Extending Feature Decay Algorithms using Alignment Entropy

Alberto Poncelas, Andy Way, and Antonio Toral

ADAPT Centre, School of Computing,
Dublin City University, Dublin, Ireland
{firstname.lastname}@adaptcentre.ie

Abstract

In machine-learning applications, data selection is of crucial importance if good runtime performance is to be achieved. Feature Decay Algorithms (FDA) have demonstrated excellent performance in a number of tasks. While the decay function is at the heart of the success of FDA, its parameters are initialised with the same weights. In this paper, we investigate the effect on Machine Translation of assigning more appropriate weights to words using word-alignment entropy. In experiments on German to English, we show the effect of calculating these weights using two popular alignment methods, GIZA++ and FastAlign, using both automatic and human evaluations. We demonstrate that our novel FDA model is a promising research direction.

Keywords: Data Selection, Machine Translation, Mathematical Foundations

1 Introduction

Machine-learning approaches dominate in many fields. Many researchers have demonstrated that for a range of tasks, the more training data available, the better the system performance. In the field of Statistical Machine Translation (SMT), this was underlined in one of the first papers on phrase-based SMT [1], where improvements in BLEU score [2] of approximately 1 point were seen for a range of European language pairs each time the amount of training data from Europarl [3] was doubled.

However, others have shown that it is not always the case that having more data is necessarily better. [4] demonstrated that SMT performance decreases when additional training data is used to build the underlying models. Furthermore, a range of work beginning with [5] has shown that competitive SMT performance can still be achieved with fractions of the original training data if the characteristics of the test data can be examined and the optimal training data selected for the translation of that test set. The specific technique used is Feature Decay Algorithms (FDA), which have demonstrated excellent performance in a number of tasks by maximizing the diversity of the test set features while simultaneously increasing target coverage by using smaller yet more relevant amounts of training data. While the decay function is at the heart of the

success of FDA, its parameters are initialised with the same weights. In this paper, we investigate the effect on MT of assigning more appropriate weights to words using word-alignment entropy.

In SMT at least, it is clear that data selection is of crucial importance if we are to avoid the "garbage in, garbage out" syndrome. If optimal runtime performance is to be achieved, then the quality of the training data needs to be as good as it can be for the task at hand. Techniques have been developed in SMT which result in huge speed-ups in runtime performance. One such method is the FastAlign word-alignment model [6], which has been shown to deliver speedups of up to tenfold with no discernible drop-off in performance, compared to using GIZA++ [7], the most popular tool used in SMT for word alignment. As might be expected, these speedups have proved attractive to industry, and have been deployed in translation pipelines to good effect [8].

Nonetheless, MT engine training is a task that (generally) only needs to be done once, so in our view quality clearly trumps speed. Accordingly, in this paper we also set out to test whether there is any drop-off in performance by using FastAlign for the calculation of word-alignment entropy in FDA compared to using GIZA++. In experiments on German to English, we show the effect of calculating these weights using these two popular alignment methods, and examine the results using both automatic and human evaluations.

The remainder of this paper is organised as follows. In Section 2, we describe the related work on which our own research is based, with a special focus on FDA and word alignment. In Section 3, we detail our methodology which extends FDA using word-alignment entropy. Section 4 describes the experiments conducted, with the results discussed in Section 5. In Section 6, we conclude, and list a number of avenues for further work in this area.

2 Related Work

There are several methods for data selection [9]. Those most closely related to this work iteratively select one or more sentences from a candidate pool, updating at each step a set of sentences obtained in previous iterations (the "selected pool"). Those functions that select the next sentences depending on the selection pool are called "context-dependent" functions. The most related work to ours is the context-dependent function known as FDA [5; 10]. We provide an overview of FDA in Section 2.1, and outline the alignment models in Section 2.2.

2.1 Feature Decay Algorithms

FDA [5; 10] is a method that tries to maximize the variability of n-grams in the training set by decreasing their value as they are added to the selected pool. In order to do that, the n-grams in the test set (the document we want to translate) are extracted as features with an initial value. These features are then extracted from the training set. Each sentence has an importance score of being selected which is the normalized sum of the value of its features. At each step the sentence with the highest score is selected. Then the values of the features of the selected sentence are decreased as in (1):

$$decay(f) = init(f)\frac{d^{C_L(f)}}{(1 + C_L(f))^c} \quad (1)$$

L is the selected pool, d is the feature score polynomial decay factor, while c is the feature score exponential decay factor. $C_L(f)$ is the count of the feature f in L, which makes the most frequent features decay faster, thereby allowing an increase in variability of n-grams in the training data. The initialization function is defined in (2):

$$init(f) = log(|U|/C_U(f))^i |f|^l \quad (2)$$

where $|U|$ is the size of the training data, $C_U(f)$ is the count of the feature f in the training data and $|f|$ is the number of tokens of f.

2.2 Word-Alignment Models

IBM models [11] introduced the idea of adding alignment variables to the conditional probability $p(f_1...f_m|e_1...e_l)$ of a sentence $f_1 \dots f_m$ in the target language being the translation of a sentence $e_1 \dots e_l$ in the source. Concretely, (3) describes the conditional probability of IBM model 2:

$$p(f_1...f_m, a_1...a_m|e_1...e_l, m, l) \quad (3)$$

where the alignment variables $a_1 \dots a_m$ map each foreign word f_i with $i \in \{1 \dots m\}$ to an e_j with $j \in \{1 \dots l\}$.

GIZA++ [7] is the most widely used language-independent toolkit for calculating word alignments from bilingual corpora according to the IBM models. The FastAlign alignment model [6] is a variation of IBM model 2 that introduces a diagonal tension λ parameter that measures the overall correspondence of word order and an efficient re-estimation of the parameters that makes it around 10 times faster than GIZA++ while still obtaining comparable quality.

3 Applying Alignment Entropy in FDA

The FDA algorithm has already demonstrated its competitiveness by achieving excellent results in several Workshops on SMT from 2013–2015, on both MT and quality estimation tasks. Nonetheless, there appears to be scope to improve FDA still further by using word-alignment entropies. [5] show that the feature decay rate has a very strong effect on the final translation quality whereas the initial feature values, inclusion of higher order features, or sentence length normalizations do not.

3.1 Where should Alignment Entropy be applied?

In FDA, in formula (1), the parameters c (the feature score exponential decay factor) and d (the feature score polynomial decay factor) of the decay function are the same for every feature by default.

We propose instead that each feature should have different decay ratios. We contend that source n-grams that are regularly aligned to the same n-grams in the target language should have a higher decay ratio, since we require fewer

occurrences to 'guess' the translation. For instance, a word in German like "Deutschland" should have a more rapid decay as we would expect it to be aligned to the English word "Germany" in most cases when translating from German to English. In contrast, other German words like "zu" or "von" could be aligned to many different English words depending on the context of the sentence, and so the uncertainty of alignment is higher.

We want, therefore, to assign different values of d in (1) which by default is 0.5. In the implementation, the calculations of the decay function are made in the logarithmic scale, so if the value of d is in the range (0-1) higher values will result in slower decay.

According to [5], the choice of the *init* function does not affect the result as much as the decay function does, so we do not change it. Furthermore, in our experiments, we are varying only the value of d because in previous experiments variations in the value of the parameter c did not lead to as good results as those where the value of d was varied.

A method for measuring the difficulty of an n-gram to be aligned is using alignment entropy. Entropy measures uncertainty, as defined in (4):

$$entropy(x) = -\sum_i p(x_i) * log(p(x_i)) \tag{4}$$

3.2 Computing Alignment Entropy in FDA

In order to calculate the alignment entropy of a source word, we need to know the probability of its being aligned to words in the target language. Using FastAlign or GIZA++, it is possible to obtain the alignment probabilities of unigrams, which can be used to calculate the translation entropies. In this paper the experiments have been conducted calculating only the unigram entropies; if this proves to be a promising direction, we can always extend this method to higher-order n-grams. Furthermore, introducing alignment entropy to n-grams one order at a time will help us understand its benefits; if we were to perform this technique on (say) unigrams to 5-grams, it could not be guaranteed that we would understand exactly where the benefits of such an approach were to be attributed.

Let A_s be the set of words that are potential translations of the source word s, and $p(s,t)$ be the probability of s being aligned with the word t. Accordingly, the new decay ratio d will be given by the decay score computed in (5):

$$score(s) = \frac{\sum_{t \in A_s} p(s,t) * log(p(s,t))}{log(|A_s|)} \tag{5}$$

In order to have the decay score in the (0-1) range, we normalize the entropy of alignment of each word by dividing by $log(|A_s|)$, the maximum possible entropy.

For unfound words (i.e. whose alignment probability cannot be retrieved via FastAlign or GIZA++), we cannot calculate their alignment entropy in (5), so we assign them the average alignment entropy value of the rest of the (found) words.

In what follows, we use $score(w)$ to indicate the alignment entropy (or "decay score") of a word w. This is the value that will be used as the d parameter in the decay function in (1).

4 Experiments

Our experiments have two goals: (i) to explore and compare the performance of GIZA++ and FastAlign when using their alignment probabilities for calculating the decay scores used in FDA; and (ii) to improve the results obtained by the default FDA using unigrams as features. In order to explore both objectives, we designed three experiments:

- Baseline: data selection performed via FDA, using default decay scores.
- FastAlign experiment: data selection performed via FDA using the probabilities obtained with FastAlign to score the words.
- GIZA++ experiment: data selection performed via FDA using the probabilities obtained with GIZA++ to score the words.

FDA selects unigrams, bigrams and trigrams as features by default. However, given that in this work we begin calculating decay scores only for unigrams, we decided to perform the three experiments using only unigram features.

We train SMT systems on the selected data with the Moses toolkit [12] with default settings and using GIZA++ for word alignment. We also perform four tuning executions of each experiment using MERT [13], so that the reported scores are based on the average of the runs, and significance tests are more robust [14].

4.1 Data Sets

Based on the work described in [10] for WMT-15, we perform a similar experiment. However, our approach has significant differences since we use unigrams as a feature in this work. The data sets used in the experiments are as follows: (i) *Test data*: The test document provided in the WMT 2015 German-to-English translation task; (ii) *Training data*: The training data provided in the WMT 2015 translation task setting a maximum sentence length of 126 words; (iii) *Selected data*: We select 66.4 million words in total (source- and target-language sides); (iv) *Language Model*: 3-gram and 8-gram Language Models (LMs) built using the target-language side of the selected data via the KenLM toolkit [15]; (v) *Tuning data*: 5K randomly sampled sentences from development sets provided in the WMT Translation Tasks from the years 2010 to 2014.

5 Results

After obtaining the results, we conduct a comparison of the performance of the two decay score models, and also compare both experiments with the baseline.

5.1 Comparison of FastAlign And GIZA++

5.1.1 The Effect of Found vs. Unfound Words

In Table 1, we present a summary of the scores obtained by using GIZA++ and FastAlign. We observe that we obtain higher decay scores in both experiments compared to the default 0.5 value of the baseline system. This demonstrates that

Table 1: Statistics of the scores obtained using GIZA++ and FastAlign, with a comparison with the baseline system. Found-words are those words for which their alignment probability has been obtained. Mean and stdev indicate the mean and standard deviation of the scores of all the words in the test set.

	baseline	**FastAlign**	**GIZA++**
found-words	–	54.4%	87.6%
mean	0.5	0.9401	0.7198
stdev	0.0	0.0510	0.2439

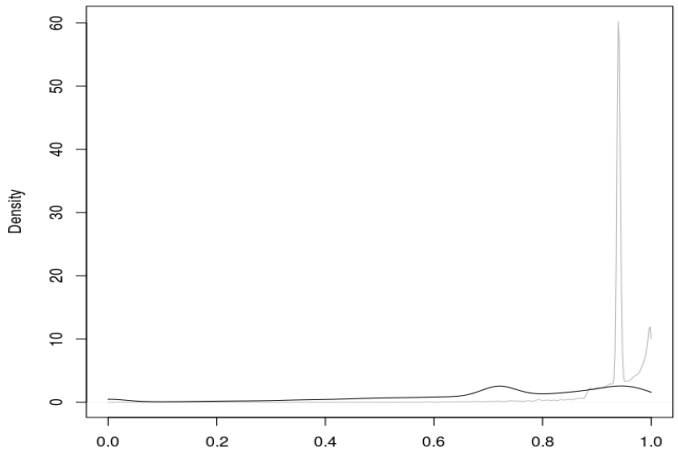

Figure 1: Density plot of the decay scores of FastAlign (grey) and GIZA++ (black) experiments.

the FDA algorithm may indeed be very sensitive to changes in the parameter d in (1), as we suspected.

For the purpose of readability, we also present the numbers in Table 1 in graphical format in Figure 1. There are essentially two observations: (i) for FastAlign, there are two spikes in the plot: the high one around the mean of 0.94 (because of the high number of unfound words), and a smaller one around 1.0, indicating FastAlign's tendency to give high scores; and (ii) for GIZA++, there is a small bump around the mean (0.71), but the decay scores are almost equally distributed.

Note also that the percentage of words found by GIZA++ is higher than the percentage of FastAlign: only 12.4% of the words are unfound by GIZA++, whereas 45.6% are unfound by FastAlign, almost four times as many.

In addition, the scores obtained using GIZA++ have a higher standard deviation. Recall that the decay score of the unfound words was assigned as the average of the decay score of the found words. Since the FastAlign experiment shows a higher percentage of unfound words, the standard deviation is lower. In Figure 1, we can see the effect of this, with most FastAlign words being clustered around the mean. Even though the means of all three systems are quite different, this very low standard deviation for FastAlign is noteworthy.

Table 2: Results of the average of the scores after 4 tuning executions for the baseline, FastAlign and GIZA++ experiments with LM order 3 and 8. The results in bold for GIZA++ indicate a statistically significant improvement (at level p=0.01) compared to FastAlign.

	baseline		**FastAlign**		**GIZA++**	
	LM=3	LM=8	LM=3	LM=8	LM=3	LM=8
BLEU	0.2291	0.2299	0.2237	0.2232	0.2282	**0.2279**
NIST	6.9475	6.9667	6.8911	6.8728	6.9327	6.9496
TER	0.5970	0.5957	0.5988	0.6001	0.5984	0.5973
METEOR	0.2833	0.2840	0.2818	0.2811	**0.2827**	**0.2836**
CHRF3	50.06	50.09	49.54	49.46	49.83	50.02
CHRF1	50.66	50.76	50.39	50.29	50.59	50.65

5.1.2 Implications for MT Performance

We also present the performance of the MT systems after tuning in Table 2 using a range of different evaluation metrics: BLEU [2], NIST [16], TER [17], METEOR [18] and CHRF [19]. These scores give an estimation of the quality of the translated output compared to a translated reference. The higher the score, the better the translation is estimated to be, except for TER, which being an error measure, indicates better translation output by lower scores.

We also provide comparative results with two different LMs: one with n-gram order 3 and the other with n-gram order 8.

As we can see, the entropies calculated with the probabilities of GIZA++ outperform those of FastAlign, for all metrics. The improvements over FastAlign are statistically significant for METEOR, as calculated with Bootstrap Resampling [20].[1] The influence of the larger n-gram order can be seen to good effect, too, with better results occurring with the larger LM for both the baseline and GIZA++ experiments, but interestingly not for FastAlign. Note too that the BLEU score for GIZA++ with the 8-gram LM – despite being statistically significantly better than the FastAlign score – is a little lower than with the trigram LM.

5.2 Comparison with the Baseline

5.2.1 Automatic Evaluation

As we saw in the previous section, the results computing word-alignment entropy with GIZA++ surpass those when FastAlign was used. However, it can be seen from Table 2 that the GIZA++ system *never* improves over the baseline engine.[2] However, none of the results of the baseline system are statistically significantly better than those of the GIZA++ engine.

As expected, results after tuning result in better performance; BLEU score improves by about 2%. However, the TER scores are worse after tuning for all

[1] Note that we were unable to calculate statistical significance for the CHRF metric. Note too that prior to tuning, statistically significant improvements were seen for GIZA++ over FastAlign for BLEU, NIST, TER and METEOR.

[2] In experiments before tuning (excluded here for reasons of space), the METEOR and CHRF scores of the output of the system executed with GIZA++ did outperform the baseline system before tuning.

three systems. The system parameters were optimized using BLEU score. Note that [21] observed that performing tuning using a particular metric may not lead to optimal scores on the test set for that metric, so something similar may be going on here.

In addition, recall that the results in Table 2 report the mean of the four tuning executions. It is instructive to investigate the difference in scores obtained from each of these runs, as seen in Table 3. It can be seen very clearly that for most metrics, the GIZA++ system can outperform the baseline; for METEOR, results are better in two of the four runs.[3] Of course, while we would not selectively pick the set-up with the best run for our purposes, we nonetheless take some encouragement from these results, as they demonstrate that our method does have the promise to outperform the baseline set-up.

5.2.2 Human Evaluation

As the results from the automatic evaluation were somewhat mixed, we decided to conduct a human evaluation of the outputs of the system with the trigram LM. Automatic scores offer some insight into system performance, but sometimes good output is penalised by the automatic metrics owing to the output being significantly different from the 'gold standard' reference translation.

Table 4 provides some instances where the GIZA++ outputs are considered worse by more than 0.3 BLEU points than the respective outputs of the baseline. However, in the first sentence, we see that the GIZA++ system produces *difficult* compared to the baseline's *hard*, which is perfectly acceptable. Furthermore, we argue that the way the GIZA++ translation ends is better even than the reference translation.

In the second example, the GIZA++ engine outputs *would like* instead of *want*, which again is a perfectly acceptable translation. In the third case, the system output by GIZA++ is an acceptable English sentence, unlike the string produced by the baseline, which is nonetheless a better match of the reference supplied. Note that all these examples contain words with similar semantics to the reference, which may explain the higher METEOR scores obtained by the GIZA++ system compared to the other metrics (cf. footnote 3).

By contrast, Table 5 provides two examples where the GIZA++ sentences are adjudged by BLEU to be better than the baseline outputs by more than 0.3 points. In the first example, the GIZA++ output mirrors the reference exactly, while the baseline output suffers from poor word order. The second example improves over the baseline in terms of pronominal ellipsis, pronoun selection and correctly inserting the adverb *there*.

Finally, we performed a ranking experiment of the outputs of the baseline and the GIZA++ systems on a random sample of 100 outputs. We found that 48% of the sentences were similar in quality, in 24% of the cases the baseline was better and in 28% GIZA++ was better.

[3] Why is it the case that better scores are more likely with the METEOR evaluation metric? This measure evaluates a hypothesis against a reference calculating sentence-level similarity scores. In so doing it searches for all the possible matches of the words between the two sentences. The words can match (i) if they are the same, (ii) if they have the same stem, (iii) if they are synonyms, or (iv) if they are found as a match in a paraphrase table. Therefore, this metric takes into consideration not only the n-grams, but also the semantic of the words. As the human evaluation shows, many semantically equivalent translations are output by our GIZA++ system, which are penalised by most automatic metrics, but not by METEOR.

Table 3: Results of the average of the scores as well as those of the 4 tuning executions for the baseline and GIZA++ experiments with LM of order 3 and 8. The results in bold show improvements over the baseline for individual runs.

LM 3		
	mean	**scores for 4 tuning runs**
baseline		
BLEU	0.2291	0.2291, 0.2291, 0.2289, 0.2291
NIST	6.9475	6.9441, 6.9441, 6.9575, 6.9441
TER	0.5970	0.5968, 0.5968, 0.5976, 0.5968
METEOR	0.2833	0.2832, 0.2832, 0.2833, 0.2832
CHRF3	50.064	50.133, 50.133, 49.858, 50.133
CHRF1	50.662	50.668, 50.668, 50.644, 50.668
GIZA++		
BLEU	0.2282	**0.2321**, 0.2269, 0.225, 0.2287
NIST	6.9327	**7.003**, 6.9297, 6.8536, 6.9443
TER	0.5984	**0.5935**, 0.5991, 0.6037, 0.5974
METEOR	0.2827	**0.2848**, 0.2824, 0.2793, **0.2844**
CHRF3	49.827	50.019, 49.737, 49.352, **50.200**
CHRF1	50.591	50.928, 50.528, 50.22, **50.690**

LM 8		
	mean	**scores for 4 tuning runs**
baseline		
BLEU	0.2299	0.2306, 0.2308, 0.2290, 0.2291
NIST	6.9667	6.9799, 6.9863, 6.9565, 6.9441
TER	0.5957	0.5944, 0.5929, 0.5988, 0.5968
METEOR	0.2840	0.2845, 0.2846, 0.2838, 0.2832
CHRF3	50.089	50.005, 50.078, 50.140, 50.133
CHRF1	50.759	50.815, 50.863, 50.687, 50.668
GIZA++		
BLEU	0.2279	0.2304, 0.2223, 0.2305, 0.2283
NIST	6.9496	6.9850, 6.8628, **6.9901**, 6.9604
TER	0.5973	0.5934, 0.6047, 0.5937, 0.5974
METEOR	0.2836	0.2843, 0.2807,**0.2854** , 0.2841
CHRF3	50.024	49.976, 49.846, **50.293**, 49.979
CHRF1	50.649	50.840, 50.146, **50.903**, 50.706

Table 4: Example translations obtained from the baseline and GIZA++ systems (*before tuning*). In this table, the BLEU scores of the baseline translations exceed those of the GIZA++ system by 0.3 points or more.

Reference	**baseline**	**GIZA++**
It is **hard** to accept that life goes on , even if you do not want it .	It is **hard** to accept that life goes on , even if you do not want .	It is **difficult** to accept that life goes on , even if we do not want to .
I **want** to help so much .	I **want** to help .	I **would like** to help .
No compensation **is paid** .	There will be no compensation **is paid** .	There will be no compensation **being paid** .

Table 5: Comparing the outputs of the best baseline and GIZA++ systems (*after tuning*). The GIZA++ sentences are adjudged by BLEU to be better than the baseline outputs by more than 0.3 points.

Reference	**baseline**	**GIZA++**
But in the end they all die .	But in the end will die them all .	But in the end they all die .
We 'll go on to Richmond and hope we can do better there .	We drive to Richmond and we hope that it can do better .	We drive to Richmond and hope that we can do better there .

5.2.3 Data Set Analysis

In the GIZA++ experiment we obtain fewer occurrences of proper names in the training set, e.g. the baseline data contains the word *Sydney* 600 times, but this word occurs only 333 times in the GIZA++ experiment. This is to be expected as such words typically have lower translation entropies. In contrast, we obtain more occurrences of other words, e.g. *schwer* ('difficult/heavy') occurs 5 times with GIZA++ compared to 4 in the baseline, while *gehen* ('to go') occurs 21 times with GIZA++ compared to 17 in the baseline. This increases the chances of obtaining synonyms, of course, which might explain the results in Table 4 and Table 5.

Note too that there are more Out-of-Vocabulary items in the training set of the GIZA++ experiment (4284) compared to the baseline training set (4175). In Figure 1, we observed that the distribution of the entropies is highly skewed, which causes a large amount of words to have a slow decay. This can be disadvantageous, as too many occurrences of a particular set of the vocabulary might be required before arriving at the threshold where other words can be selected.

6 Conclusions and Future Work

This work has an ambitious goal, namely to try to improve the performance of FDA, a method which has obtained a number of first-place results on a range of tasks at WMT evaluations over several years.

We observed that certain decay parameters crucial to the excellent showing of FDA in these competitions received default values. We conducted a range of experiments to see the effect of replacing these default values with a word-specific alignment-entropy score. We demonstrated that alignment entropies computed with FastAlign led to inferior performance with respect to those calculated via GIZA++. Any speed-quality trade-off by using FastAlign appears detrimental to this task.

Nonetheless, it proved difficult to outperform the baseline SMT scores. However, inspecting the individual MERT runs showed that the GIZA++ system had the potential to outperform the baseline. Furthermore, human evaluations demonstrated (i) that GIZA++ outputs were being unreasonably penalised in the automatic evaluations, and (ii) that GIZA++ outputs could be generated with clearly better quality than those of the baseline.

Accordingly, we deem these experimental findings to be promising enough to warrant an extension to higher-order n-grams. We also intend to conduct

experiments on different language pairs and data sets. Finally, we propose to discover whether combining the outputs of the default FDA and our new model may improve the overall results.

Acknowledgements

This research is supported by the ADAPT Centre for Digital Content Technology, funded under the SFI Research Centres Programme (Grant 13/RC/2106) and co-funded under the European Regional Development Fund, and the European Union Seventh Framework Programme FP7/2007-2013 under grant agreement PIAP-GA-2012-324414 (AbuMaTran).

References

[1] P. Koehn, F. Och, and D. Marcu, "Statistical phrase-based translation," in *Proceedings of Conference combining Human Language Technology conference series and the North American Chapter of the Association for Computational Linguistics conference series*, (Edmonton, Canada), pp. 48–54, 2003.

[2] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: a method for automatic evaluation of machine translation," in *Proceedings of the 40th ACL*, (Philadelphia, PA, USA), pp. 311–318, 2002.

[3] P. Koehn, "Europarl: a parallel corpus for statistical machine translation," in *MT Summit X Conference Proceedings: the Tenth Machine Translation Summit*, (Phuket, Thailand), pp. 79–86, 2005.

[4] S. Ozdowska and A. Way, "Optimal Bilingual Data for French-English PB-SMT," in *Proceedings of the 13th Annual Meeting of the European Association for Machine Translation*, (Barcelona, Spain), pp. 96–103, 2009.

[5] E. Biçici and D. Yuret, "Instance selection for machine translation using feature decay algorithms," in *Proceedings of the Sixth Workshop on Statistical Machine Translation*, (Edinburgh, Scotland), pp. 272–283, 2011.

[6] C. Dyer, V. Chahuneau, and N. Smith, "A simple, fast, and effective reparameterization of IBM model 2," in *Proceedings of Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings*, (Atlanta, Georgia, USA), pp. 644–648, 2013.

[7] F. Och and H. Ney, "A systematic comparison of various statistical alignment models," *Computational Linguistics*, vol. 29, no. 1, pp. 19–51, 2003.

[8] D. Shterionov, J. Du, M. A. Palminteri, L. Casanellas, T. O'Dowd, and A. Way, "Improving KantanMT training efficiency with FastAlign," in *Proceedings of AMTA 2016*, (Austin, TX), pp. 222–231, 2016.

[9] S. Eetemadi, W. Lewis, K. Toutanova, and H. Radha, "Survey of data-selection methods in statistical machine translation," *Machine Translation*, vol. 29, no. 3-4, pp. 189–223, 2015.

[10] E. Biçici, Q. Liu, and A. Way, "ParFDA for fast deployment of accurate statistical machine translation systems, benchmarks, and statistics," in *Proceedings of the Tenth Workshop on Statistical Machine Translation*, (Lisbon, Portugal), pp. 74–78, 2015.

[11] P. F. Brown, S. A. Della Pietra, V. J. Della Pietra, and R. L. Mercer, "The mathematics of statistical machine translation: parameter estimation," *Computational Linguistics*, vol. 19, pp. 263–311, 1993.

[12] P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin, and E. Herbst, "Moses: Open source toolkit for SMT," in *Proceedings of 45th annual meeting of the ACL on interactive poster & demonstration sessions*, (Prague, Czech Republic), pp. 177–180, 2007.

[13] F. Och, "Minimum error rate training in statistical machine translation," in *ACL-2003: 41st Annual Meeting of the Association for Computational Linguistics, Proceedings*, (Sapporo, Japan), pp. 160–167, 2003.

[14] J. H. Clark, C. Dyer, A. Lavie, and N. A. Smith, "Better hypothesis testing for statistical machine translation: Controlling for optimizer instability," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, (Portland, Oregon), p. 176–181, 2011.

[15] K. Heafield, "KenLM: Faster and smaller language model queries," in *Proceedings of the Sixth Workshop on Statistical Machine Translation*, (Edinburgh, Scotland), pp. 187–197, 2011.

[16] G. Doddington, "Automatic evaluation of machine translation quality using n-gram co-occurrence statistics," in *Proceedings of the second international conference on Human Language Technology Research*, (San Diego, CA), pp. 138–145, 2002.

[17] M. Snover, B. Dorr, R. Schwartz, L. Micciulla, and J. Makhoul, "A study of translation edit rate with targeted human annotation," in *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas*, (Cambridge, Massachusetts, USA), pp. 223–231, 2006.

[18] S. Banerjee and A. Lavie, "Meteor: An automatic metric for MT evaluation with improved correlation with human judgments," in *Proceedings of the ACL workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, (Ann Arbor, Michigan), pp. 65–72, 2005.

[19] M. Popovic, "chrF: character n-gram F-score for automatic MT evaluation," in *Proceedings of the Tenth Workshop on Statistical Machine Translation*, (Lisbon, Portugal), pp. 392–395, 2015.

[20] P. Koehn, "Statistical significance tests for machine translation evaluation," in *Proceedings of EMNLP 2004*, (Barcelona, Spain), pp. 388–395, 2004.

[21] Y. He and A. Way, "Improving the objective function in minimum error rate training," in *Proceedings of the Twelfth Machine Translation Summit*, (Ottawa, Canada), pp. 238–245, 2009.