Dramatically Reducing Training Data Size Through Vocabulary Saturation

William D. Lewis
Microsoft Research
One Microsoft Way
Redmond, WA 98052
wilewis@microsoft.com

Sauleh Eetemadi
Microsoft Research
One Microsoft Way, Redmond, WA 98052
Michigan State University, East Lansing, MI 48824
saulehe@microsoft.com

Abstract

Our field has seen significant improvements in the quality of machine translation systems over the past several years. The single biggest factor in this improvement has been the accumulation of ever larger stores of data. However, we now find ourselves the victims of our own success, in that it has become increasingly difficult to train on such large sets of data, due to limitations in memory, processing power, and ultimately, speed (i.e., data to models takes an inordinate amount of time). Some teams have dealt with this by focusing on data cleaning to arrive at smaller data sets (Denkowski et al., 2012a; Rarrick et al., 2011), "domain adaptation" to arrive at data more suited to the task at hand (Moore and Lewis, 2010; Axelrod et al., 2011), or by specifically focusing on data reduction by keeping only as much data as is needed for building models *e.g.*, (Eck et al., 2005). This paper focuses on techniques related to the latter efforts. We have developed a very simple n-gram counting method that reduces the size of data sets dramatically, as much as 90%, and is applicable independent of specific dev and test data. At the same time it reduces model sizes, improves training times, and, because it attempts to preserve contexts for all n-grams in a corpus, the cost in quality is minimal (as measured by BLEU). Further, unlike other methods created specifically for data reduction that have similar effects on the data, our method scales to very large data, up to tens to hundreds of millions of parallel sentences.

1 Introduction

The push to build higher and higher quality Statistical Machine Translation systems has led the efforts to collect more and more data. The English-French (nearly) Gigaword Parallel Corpus (Callison-Burch et al., 2009), which we will refer to henceforth as EnFrGW, is the result of one such effort. The EnFrGW is a publicly available corpus scraped from Canadian, European and international Web sites, consisting of over 22.5M parallel English-French sentences. This corpus has been used regularly in the WMT competition since 2009.

As the size of data increases, BLEU scores increase, but the increase in BLEU is not linear in relation to data size. The relationship between data size and BLEU flattens fairly quickly, as demonstrated in Figure 1. Here we see that BLEU scores increase rapidly with small amounts of data, but they taper off and flatten at much larger amounts. Clearly, as we add more data, the value of the new data diminishes with each increase, until very little value is achieved through the addition of each new sentence. However, given that this figure represents samples from EnFrGW, can we be more efficient in the samples we take? Can we achieve near equivalent BLEU scores on much smaller amounts of data drawn from the same source, most especially better than what we can achieve through random sampling?

The focus of this work is three-fold. First, we seek to devise a method to reduce the size of training data, which can be run independently of particular dev and test data, so as to maintain the independence of the data, since we are not interested here in domain adaptation or selective tuning. Second, we desire an algorithm that is (mostly) quality preserving, as measured by BLEU, OOV rates, and human eval, ultimately resulting in decreased training times and reduced model sizes. Reduced

Proceedings of the Eighth Workshop on Statistical Machine Translation, pages 281–291,
Sofia, Bulgaria, August 8-9, 2013

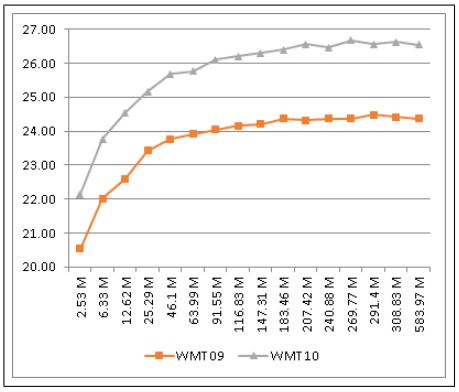


Figure 1: BLEU score increase as more data is added (in millions of words), random samples from EnFrGW

training times provide for greater iterative capacity, since we can make more rapid algorithmic improvements and do more experimentation on smaller data than we can on much larger data. Since we operate in a production environment, deploying smaller models is also desirable. Third, we require a method that scales to very large data. We show in the sections below the application of an algorithm at various settings to the 22.5M sentence EnFrGW corpus. Although large, 22.5M sentences does not represent the full total of the English-French data on the Web. We require an algorithm that can apply to even larger samples of data, on the order of tens to hundreds of millions of sentences.

2 Related Work

In statistical machine translation, selection, preparation and processing of parallel training data is often done to serve one of the following scenarios:

- Low Resource Languages: In languages with low parallel data availability, a subset of a monolingual corpus is selected for human translation ((Ananthakrishnan et al., 2010), (Eck et al., 2005) and (Haffari et al., 2009)).
- Mobile device deployment: For many languages, translation model sizes built on all available parallel data are too large to be hosted on mobile devices. In addition to translation model pruning, a common solution is selecting a subset of the data to be trained on ((Ananthakrishnan et al., 2010) and (Yasuda et al., 2008)).
- Quick turn-around time during development: A common motivation for training on a subset of a parallel corpus is to reduce training time during the development cycle of a statistical machine translation system ((Lin and Bilmes, 2011) and (Chao and Li, 2011a)).
- Noise reduction: Simple noise reduction techniques like sentence length and alpha numeric ratio are often used in data preparation. However, more sophisticated techniques have been developed to filter out noise from parallel data ((Denkowski et al., 2012a) and (Taghipour et al., 2010)).
- Domain Adaptation: Recently there has been significant interest in domain adaptation for statistical machine translation. One of the approaches to domain adaptation is selecting a subset of a data that is closer to the target domain ((Moore and Lewis, 2010), (Axelrod et al., 2011)).
- Improve translation quality: An interesting area of research is selecting a subset of the training data that is more suitable for statistical machine translation learning ((Okita, 2009)).

In comparison, the goal of this work is to efficiently reduce very large parallel data sets (in excess of tens of billions of tokens) to a desired size in a reasonable amount of time. In the related work referenced above two primary methods have been used.

1. Maximizing n-gram coverage with minimal data.
2. Filtering out noisy data based on sentence-pair based features.

One of the earliest and most cited works using the first method is (Eck et al., 2005). In this work, a greedy algorithm is developed to select a subset of the entire corpus that covers most n-grams with minimum number of words. In a later work by the same author, the algorithm was modified to give higher weight to more frequent words. Although this is a greedy algorithm and does not provide the optimum solution, its complexity is quadratic in the number of sentences. Hence it is not practical to run this algorithm over very large data sets.

Recently (Ananthakrishnan et al., 2010) introduced a new algorithm that is an improvement

over (Eck et al., 2005). In this work discriminative training is used to train a maximum entropy pairwise comparator with n-gram based features. The pair-wise comparator is used to select the highest scoring sentence followed by discounting features used for the sentence, which are drawn from the global pool of features. The complexity of this algorithm after training the pairwise comparator is $O\left(N \times K \times \log(F)\right)$ where N is the number of sentences in the entire corpus, K is the number of sentences to be selected and F is the size of the feature space. Although this method works well for a constant K, its complexity is quadratic when K is a fraction of N. This method is reported to improve the BLEU score close to 1% over the work done by (Eck et al., 2005).

(Denkowski et al., 2012a) have developed relatively scalable algorithms that fit in the second category above. This algorithm automatically filters out noisy data primarily based on the following feature functions: normalized source and target language model scores, word alignment scores and fraction of aligned words. Sentences that don't score above a certain threshold (mean minus one or two standard deviations) for all their features are filtered out. In a similar work, (Taghipour et al., 2010) use an approach where they incorporate similar features based on translation table entries, word alignment models, source and target language models and length to build a binary classifier that filters out noisy data.

Our work incorporates both methods listed above in a scalable fashion where it selects a subset of the data that is less noisy with a reasonable n-gram representation of the superset parallel corpus. To put the scalability of our work in perspective we compiled Table 1, which shows the maximum size of the data sets reported in each of the relevant papers on the topic. Despite the public availability of parallel corpora in excess of tens of millions of sentence pairs, none of the related works, using the first method above, exceed couple of millions of sentences pairs. This demonstrates the importance of developing a scalable algorithm when addressing the data selection problem.

The careful reader may observe that an alternate strategy for reducing model sizes (*e.g.*, useful for the Mobile scenario noted above, but also in any scenario where space concerns are an issue), would be to reduce phrase table size rather than reduce training data size. A good example of work in this space is shown in (Johnson et al., 2007), who describe a method for phrase table reduction, sometimes substantial (>90%), with no impact on the resulting BLEU scores. The principal of our work versus theirs is where the data reductions occur: before or after training. The primary benefit of manipulating the training data directly is the impact on training performance. Further, given the increasing sizes of training data, it has become more difficult and more time consuming to train on large data, and in the case of very large data (say tens to hundreds of millions of sentence pairs), it may not even be possible to train models at all. Reduced training data sizes increases iterative capacity, and is possible in cases where phrase table reduction may not be (*i.e.*, with very big data).

Reference	Total Sentences
(Ananthakrishnan et al., 2010)	253K
(Eck et al., 2005)	123K
(Haffari et al., 2009)	1.8M[1]
(Lin and Bilmes, 2011)	1.2M[2]
(Chao and Li, 2011b)	2.3M

Table 1: Data Sizes for Related Systems

3 Vocabulary Saturation Filter (VSF)

The effects of more data on improving BLEU scores is clearly discernible from Figure 1: as more data is added, BLEU scores increase. However, the relationship between quantity of data and BLEU is not linear, such that the effects of more data diminishes with each increase in data size, effectively approaching some asymptote. One might say that the vocabulary of the phrase mappings derived from model training "saturate" as data size increases, since less and less novel information can be derived from each succeeding sentence of data added to training. It is this observation that led us to develop the Vocabulary Saturation Filter (VSF).

VSF makes the following very simple assumption: for any given vocabulary item *v* there is some point where the contexts for *v*—that is, the n-gram

[1]Sentence count was not reported. We estimated it based on 18M tokens.

[2]This is a very interesting work, but is only done for selecting speech data. The total number of sentences is not reported. We given a high-end estimate based on 128K selected tokens.

sequences that contain *v*—approach some level of saturation, such that each succeeding sentence containing *v* contributes few or no additional contexts, and thus has little impact on the frequency distributions over *v*. In other words, at a point where the diversity of contexts for *v* approach a maximum, there is little value in adding additional contexts containing *v*, *e.g.*, to translation models.

The optimal algorithm would then, for each *v* ∈ *V*, identify the number of unique contexts that contain *v* up to some threshold and discard all others. An exhaustive algorithm which sets thresholds for all n-gram contexts containing *v*, however, would take a large amount of time to run (minimally quadratic), and may also overrun memory limitations on large data sets.

For VSF, we made the following simplifying assumption: we set an arbitrary count threshold *t* for all vocabulary items. For any given *v*, when we reach *t*, we no longer need to keep additional sentences containing *v*. However, since each instance of *v* does not exist in isolation, but is rather contained within sentences that also contain other vocabulary items *v*, which, in turn, also need to be counted and thresholded, we simplified VSF even further with the following heuristic: for any given sentence *s*, if all *v* ∈ *V* within *s* have *not* reached *t*, then the sentence is kept. This has the direct consequence that many vocabulary items will have frequencies above *t* in the output corpus.

The implementation of VSF is described in **Algorithm 1** below.

VSF clearly makes a number of simplifying assumptions, many of which one might argue would reduce the value of the resulting data. Although easy to implement, it may not achieve the most optimal results. Assuming that VSF might be defective, we then looked into other algorithms attempting to achieve the same or similar results, such as those described in Section 2, and explored in-depth the algorithms described in (Eck et al., 2005).

Input: ParallelCorpus, N, L
Output: SelectedCorpus
foreach $sp \in$ ParallelCorpus **do**
  $S \leftarrow$ `EnumNgrams(`$sp.src$, L`)`;
  $T \leftarrow$ `EnumNgrams(`$sp.tgt$, L`)`;
  $selected \leftarrow false$;
  foreach $(s, t) \in (S, T)$ **do**
    if SrcCnt [s]<N ∨ TgtCnt [t]<N **then**
      $selected \leftarrow true$;
    end
  end
  if $selected$ **then**
    SelectedCorpus.`Add(`sp`)`;
    foreach $(s, t) \in (S, T)$ **do**
      SrcCnt [s]++;
      TgtCnt [t]++;
    end
  end
end

Algorithm 1: Pseudocode for implementing VSF. L: n-gram length, N: n-gram threshold.

4 An Alternative: (Eck et al., 2005)

In our pursuit of better and generic data reduction algorithms, we did a number of experiments using the algorithms described in (Eck et al., 2005). In the n-gram based method proposed by this work the weight of each function is calculated using Equation 1, where j is the n-gram length. In each iteration of the algorithm, the weight of each sentence is calculated and the sentence with the highest weight is selected. Once a sentence is selected, the n-grams in the sentence are marked as seen and have a zero weight when they appear in subsequent sentences. Therefore, the weights of all remaining sentences have to be recalculated before the next sentence can be selected. We refer to this algorithm henceforth as the Eck algorithm.

$$W_j\,(sentence) = \frac{\sum_{i=1}^{j} \left[\sum_{\substack{unseen \\ ngrams}} Freq(ngram) \right]}{|sentence|} \quad (1)$$

To compare VSF against the Eck algorithm we selected the English-Lithuanian parallel corpus from JRC-ACQUIS (Steinberger et al., 2006). We selected the corpus for the following reasons:

- VSF performance on this particular data set was at its lowest compared to a number of other data sets, so there was room for improvement by a potentially better algorithm.
- With almost 50 million tokens combined (English and Lithuanian) we were able to optimize the Eck algorithm and run it on this data set in a reasonable amount of time. The experiments run by the original paper in 2005 were run on only 800,000 tokens.

Using the Eck algorithm with n-gram length set to one ($j \leftarrow 1$ in Equation 1) only 10% (5,020,194 tokens total) of the data is sorted, since all n-grams of size one have been observed by that point and the weight function for the remaining sentences returns zero. In other words, since there are no unseen unigrams after 10% of the data has been sorted, in Equation 1, the numerator becomes zero there after and therefore the remaining 90% of sentence pairs are not sorted. This must be taken into consideration when examining the comparison between unigram VSF and the Eck algorithm with n-gram length set to one in Figure 2. VSF with its lowest setting, that is threshold t=1, selects 20% of the data, so this chart may not be a fair comparison between the two algorithms.

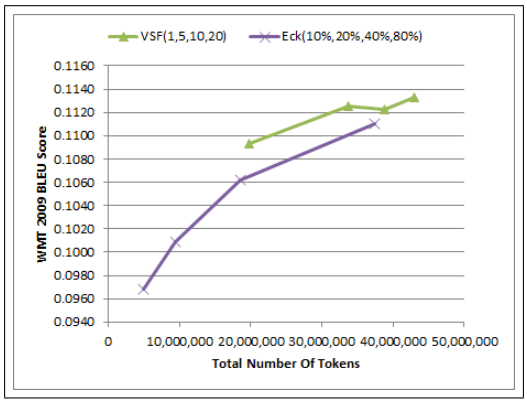


Figure 2: Unigram Eck vs. Unigram VSF

In an attempt to do a fairer comparison, we also tried n-grams of length two in the Eck algorithm, where 50% of the data can be sorted (since all unigrams and bigrams are observed by that point). As seen in Figure 3, the BLEU scores for the Eck and VSF systems built on the similar sized data score very closely on the WMT 2009 test set.[3]

Further exploring options using Eck, we developed the following two extensions to the Eck algorithm, none of which resulted in a significant gain in BLEU score over VSF with n-gram lengths set up to three.

- Incorporating target sentence n-grams in addition to source side sentence n-grams.
- Dividing the weight of an n-gram (its frequency) by a constant number each time a sentence that contains the n-gram is selected, as opposed to setting the weight of an n-gram to zero after it has been seen for the first time.[4]

In relatively small data sets there is not a significant difference between the two algorithms. The Eck algorithm does not scale to larger data sets and higher n-grams. Since a principal focus of our work is on scaling to very large data sets, and since Eck could not scale to even moderately sized data sets, we decided to continue our focus on VSF and improvements to that algorithm.

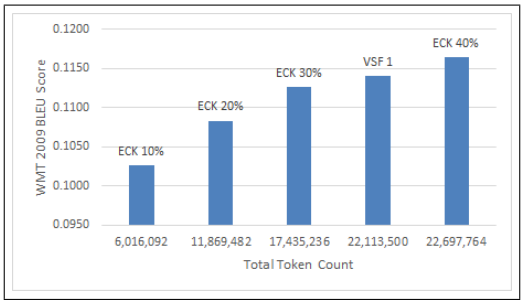


Figure 3: Bigram Eck vs. Unigram VSF

5 Data Order

Unlike the Eck algorithm, VSF is sensitive to the order of the input data due to the nature of the algorithm. Depending on the order of sentences in the input parallel corpus, VSF could select different subsets of the parallel corpus that would eventually (after training and test) result in different BLEU scores. To address this concern we use a feature function inspired by (Denkowski et al., 2012a) which is a normalized combined alignment score. This feature score is obtained by geometric averaging of the normalized forward and backward alignment scores which in turn are calculated using the process described in (Denkowski et al., 2012a). To keep the algorithm as scalable as possible we use radix sort. This ordering of the data ensures sentences with high normalized alignment scores appear first and sentences with low normalized alignment appear last. As a result, for each n-gram, VSF will choose the top-N highest scoring sentence pairs that contain that n-gram.

[3]The careful reader may note that there is no official WMT09 test set for Lithuanian, since Lithuanian is not (yet) a language used in the WMT competition. The test set mentioned here was created from a 1,000 sentence sample from the English-side of the WMT09 test sets, which we then manually translated into Lithuanian.

[4]Further details of the modifications to the Eck algorithm are not discussed here as they did not yield improvements over the baseline algorithm and the focus of our work presented here was shifted to improvements over VSF.

5.1 Data Ordering Complexity

Ordering the data based on normalized combined alignment score requires two steps. First, the normalized combined alignment score is computed for each sentence pair using an existing HMM alignment model. Next, sentence pairs are sorted based on the calculated score. The computational complexity of aligning a single sentence pair is $O\left(J + I^2\right)$ where J is the number of words in the source sentence and I is the number of words in the target sentence (Gao and Vogel, 2008). Therefore the complexity of calculating the combined alignment score would be $O\left(N \times (J^2 + I + I^2 + J)\right)$ or $O\left(N \times \max(I, J)^2\right)$ after simplification. Since radix sort is used for sorting the data, the data can be sorted in $O(d \times N)$ where d is the number of significant digits used for sorting. Since d is kept constant[5], the overall computational complexity for data ordering is $O\left(N + N \times \max(I, J)^2\right)$.

6 Experiments

6.1 The Machine Translation and Training Infrastructure

We used a custom-built tree-to-string (T2S) system for training the models for all experiments. The T2S system that we developed uses technology described in (Quirk et al., 2005), and requires a source-side dependency parser, which we have developed for English.[6] We trained a 5-gram French LM over the entire EnFrGW, which we used in all systems. We used Minimum Error Rate Training (MERT) (Och, 2003) for tuning the lambda values for all systems, tuned using the official WMT2010 dev data.

6.2 Test and Training Data

In all experiments, we used the EnFrGW corpus, or subsets thereof. [7] We used three test sets in all experiments, as well. Two consisted of the WMT 2009 and 2010 test sets, used in the WMT competitions in the respective years. The third consisted of 5,000 parallel English/French sentences sampled from logs of actual traffic received by our production service, Bing Translator (http://bing.com/translator), which were then manually translated. The first two test sets are publicly available, but are somewhat news focused. The third, which we will call ReqLog, consists of a mix of content and sources, so can be considered a truly "general" test set.

t =	Random	VSF	Ordered VSF
1	1.83 M	1.83 M	1.68 M
2	2.53 M	2.53 M	2.34 M
5	3.62 M	3.62 M	3.35 M
10	4.62 M	4.62 M	4.29 M
20	5.83 M	5.83 M	5.44 M
40	7.26 M	7.26 M	6.83 M
60	8.21 M	8.21 M	7.78 M
100	9.53 M	9.53 M	9.13 M
150	10.67 M	10.67 M	10.33 M
200	11.53 M	11.53 M	11.23 M
250	12.22 M	12.22 M	11.97 M
All	22.5 M		

Table 2: English-side Sentence Counts (in millions) for different thresholds for VSF, VSF after ordering the data based on normalized combined alignment score and random baselines.

To discern the effects of VSF at different degrees of "saturation", we tried VSF with different threshold values *t*, ranging from 1 to 250. For each *t* value we actually ran VSF twice. In the first case, we did no explicit sorting of the data. In the second case, we ranked the data using the method described in Section 5.

Finally, we created random baselines for each *t*, where each random baseline is paired with the relevant VSF run, controlled for the number of sentences (since *t* has no relevance for random samples). The different *t* values and the resulting training data sizes (sentence and word counts) are shown in Tables 2 and 3.

Since our interest in this study is scaling parallel data, for all trainings we used the same LM, which was built over all training data (the French side of the full EnFrGW). Because monolingual training scales much more readily than parallel,

[5] In experiments described in Section 6 five significant digits were used for radix sort.

[6] Further details about the decoders is beyond the scope of this paper. The reader is encouraged to refer to the sources provided for additional information.

[7] Because of some data cleaning filters we applied to the data, the actual full sized corpus we used consisted of slightly less data than that used in the WMT competitions. Every team has its own set of favorite data cleaning heuristics, and ours is no different. The filters applied to this data are focused mostly on noise reduction, and consist of a set of filters related to eliminating content that contains badly encoded characters, removing content that is too long (since there is little value in training on very long sentences), removing content where the ratio between numeric versus alphabetic characters is excessively large, deleting content where the script of the content is mostly not in latin1 (relevant for French), and some additional filters described in (Denkowski et al., 2012b). If the reader wishes additional material on data filtration, please see (Denkowski et al., 2012b) and (Lewis and Quirk, 2013).

t =	Random	VSF	Ordered VSF
1	46.1 M	64.52 M	65.74 M
2	63.99 M	87.41 M	88.12 M
5	91.55 M	121.3 M	120.86 M
10	116.83 M	151.53 M	149.95 M
20	147.31 M	186.99 M	184.14 M
40	183.46 M	228.14 M	224.29 M
60	207.42 M	254.89 M	250.68 M
100	240.88 M	291.45 M	287.02 M
150	269.77 M	322.5 M	318.33 M
200	291.4 M	345.37 M	341.69 M
250	308.83 M	363.44 M	360.32 M
All	583.97 M		

Table 3: English-side Word Counts for different thresholds for VSF, VSF after ordering the data based on normalized combined alignment score and random baselines.

t =	Random	VSF	Ordered VSF
1	1:07	2:17	1:56
2	1:33	2:55	2:39
5	2:15	4:05	3:47
10	2:43	4:49	4:50
20	3:23	5:25	5:14
40	4:12	6:16	5:56
60	4:45	6:41	7:15
100	5:31	7:32	7:55
150	6:07	8:20	8:18
200	6:36	8:31	8:52
250	7:30	9:19	9:11
All	13:12		

Table 4: Word alignment times (hh:mm) for different thresholds for VSF, VSF after model score ordering, and a random baseline

this seemed reasonable. Further, using one LM controls one parameter that would otherwise fluctuate across trainings. The result is a much more focused view on parallel training diffs.

6.3 Results

We trained models over each set of data. In addition to calculating BLEU scores for each resulting set of models in (Table 5), we also compared OOV rates (Table 6) and performance differences (Table 4). The former is another window into the "quality" of the resulting models, in that it describes vocabulary coverage (in other words, how much vocabulary is recovered from the full data). The latter gives some indication regarding the time savings after running VSF at different thresholds.

On the WMT09 data set, both sets of VSF models outperformed the relevant random baselines. On the WMT10 and ReqLog test sets, the pre-sorted VSF outperformed all random baselines, with the unsorted VSF outperforming most random baselines, except at t=60 and t=200 for WMT10. For the ReqLog, unsorted VSF drops below random starting at t=200. Clearly, the t=200 results show that there is less value in VSF as we approach the total data size.

The most instructive baseline, however, is the one built over all training data. It is quite obvious that at low threshold values, the sampled data is not a close approximation of the full data: not enough vocabulary and contextual information is preserved for the data to be taken as a proxy for the full data. However, with t values around 20-60 we recover enough BLEU and OOVs to make the datasets reasonable proxies. Further, because we see a relative reduction in data sizes of 32-44%, model size reductions of 27-39%, and performance improvements of 41-50% at these t values further argues for the value of VSF at these settings. Even at t=250, we have training data that is 54% of the full data size, yet fully recovers BLEU.

7 Discussion

VSF is a simple but effective algorithm for reducing the size of parallel training data, and does so independently of particular dev or test data. It performs as well as related algorithms, notably (Eck et al., 2005), but more importantly, it is able to scale to much larger data sets than other algorithms. In this paper, we showed VSF applied to the EnFrGW corpus. It should be noted, however, that we have also been testing VSF on much larger sets of English-French data. Two notable tests are one applied to 65.2M English-French sentence pairs and another applied to one consisting of 162M. In the former case, we were able to reduce the corpus size from 65.2M sentences/1.28B words[8] to 26.2M sentences/568M words. The BLEU score on this test was stable on the three test sets, as shown in Table 7. When applied to the 162M sentence/2.1B word data set, we were able to reduce the data size to 40.5M sentences/674M words. In this case, sorting the data using model scores produced the most desirable results, actually increasing BLEU by 0.90 on WMT09, but, unfortunately, showing a 0.40 drop on WMT10.

The fact that VSF runs in one pass is both an asset and a liability. It is an asset since the algorithm is able to operate linearly with respect to the size the data. It is a liability since the algorithm is

[8]Word counts based on the English-side, unwordbroken.

		WMT09			WMT10			ReqLog	
t =	Random	VSF	S+VSF	Random	VSF	S+VSF	Random	VSF	S+VSF
1	23.76	23.83	23.84	25.69	25.78	25.68	26.34	26.63	26.67
2	23.91	24.04	24.07	25.76	26.21	26.14	26.54	26.99	26.94
5	24.05	24.29	24.40	26.10	26.40	26.32	26.79	27.22	27.12
10	24.15	24.37	24.45	26.21	26.63	26.32	26.98	27.37	27.62
20	24.20	24.40	24.55	26.30	26.46	26.56	27.22	27.38	27.44
40	24.37	24.43	24.65	26.40	26.55	26.53	27.30	27.38	27.62
60	24.32	24.43	24.64	26.56	26.56	26.61	27.38	27.50	27.64
100	24.37	24.49	24.71	26.46	26.75	26.70	27.37	27.52	27.75
150	24.37	24.61	24.71	26.67	26.67	26.70	27.48	27.62	27.75
200	24.48	24.63	24.69	26.56	26.65	26.78	27.57	27.47	27.72
250	24.41	24.57	24.85	26.62	26.74	26.68	27.63	27.45	27.76
All	24.37			26.54			27.63		

Table 5: BLEU Score results for VSF, S+VSF (Sorted VSF), and Random Baseline at different thresholds t.

		WMT09			WMT10			ReqLog	
t =	Random	VSF	S+VSF	Random	VSF	S+VSF	Random	VSF	S+VSF
1	630	424	450	609	420	445	1299	973	1000
2	588	374	395	559	385	393	1183	906	919
5	520	343	347	492	350	356	1111	856	853
10	494	336	335	458	344	344	1092	837	848
20	453	335	335	432	339	341	1016	831	834
40	423	330	331	403	336	337	986	828	833
60	419	329	330	407	333	336	964	831	832
100	389	330	329	391	333	335	950	830	830
150	397	330	330	384	332	332	930	828	828
200	381	328	330	371	331	332	912	827	826
250	356	329	328	370	333	331	884	823	823
All	325			331			822		

Table 6: OOV rates for VSF, S+VSF (Sorted VSF), and Random Baseline at different thresholds t.

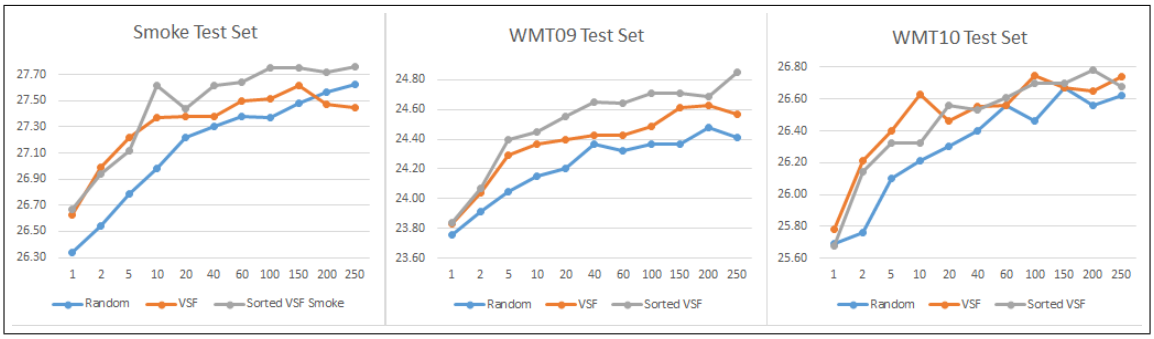


Figure 4: Comparative BLEU scores for two VSF implementations, against a randomly sampled baseline.

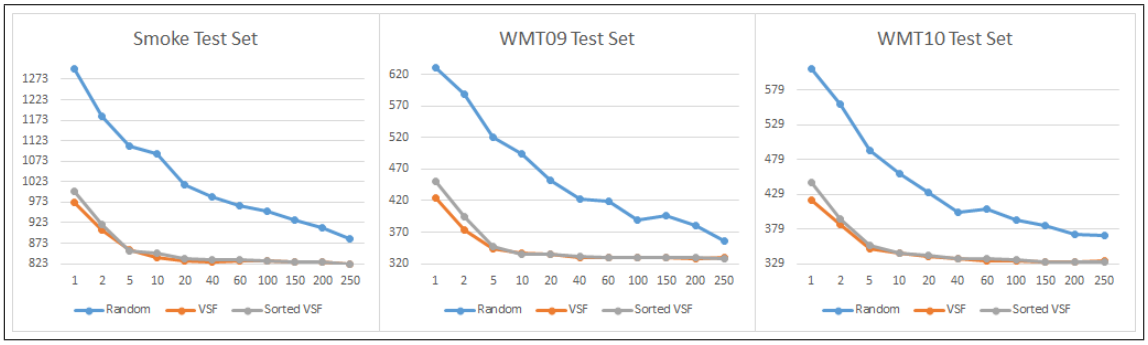


Figure 5: Comparative OOV rates for two VSF implementations, against a randomly sampled baseline.

	ReqLog	WMT09	WMT10
65.2 snts	32.90	26.77	29.05
VSF 26.2M snts	33.34	26.75	29.07

Table 7: VSF applied to a 65.2M sentence baseline system.

sensitive to the order of the data. The latter leads to issues of reproducibility: with poorly ordered data, one could easily arrive at a much less than optimal set of data. However, by adding an additional pass to build model scores, and then ranking the data by these scores, we address the serious issue of reproducibility. Further, the ranking tends to arrive at a better selection of data.

In an attempt to better understand the behavior of VSF and how VSF changes the n-gram distributions of vocabulary items in a sample as compared to the full corpus, we created log_2-scale scatter plots, as seen in Figure 6. In these plots, unigram frequencies of unfiltered data (*i.e.*, the full corpus, EnFrGW) are on the vertical axis, and unigram frequencies of the VSF filtered data are on the horizontal axis. The three plots show three different settings for *t*. There following observations can be made about these plots:

1. On the horizontal axis before we reach $log_2(t)$, all data points fall on the $x = y$ line.
2. As the threshold increases the scatter plot gets closer to the $x = y$ line.
3. VSF has the highest impact on the "medium" frequency unigrams, that is, those with a frequency higher than the threshold.

The third point speaks the most to the effects that VSF has on data: Very low frequency items, specifically those with frequencies below the threshold *t*, are unaffected by the algorithm, since we guarantee including all contexts in which they occur. Low frequency items are at the lower left of the plots, and their frequencies follow the $x = y$ line (point 1 above). Medium frequency items, however, specifically those with frequencies immediately above *t*, are the most affected by the algorithm. The "bulge" in the plots shows where these medium frequency items begin, and one can see plainly that their distributions are perturbed. The "bulge" dissipates as frequencies increase, until the effects diminish as we approach much higher frequencies. The latter is a consequence of a simplifying heuristic applied in VSF (as described in Section 3): *t* is not a hard ceiling, but rather a soft one. Vocabulary items that occur very frequently in a corpus will be counted many more times than *t*; for very high frequency items, their sampled distributions may approach those observed in the full corpus, and converge on the $x = y$ line. The authors suspect that the BLEU loss that results from the application of VSF is the result of the perturbed distributions for medium frequency items. Adjusting to higher *t* values decreases the degree of the perturbation, as noted in the second point, which likewise recovers some of the BLEU loss observed in lower settings.

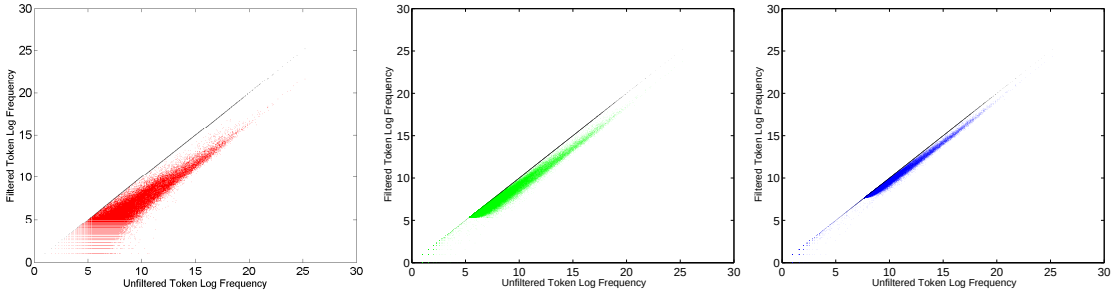
8 Future Work

There are several future directions we see with work on VSF. Because one threshold *t* for *all* vocabulary items may be too coarse a setting, we first plan to explore setting *t* based on frequency, especially for vocabulary in the most affected mid-range (at and above t). If we set *t* based on unigrams falling into frequency buckets, rather than one setting for all unigrams, we may arrive earlier at a more distributionally balanced corpus, one that may better match the full corpus. That said, additional passes over the data come at additional cost.

Second, we plan to explore applying the VSF algorithm to higher order n-grams (all experiments thus far have been on unigrams). Preliminary experiments on bigram VSF, however, show that with even the lowest setting (*t*=1), we already preserve well over 50% of the data.

In this work we only experimented with sorting the data based on the normalized combined alignment score inspired by (Eck et al., 2005). A third direction for future work would be to consider ordering the data based on other feature functions presented in Eck, *e.g.*, source and target language model, alignment ratio, as well as and feature functions introduced in (Taghipour et al., 2010), or a combination of all of these feature functions.

In the fourth case, we plan to do more sophisticated statistical analysis of the effects of VSF on n-gram distributions and phrase-table entropy. We also plan to explore the interactions between VSF and data "diversity". For instance, VSF may have a greater positive impact on more narrowly focused domains than on those that are more generally focused.



(a) VSF $t = 1$ (b) VSF $t = 40$ (c) VSF $t = 200$

Figure 6: log_2-scale Unigram Frequency scatter plot before VSF versus after VSF

References

S. Ananthakrishnan, R. Prasad, D. Stallard, and P. Natarajan. 2010. Discriminative sample selection for statistical machine translation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, page 626635.

A. Axelrod, X. He, and J. Gao. 2011. Domain adaptation via pseudo in-domain data selection. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, page 355362.

Chris Callison-Burch, Philipp Koehn, Christof Monz, and Josh Schroeder. 2009. Findings of the 2009 Workshop on Statistical Machine Translation. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 1–28, Athens, Greece, March. Association for Computational Linguistics.

W. Chao and Z. Li. 2011a. A graph-based bilingual corpus selection approach for SMT. In *Proceedings of the 25th Pacific Asia Conference on Language, Information and Computation*.

WenHan Chao and ZhouJun Li. 2011b. Improved graph-based bilingual corpus selection with sentence pair ranking for statistical machine translation. In *2011 23rd IEEE International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 446 –451, November.

Michael Denkowski, Greg Hanneman, and Alon Lavie. 2012a. The CMU-Avenue French-English translation system. In *Proceedings of the NAACL 2012 Workshop on Statistical Machine Translation*.

Michael Denkowski, Greg Hanneman, and Alon Lavie. 2012b. The CMU-Avenue French-English Translation System. In *Proceedings of the NAACL 2012 Workshop on Statistical Machine Translation*.

M. Eck, S. Vogel, and A. Waibel. 2005. Low cost portability for statistical machine translation based in n-gram frequency and TF-IDF. In *International Workshop on Spoken Language Translation (IWSLT)*.

Qin Gao and Stephan Vogel. 2008. Parallel implementations of word alignment tool. In *Software Engineering, Testing, and Quality Assurance for Natural Language Processing*, page 4957.

G. Haffari, M. Roy, and A. Sarkar. 2009. Active learning for statistical phrase-based machine translation. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, page 415423.

Howard Johnson, Joel D. Martin, George F. Foster, and Roland Kuhn. 2007. Improving translation quality by discarding most of the phrasetable. In *Proceedings of EMNLP*, pages 967–975.

William D. Lewis and Chris Quirk. 2013. Controlled Ascent: Imbuing Statistical MT with Linguistic Knowledge. In *Proceedings of the Second Hytra (Hybrid Approaches to Translation) Workshop*, Sofia, Bulgaria, August.

H. Lin and J. Bilmes. 2011. Optimal selection of limited vocabulary speech corpora. In *Proc. Interspeech*.

Robert C. Moore and William D. Lewis. 2010. Intelligent Selection of Language Model Training Data. In *Proceedings of the ACL 2010 Conference Short Papers*, Uppsala, Sweden, July.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st ACL*, Sapporo, Japan.

T. Okita. 2009. Data cleaning for word alignment. In *Proceedings of the ACL-IJCNLP 2009 Student Research Workshop*, page 7280.

Chris Quirk, Arul Menezes, and Colin Cherry. 2005. Dependency tree translation: Syntactically informed phrasal smt. In *Proceedings of ACL 2005*.

Spencer Rarrick, Chris Quirk, and William D. Lewis. 2011. MT Detection in Web-Scraped Parallel Corpora. In *Proceedings of MT Summit XIII*, Xiamen, China, September.

Ralf Steinberger, Bruno Pouliquen, Anna Widiger, Camelia Ignat, Toma Erjavec, and Dan Tufi. 2006.

The JRC-Acquis: a multilingual aligned parallel corpus with 20+ languages. In *In Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC'2006*, page 21422147.

K. Taghipour, N. Afhami, S. Khadivi, and S. Shiry. 2010. A discriminative approach to filter out noisy sentence pairs from bilingual corpora. In *2010 5th International Symposium on Telecommunications (IST)*, pages 537 –541, December.

K. Yasuda, R. Zhang, H. Yamamoto, and E. Sumita. 2008. Method of selecting training data to build a compact and efficient translation model. In *Proceedings of the Third International Joint Conference on Natural Language Processing*, volume 2, page 655660.