scov	.70	.74	.85	.82	.66	.82	.82	.84	.84	.78
	tcov	.74	.75	.77	.78	.75	.59	.67	.78	.76	.61

Table 4: Source (scov) and target (tcov) 2-gram feature coverage comparison of the training corpora (train) with the training sets obtained with parallel FDA (FDA).

corpus has 247M tokens for en and 218M tokens for de. We assume that each OOV word in *en* or *de* contributes $\log(1/218M)$ to the log probability, which we round to -19. We also present results for the case when we handle OOV words better with a cost of -11 each in Table 5.

Table 5 shows that we reduce the perplexity with a LM built on the training set selected with parallel FDA, which uses only 15% of the training data for de-en. More significantly, the LM build on the LM corpus selected by the parallel FDA is able to decrease both the number of OOV tokens and the perplexity and allows us to efficiently model higher order relationships as well. We reach up to 86% reduction in the number of OOV tokens and up to 74% reduction in the perplexity.

		log OOV = -19			log OOV = -11		
ppl		train	FDA	FDA LM	train	FDA	FDA LM
en	3	763	774	203	431	419	187
	4	728	754	192	412	409	178
	5	725	753	191	410	408	176
	6	724	753	190	409	408	176
	7	724	753	190	409	408	176
de	3	1255	1449	412	693	713	343
	4	1216	1428	398	671	703	331
	5	1211	1427	394	668	702	327
	6	1210	1427	393	668	702	326
	7	1210	1427	392	668	702	326

Table 5: Perplexity comparison of the LM built from the training corpus (train), parallel FDA selected training corpus (FDA), and the parallel FDA selected LM corpus (FDA LM).

3.4 Computational Costs

In this section, we quantify how fast the overall system runs for a given language pair. The instance selection times are dependent on the number of training sentences available for the language pair for training set selection and for the target language for LM corpus selection. We give the average number of minutes it takes for the parallel FDA to finish selection for each direction and for each target language in Table 6.

time (minutes)	en-fr	en-ru
Parallel FDA train	50	18
Parallel FDA LM	66	50

Table 6: The average time in the number of minutes for parallel FDA to select instances for the training set or for the LM corpus for language pairs en-fr and en-ru.

Once the training set and the LM corpus are ready, the training of the phrase-based SMT model Moses takes about 12 hours. Therefore, we are able to deploy an SMT system for the target translation task in about half a day and still obtain very accurate translation results.

4 Contributions

We develop parallel FDA for solving computational scalability problems caused by the abundance of training data for SMT models and LMs and still achieve SMT performance that is on par with the top performing SMT systems. The high quality of the selected training data and the LM corpus allows us to obtain very accurate translation outputs while the selected the LM corpus results in up to 86% reduction in the number of OOV tokens and up to 74% reduction in the perplexity and allows us to model higher order dependencies.

FDA and parallel FDA raise the bar of expectations from SMT translation outputs with highly accurate translations and lowering the bar to entry for SMT into new domains and tasks by allowing fast deployment of SMT systems in about half a day. Parallel FDA provides a new step towards rapid SMT system development in budgeted training scenarios and can be useful in developing machine translation systems in target domains with limited resources or in disaster and crisis situations where parallel corpora can be gathered by crawling and selected by parallel FDA. Parallel FDA is also allowing a shift from general purpose SMT systems towards task adaptive SMT solutions.

Acknowledgments

This work is supported in part by SFI (07/CE/I1142) as part of the Centre for Next Generation Localisation (www.cngl.ie) at Dublin City University and in part by the European Commission through the QTLaunchPad FP7 project (No: 296347). We also thank the SFI/HEA Irish Centre for High-End Computing (ICHEC), Koç University, and Deniz Yuret for the provision of computational facilities and support.

References

Ergun Biçici and Deniz Yuret. 2011a. Instance selection for machine translation using feature decay algorithms. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 272–283, Edinburgh, Scotland, July. Association for Computational Linguistics.

Ergun Biçici and Deniz Yuret. 2011b. RegMT system for machine translation, system combination, and evaluation. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 323–329, Edinburgh, Scotland, July. Association for Computational Linguistics.

Ergun Biçici. 2008. Context-based sentence alignment in parallel corpora. In *Proceedings of the 9th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing 2008), LNCS*, Haifa, Israel, February.

Ergun Biçici. 2011. *The Regression Model of Machine Translation*. Ph.D. thesis, Koç University. Supervisor: Deniz Yuret.

Chris Callison-Burch, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia. 2012. Findings of the 2012 workshop on statistical machine translation. In *Proc. of the Seventh Workshop on Statistical Machine Translation*, pages 10–51, Montréal, Canada, June. Association for Computational Linguistics.

Chris Callison-Burch, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia. 2013. Findings of the 2013 workshop on statistical machine translation. In *Proc. of the Eighth Workshop on Statistical Machine Translation*, pages 10–51. Association for Computational Linguistics, August.

Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. 2009. *Introduction to Algorithms (3. ed.)*. MIT Press.

Francisco Guzman, Preslav Nakov, Ahmed Thabet, and Stephan Vogel. 2012. Qcri at wmt12: Experiments in spanish-english and german-english machine translation of news text. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 298–303, Montréal, Canada, June. Association for Computational Linguistics.

Philipp Koehn and Barry Haddow. 2012. Towards effective use of training data in statistical machine translation. In *Proc. of the Seventh Workshop on Statistical Machine Translation*, pages 317–321, Montréal, Canada, June. Association for Computational Linguistics.

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

Philipp Koehn. 2006. Statistical machine translation: the basic, the novel, and the speculative. Tutorial at EACL 2006.

William Lewis, Robert Munro, and Stephan Vogel. 2011. Crisis mt: Developing a cookbook for mt in crisis situations. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 501–511, Edinburgh, Scotland, July. Association for Computational Linguistics.

Freitag Markus, Peitz Stephan, Huck Matthias, Ney Hermann, Niehues Jan, Herrmann Teresa, Waibel Alex, Hai-son Le, Lavergne Thomas, Allauzen Alexandre, Buschbeck Bianka, Crego Joseph Maria, and Senellart Jean. 2012. Joint wmt 2012 submission of the quaero project. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 322–329, Montréal, Canada, June. Association for Computational Linguistics.

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

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proc. of 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July. Association for Computational Linguistics.

Andreas Stolcke. 2002. Srilm - an extensible language modeling toolkit. In *Proc. Intl. Conf. on Spoken Language Processing*, pages 901–904.

Deniz Yuret and Ergun Biçici. 2009. Modeling morphologically rich languages using split words and unstructured dependencies. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 345–348, Suntec, Singapore, August. Association for Computational Linguistics.

Deniz Yuret. 2008. Smoothing a tera-word language model. In *Proceedings of ACL-08: HLT, Short Papers*, pages 141–144, Columbus, Ohio, June. Association for Computational Linguistics.