Why Not Grab a Free Lunch? Mining Large Corpora for Parallel Sentences to Improve Translation Modeling

Ferhan Ture
Dept. of Computer Science,
University of Maryland
fture@cs.umd.edu

Jimmy Lin
The iSchool
University of Maryland
jimmylin@umd.edu

Abstract

It is well known that the output quality of statistical machine translation (SMT) systems increases with more training data. To obtain more parallel text for translation modeling, researchers have turned to the web to mine parallel sentences, but most previous approaches have avoided the difficult problem of pairwise similarity on cross-lingual documents and instead rely on heuristics. In contrast, we confront this challenge head on using the MapReduce framework. On a modest cluster, our scalable end-to-end processing pipeline was able to automatically gather 5.8m parallel sentence pairs from English and German Wikipedia. Augmenting existing bitext with these data yielded significant improvements over a state-of-the-art baseline (2.39 BLEU points in the best case).

1 Introduction

It has been repeatedly shown that "throwing more data at the problem" is effective in increasing SMT output quality, both for translation modeling (Dyer et al., 2008) and for language modeling (Brants et al., 2007). In this paper, we bring together two related research threads to gather parallel sentences for improved translation modeling: cross-lingual pairwise similarity to mine comparable documents and classification to identify sentence pairs that are mutual translations.

Unlike most previous work, which sidesteps the computationally-intensive task of pairwise comparisons to mine comparable documents and instead relies on heuristics, we tackle the challenge head on. This paper describes a fully open-source, scalable MapReduce-based processing pipeline that is able to automatically extract large quantities of parallel sentences. Experiments examine the impact data size has on a state-of-the-art SMT system.

We acknowledge that different components of this work are not novel and the general principles behind "big data" MT are well known. However, when considered together with our previous work (Ture et al., 2011), to our knowledge this is the first exposition in which all the pieces have been "put together" in an end-to-end pipeline that is accessible to academic research groups. The framework described in this paper is entirely open source, and the computational resources necessary to replicate our results are relatively modest.

Starting from nothing more than two corpora in different languages (in German and English, in our case), we are able to extract bitext and improve translation quality by a significant margin (2.39 BLEU points), essentially "for free". By varying both the quantity and quality of the bitext, we characterize the tradeoffs between the amount of data, computational costs, and translation quality.

2 Related Work

The idea of mining parallel sentences, particularly from the web, is of course not new. Most adopt a two step process: 1. identify comparable documents and generate candidate sentence pairs, and 2. filter candidate pairs to retain parallel sentences.

The general solution to the first step involves computing pairwise similarities across multi-lingual corpora. As this is computationally intensive, most

2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 626–630,
Montréal, Canada, June 3-8, 2012.

studies fall back to heuristics, e.g., comparing news articles close in time (Munteanu and Marcu, 2005), exploiting "inter-wiki" links in Wikipedia (Smith et al., 2010), or bootstrapping off an existing search engine (Resnik and Smith, 2003). In contrast, we adopt a more exhaustive approach by directly tackling the cross-lingual pairwise similarity problem, using MapReduce on a modest cluster. We perform experiments on German and English Wikipedia (two largest available), but our technique is general and does not depend on sparse, manually-created inter-wiki links. Thus, compared to those approaches, we achieve much higher recall.

The second step (filtering candidate sentence pairs) is relatively straightforward, and we adopt the classification approach of Munteanu and Marcu (2005). However, unlike in previous work, we need to classify large volumes of data (due to higher recall in the first step). Therefore, we care about the relationship between classification accuracy and the speed of the classifier. Our two-stage approach gives us both high effectiveness (accuracy) and efficiency (speed).

A recent study from Google describes a general solution to our problem that scales to web collections (Uszkoreit et al., 2010). The authors translate all documents from one language into another, thus transforming the problem into identifying similar mono-lingual document pairs. Nevertheless, our approach makes several additional contributions. First, we explore the effect of dataset size on results. Our conclusions are more nuanced than simply "more data is better", since there is a tradeoff between quality and quantity. Our experiments involve orders of magnitude *less* data, but we nevertheless observe significant gains over a strong baseline. Overall, our approach requires far less computational resources and thus is within the reach of academic research groups: we do not require running an MT system on one side of the entire collection, and we carefully evaluate and control the speed of sentence-classification. Finally, in support of open science, our code[1] and data[2] are available as part of Ivory, an open-source Hadoop toolkit for web-scale information retrieval (Lin et al., 2009).

[1] ivory.cc

[2] github.com/ferhanture/WikiBitext

3 Generating Candidate Sentences

We applied our approach on English Wikipedia (10.9m documents, 30.6GB) and German Wikipedia (2.4m articles, 8.5GB), using XML dumps from January 2011. English and German Wikipedia were selected because they are the largest Wikipedia collections available, and we want to measure effects in a language for which we already have lots of bitext. In both collections, redirect pages and stub articles were discarded.

To mine comparable documents, we used our previously described algorithm (Ture et al., 2011), based on local-sensitive hashing, also implemented in Hadoop MapReduce. The reader is referred to the paper for details. On a 16 node (96 core) cluster, we were able to extract 64m (d_e, d_f) document pairs (with cosine similarity ≥ 0.3) in 8.8 hours.

For each of the (d_e, d_f) pairs, the next processing step involves generating the Cartesian product of sentences in both documents as candidate sentence pairs: this itself is a non-trivial problem. Although in this particular case it may be possible to load both document collections in memory, we envision scaling up to collections in the future for which this is not possible. Therefore, we devised a scalable, distributed, out-of-memory solution using Hadoop.

The algorithm works as follows: We map over (docid n, document d) pairs from both the German and English collections. In each mapper all (d_e, d_f) similarity pairs are loaded in memory. If the input document is not found in any of these pairs, no work is performed. Otherwise, we extract all sentences and retain only those that have at least 5 terms and at least 3 unique terms. Sentences are converted into BM25-weighted vectors in the English term space; for German sentences, translation into English is accomplished using the technique proposed by Darwish and Oard (2003). For every (d_e, d_f) pair that the input document is found in, the mapper emits the list of weighted sentence vectors, with the (d_e, d_f) pair as the key. As all intermediate key-value pairs in MapReduce are grouped by their keys for reduce-side processing, the reducer receives the key (d_e, d_f) and weighted sentence vectors for both the German and English articles. From there, we generate the Cartesian product of sentences in both languages. As an initial filtering step, we discard all pairs where

the ratio of sentence lengths is more than two, a heuristic proposed in (Munteanu and Marcu, 2005). Each of the remaining candidate sentences are then processed by two separate classifiers: a less accurate, fast classifier and a more accurate, slow classifier. This is described in the next section.

This algorithm is a variant of what is commonly known as a reduce-side join in MapReduce (Lin and Dyer, 2010), where (d_e, d_f) serves as the join key. Note that in this algorithm, sentence vectors are emitted multiple times, one for each (d_e, d_f) pair that they participate in: this results in increased network traffic during the sort/shuffle phase. We experimented with an alternative algorithm that processes all foreign documents similar to the same English document together, e.g., processing $(d_e, [d_{f1}, d_{f2}, \ldots])$ together. This approach, counter-intuitively, was slower despite reduced network traffic, due to skew in the distribution of similar document pairs. In our experiments, half of the source collection was not linked to any target document, whereas 4% had more than 100 links. This results in reduce-side load imbalance, and while most of the reducers finish quickly, a few reducers end up performing substantially more computation, and these "stragglers" increase end-to-end running time.

4 Parallel Sentence Classification

We built two MaxEnt parallel sentence classifiers using the OpenNLP package, with data from a sample of the Europarl corpus of European parliament speeches. For training, we sampled 1000 parallel sentences from the German-English subset of the corpus as positive instances, and 5000 non-parallel sentence pairs as negative instances. For testing, we sampled another 1000 parallel pairs and generated all possible non-parallel pairs by the Cartesian product of these samples. This provides a better approximation of the task we're interested in, since most of the candidate sentence pairs will be non-parallel in a comparable corpus. We report precision, recall, and F-score, using different classifier confidence scores as the decision threshold (see Table 1).

Our first, *simple* classifier, which uses cosine similarity between the sentences as the only feature, achieved a maximum F-score of 74%, with 80% precision and 69% recall. Following previous work

Classifier	Measure	Value
Simple	Recall @ P90	0.59
	Recall @ P80	0.69
	Best F-score	0.74
Complex	Recall @ P90	0.69
	Recall @ P80	0.79
	Best F-score	0.80

Table 1: Accuracy of the simple and complex sentence classifiers on Europarl data.

(Smith et al., 2010), we also report recall with precision at 80% and 90% in Table 1; the classifier effectiveness is comparable to the previous work. The second, *complex* classifier uses the following additional features: ratio of sentence lengths, ratio of source-side tokens that have translations on the target side, ratio of target-side tokens that have translations on the source side. We also experimented with features using the word alignment output, but there was no improvement in accuracy. The complex classifier showed better performance: recall of 79% at 80% precision and 69% at precision of 90%, with a maximum F-score of 80%.

Due to the large amounts of data involved in our experiments, we were interested in speed/accuracy tradeoffs between the two classifiers. Microbenchmarks were performed on a commodity laptop running Mac OS X on a 2.26GHz Intel Core Duo CPU, measuring per-instance classification speed (including feature computation time). The complex classifier took 100 μs per instance, about 4 times slower than the simple one, which took 27 μs.

The initial input of 64m similar document pairs yielded 400b raw candidate sentence pairs, which were first reduced to 214b by the per-sentence length filter, and then to 132b by enforcing a maximum sentence length ratio of 2. The simple classifier was applied to the remaining pairs, with different confidence thresholds. We adjusted the threshold to obtain different amounts of bitext, to see the effect on translation quality (this condition is called S_1 hereafter). The positive results of the first classifier was then processed by the second classifier (this two-level approach is called S_2 hereafter).

Candidate generation was completed in 2.4 hours on our cluster with 96 cores. These candidates went through the MapReduce shuffle-and-sort process in 0.75 hours, which were then classified in 4 hours.

Processing by the more complex classifier in S_2 took an additional 0.52 hours.

5 End-to-End MT Experiments

In all experiments, our MT system learned a synchronous context-free grammar (Chiang, 2007), using GIZA++ for word alignments, MIRA for parameter tuning (Crammer et al., 2006), cdec for decoding (Dyer et al., 2010), a 5-gram SRILM for language modeling, and single-reference BLEU for evaluation. The baseline system was trained on the German-English WMT10 training data, consisting of 3.1m sentence pairs. For development and testing, we used the newswire datasets provided for WMT10, including 2525 sentences for tuning and 2489 sentences for testing.

Our baseline system includes all standard features, including phrase translation probabilities in both directions, word and arity penalties, and language model scores. It achieves a BLEU score of 21.37 on the test set, which would place it 5^{th} out of 9 systems that reported comparable results in WMT10 (only three systems achieved a BLEU score over 22). Many of these systems used techniques that exploited the specific aspects of the task, e.g., German-specific morphological analysis. In contrast, we present a knowledge-impoverished, entirely data-driven approach, by simply looking for more data in large collections.

For both experimental conditions (one-step classification, S_1, and two-step classification, S_2) we varied the decision threshold to generate new bitext collections of different sizes. Each of these collections was added to the baseline training data to induce an entirely new translation model (note that GIZA additionally filtered out some of the pairs based on length). The final dataset sizes, along with BLEU scores on the test data, are shown in Fig. 1. In S_1, we observe that increasing the amount of data (by lowering the decision threshold) initially leads to lower BLEU scores (due to increased noise), but there is a threshold after which the improvement coming from the added data supersedes the noise. The S_2 condition increases the quality of bitext by reducing this noise: the best run, with 5.8m pairs added to the baseline (final dataset has 8.1m pairs), yields 23.76 BLEU (labeled P on figure), 2.39 points above the baseline (and higher than the best WMT10 result). These results show that the two-step classification process, while slower, is worth the additional processing time.

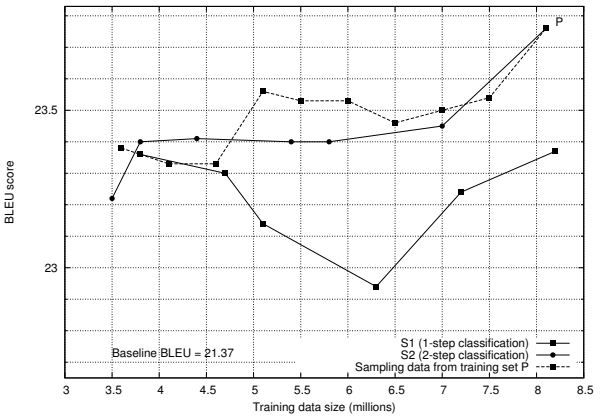

Figure 1: Evaluation results on the WMT10 test set.

Our approach yields solid improvements even with less data added: with only 382k pairs added to the baseline, the BLEU score increases by 1.84 points. In order to better examine the effect of data size alone, we created partial datasets from P by randomly sampling sentence pairs, and then repeated experiments, also shown in Fig. 1. We see an increasing trend of BLEU scores with respect to data size. By comparing the three plots, we see that S_2 and random sampling from P work better than S_1. Also, random sampling is not always worse than S_2, since some pairs that receive low classifier confidence turn out to be helpful.

6 Conclusions

In this paper, we describe a scalable MapReduce implementation for automatically mining parallel sentences from arbitrary comparable corpora. We show, at least for German-English MT, that an impoverished, data-driven approach is more effective than task-specific engineering. With the distributed bitext mining machinery described in this paper, improvements come basically "for free" (the only cost is a modest amount of cluster resources). Given the availability of data and computing power, there is simply no reason why MT researchers should not ride the large-data "tide" that lifts all boats. For the benefit of the community, all code necessary to replicate these results have been open sourced, as well as the bitext we've gathered.

Acknowledgments

This research was supported in part by the BOLT program of the Defense Advanced Research Projects Agency, Contract No. HR0011-12-C-0015; NSF under awards IIS-0916043 and CCF-1018625. Any opinions, findings, conclusions, or recommendations expressed in this paper are those of the authors and do not necessarily reflect the view of the sponsors. The second author is grateful to Esther and Kiri for their loving support and dedicates this work to Joshua and Jacob.

References

Thorsten Brants, Ashok C. Popat, Peng Xu, Franz J. Och, and Jeffrey Dean. 2007. Large language models in machine translation. *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 858–867, Prague, Czech Republic.

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

Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. 2006. Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7:551–585.

Kareem Darwish and Douglas W. Oard. 2003. Analysis of anchor text for web search. *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2003)*, pages 261–268, Toronto, Canada.

Chris Dyer, Aaron Cordova, Alex Mont, and Jimmy Lin. 2008. Fast, easy, and cheap: Construction of statistical machine translation models with MapReduce. *Proceedings of the Third Workshop on Statistical Machine Translation at ACL 2008*, pages 199–207, Columbus, Ohio.

Chris Dyer, Adam Lopez, Juri Ganitkevitch, Jonathan Weese, Ferhan Ture, Phil Blunsom, Hendra Setiawan, Vladimir Eidelman, and Philip Resnik. 2010. cdec: A decoder, alignment, and learning framework for finite-state and context-free translation models. *Proceedings of the ACL 2010 System Demonstrations*, pages 7–12, Uppsala, Sweden, July.

Jimmy Lin and Chris Dyer. 2010. *Data-Intensive Text Processing with MapReduce*. Morgan & Claypool Publishers.

Jimmy Lin, Donald Metzler, Tamer Elsayed, and Lidan Wang. 2009. Of Ivory and Smurfs: Loxodontan MapReduce experiments for web search. *Proceedings of the Eighteenth Text REtrieval Conference (TREC 2009)*, Gaithersburg, Maryland.

Dragos Stefan Munteanu and Daniel Marcu. 2005. Improving machine translation performance by exploiting non-parallel corpora. *Computational Linguistics*, 31(4):477–504.

Philip Resnik and Noah A. Smith. 2003. The web as a parallel corpus. *Computational Linguistics*, 29(3):349–380.

Jason R. Smith, Chris Quirk, and Kristina Toutanova. 2010. Extracting parallel sentences from comparable corpora using document level alignment. *Proceedings of Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics (HLT/NAACL 2010)*, pages 403–411, Los Angeles, California.

Ferhan Ture, Tamer Elsayed, and Jimmy Lin. 2011. No free lunch: Brute force vs. locality-sensitive hashing for cross-lingual pairwise similarity. *Proceedings of the 34th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2011)*, pages 943–952, Beijing, China.

Jakob Uszkoreit, Jay M. Ponte, Ashok C. Popat, and Moshe Dubiner. 2010. Large scale parallel document mining for machine translation. *Proceedings of the 23rd International Conference on Computational Linguistics (COLING 2010)*, pages 1101–1109, Beijing, China.