Intelligent Selection of Language Model Training Data

Robert C. Moore **William Lewis**
Microsoft Research
Redmond, WA 98052, USA
{bobmoore,wilewis}@microsoft.com

Abstract

We address the problem of selecting non-domain-specific language model training data to build auxiliary language models for use in tasks such as machine translation. Our approach is based on comparing the cross-entropy, according to domain-specific and non-domain-specifc language models, for each sentence of the text source used to produce the latter language model. We show that this produces better language models, trained on less data, than both random data selection and two other previously proposed methods.

1 Introduction

Statistical N-gram language models are widely used in applications that produce natural-language text as output, particularly speech recognition and machine translation. It seems to be a universal truth that output quality can always be improved by using more language model training data, but only if the training data is reasonably well-matched to the desired output. This presents a problem, because in virtually any particular application the amount of in-domain data is limited.

Thus it has become standard practice to combine in-domain data with other data, either by combining N-gram counts from in-domain and other data (usually weighting the counts in some way), or building separate language models from different data sources, interpolating the language model probabilities either linearly or log-linearly. Log-linear interpolation is particularly popular in statistical machine translation (e.g., Brants et al., 2007), because the interpolation weights can easily be discriminatively trained to optimize an end-to-end translation objective function (such as BLEU) by making the log probability according to each language model a separate feature function in the overall translation model.

The normal practice when using multiple languages models in machine translation seems to be to train models on as much data as feasible from each source, and to depend on feature weight optimization to down-weight the impact of data that is less well-matched to the translation application. In this paper, however, we show that for a data source that is not entirely in-domain, we can improve the match between the language model from that data source and the desired application output by intelligently selecting a subset of the available data as language model training data. This not only produces a language model better matched to the domain of interest (as measured in terms of perplexity on held-out in-domain data), but it reduces the computational resources needed to exploit a large amount of non-domain-specific data, since the resources needed to filter a large amount of data are much less (especially in terms of memory) than those required to build a language model from all the data.

2 Approaches to the Problem

Our approach to the problem assumes that we have enough in-domain data to train a reasonable in-domain language model, which we then use to help score text segments from other data sources, and we select segments based on a score cutoff optimized on held-out in-domain data.

We are aware of two comparable previous approaches. Lin et al. (1997) and Gao et al. (2002) both used a method similar to ours, in which the metric used to score text segments is their perplexity according to the in-domain language model. The candidate text segments with perplexity less than some threshold are selected.

The second previous approach does not explicitly make use of an in-domain language model, but is still applicable to our scenario. Klakow (2000) estimates a unigram language model from the entire non-domain-specific corpus to be selected

Proceedings of the ACL 2010 Conference Short Papers, pages 220–224,
Uppsala, Sweden, 11-16 July 2010.

from, and scores each candidate text segment from that corpus by the change in the log likelihood of the in-domain data according to the unigram model, if that segment were removed from the corpus used to estimate the unigram model. Those segments whose removal would decrease the log likelihood of the in-domain data more than some threshold are selected.

Our method is a fairly simple variant of scoring by perplexity according to an in-domain language model. First, note that selecting segments based on a perplexity threshold is equivalent to selecting based on a cross-entropy threshold. Perplexity and cross-entropy are monotonically related, since the perplexity of a string s according to a model M is simply $b^{H_M(s)}$, where $H_M(s)$ is the cross-entropy of s according to M and b is the base with respect to which the cross-entropy is measured (e.g., bits or nats). However, instead of scoring text segments by perplexity or cross-entropy according to the in-domain language model, we score them by the difference of the cross-entropy of a text segment according to the in-domain language model and the cross-entropy of the text segment according to a language model trained on a random sample of the data source from which the text segment is drawn.

To state this formally, let I be an in-domain data set and N be a non-domain-specific (or otherwise not entirely in-domain) data set. Let $H_I(s)$ be the per-word cross-entropy, according to a language model trained on I, of a text segment s drawn from N. Let $H_N(s)$ be the per-word cross-entropy of s according to a language model trained on a random sample of N. We partition N into text segments (e.g., sentences), and score the segments according to $H_I(s) - H_N(s)$, selecting all text segments whose score is less than a threshold T.

This method can be justified by reasoning simliar to that used to derive methods for training binary text classifiers without labeled negative examples (Denis et al., 2002; Elkin and Noto, 2008). Let us imagine that our non-domain-specific corpus N contains an in-domain subcorpus N_I, drawn from the same distribution as our in-domain corpus I. Since N_I is statistically just like our in-domain data I, it would seem to be a good candidate for the data that we want to extract from N. By a simple variant of Bayes rule, the probability $P(N_I|s, N)$ of a text segment s, drawn randomly from N, being in N_I is given by

$$P(N_I|s, N) = \frac{P(s|N_I, N)P(N_I|N)}{P(s|N)}$$

Since N_I is a subset of N, $P(s|N_I, N) = P(s|N_I)$, and by our assumption about the relationship of I and N_I, $P(s|N_I) = P(s|I)$. Hence,

$$P(N_I|s, N) = \frac{P(s|I)P(N_I|N)}{P(s|N)}$$

If we could estimate all the probabilities in the right-hand side of this equation, we could use it to select text segments that have a high probability of being in N_I.

We can estimate $P(s|I)$ and $P(s|N)$ by training language models on I and a sample of N, respectively. That leaves us only $P(N_I|N)$, to estimate, but we really don't care what $P(N_I|N)$ is, because knowing that would still leave us wondering what threshold to set on $P(N_I|s, N)$. We don't care about classification accuracy; we care only about the quality of the resulting language model, so we might as well just attempt to find a threshold on $P(s|I)/P(s|N)$ that optimizes the fit of the resulting language model to held-out in-domain data.

Equivalently, we can work in the log domain with the quantity $\log(P(s|I)) - \log(P(s|N))$. This gets us very close to working with the difference in cross-entropies, because $H_I(s) - H_N(s)$ is just a length-normalized version of $\log(P(s|I)) - \log(P(s|N))$, with the sign reversed. The reason that we need to normalize for length is that the value of $\log(P(s|I)) - \log(P(s|N))$ tends to correlate very strongly with text segment length. If the candidate text segments vary greatly in length—e.g., if we partition N into sentences—this correlation can be a serious problem.

We estimated this effect on a 1000-sentence sample of our experimental data described below, and found the correlation between sentence log probability difference and sentence length to be $r = -0.92$, while the cross-entropy difference was almost uncorrelated with sentence length ($r = 0.04$). Hence, using sentence probability ratios or log probability differences as our scoring function would result in selecting disproportionately very short sentences. We tested this in an experiment not described here in detail, and found it not to be significantly better as a selection criterion than random selection.

Corpus	Sentence count	Token count
Gigaword	133,310,562	3,445,946,266
Europarl train	1,651,392	48,230,859
Europarl test	2,000	55,566

Table 1: Corpus size statistics

3 Experiments

We have empirically evaluated our proposed method for selecting data from a non-domain-specific source to model text in a specific domain. For the in-domain corpus, we chose the English side of the English-French parallel text from release v5 of the Europarl corpus (Koehn, 2005). This consists of proceedings of the European Parliament from 1999 through 2009. We used the text from 1999 through 2008 as in-domain training data, and we used the first 2000 sentences from January 2009 as test data. For the non-domain-specific corpus, we used the LDC English Gigaword Third Edition (LDC Catalog No.: LDC2007T07).

We used a simple tokenization scheme on all data, splitting on white space and on boundaries between alphanumeric and nonalphanumeric (e.g., punctuation) characters. With this tokenization, the sizes of our data sets in terms of sentences and tokens are shown in Table 1. The token counts include added end-of-sentence tokens.

To implement our data selection method we required one language model trained on the Europarl training data and one trained on the Gigaword data. To make these language models comparable, and to show the feasibility of optimizing the fit to the in-domain data without training a model on the entire Gigaword corpus, we trained the Gigaword language model for data selection on a random sample of the Gigaword corpus of a similar size to that of the Europarl training data: 1,874,051 sentences, 48,459,945 tokens.

To further increase the comparability of these Europarl and Gigaword language models, we restricted the vocabulary of both models to the tokens appearing at least twice in the Europarl training data, treating all other tokens as instances of <UNK>. With this vocabulary, 4-gram language models were trained on both the Europarl training data and the Gigaword random sample using backoff absolute discounting (Ney et al. 1994), with a discount of 0.7 used for all N-gram lengths. The discounted probability mass at the unigram level was added to the probability of <UNK>. A count cutoff of 2 occurrences was applied to the trigrams and 4-grams in estimating these models.

We computed the cross-entropy of each sentence in the Gigaword corpus according to both models, and scored each sentence by the difference in cross-entropy, $H_{Ep}(s) - H_{Gw}(s)$. We then selected subsets of the Gigaword data corresponding to 8 cutoff points in the cross-entropy difference scores, and trained 4-gram models (again using absolute discounting with a discount of 0.7) on each of these subsets and on the full Gigaword corpus. These language models were estimated without restricting the vocabulary or applying count cutoffs, but the only parameters computed were those needed to determine the perplexity of the held-out Europarl test set, which saves a substantial amount of computation in determining the optimal selection threshold.

We compared our selection method to three other methods. As a baseline, we trained language models on random subsets of the Gigaword corpus of approximately equal size to the data sets produced by the cutoffs we selected for the cross-entropy difference scores. Next, we scored all the Gigaword sentences by the cross-entropy according to the Europarl-trained model alone. As we noted above, this is equivalent to the in-domain perplexity scoring method used by Lin et al. (1997) and Gao et al. (2002). Finally, we implemented Klakow's (2000) method, scoring each Gigaword sentence by removing it from the Gigaword corpus and computing the difference in the log likelihood of the Europarl corpus according to unigram models trained on the Gigaword corpus with and without that sentence. With the latter two methods, we chose cutoff points in the resulting scores to produce data sets approximately equal in size to those obtained using our selection method.

4 Results

For all four selection methods, plots of test set perplexity vs. the number of training data tokens selected are displayed in Figure 1. (Note that the training data token counts are displayed on a logarithmic scale.) The test set perplexity for the language model trained on the full Gigaword corpus is 135. As we might expect, reducing training data by random sampling always increases perplexity. Selecting Gigaword sentences by their

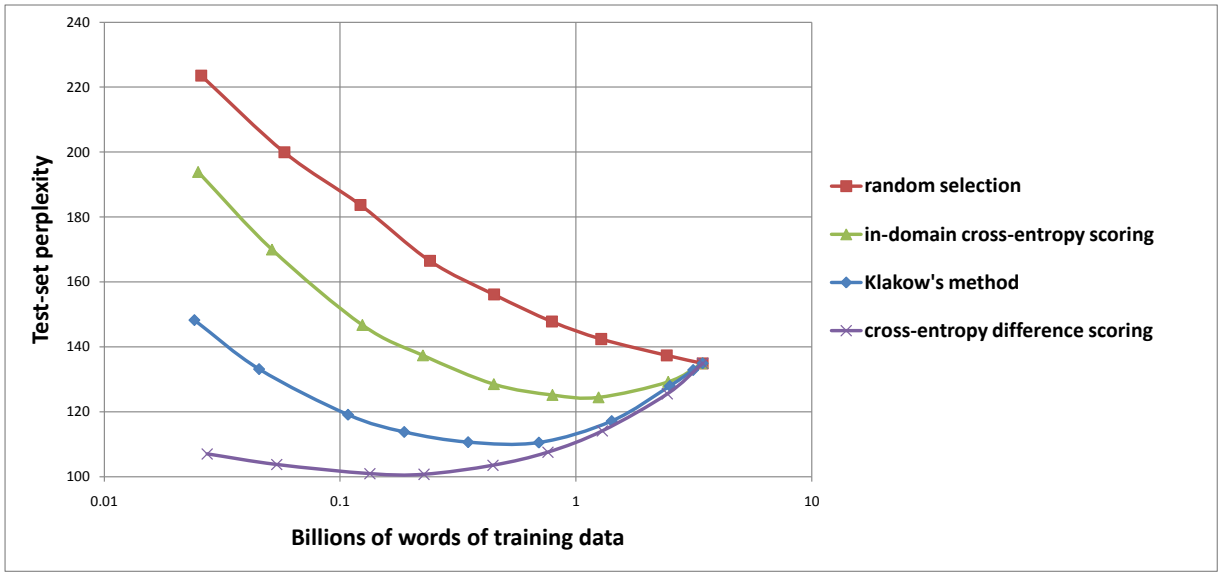

Figure 1: Test set perplexity vs. training set size

Selection Method	Original LM PPL	Modified LM PPL
in-domain cross-entropy scoring	124.4	124.8
Klakow's method	110.5	110.8
cross-entropy difference scoring	100.7	101.9

Table 2: Results adjusted for vocabulary coverage

cross-entropy according to the Europarl-trained model is effective in reducing both test set perplexity and training corpus size, with an optimum perplexity of 124, obtained with a model built from 36% of the Gigaword corpus. Klakow's method is even more effective, with an optimum perplexity of 111, obtained with a model built from 21% of the Gigaword corpus. The cross-entropy difference selection method, however, is yet more effective, with an optimum perplexity of 101, obtained with a model built from less than 7% of the Gigaword corpus.

The comparisons implied by Figure 1, however, are only approximate, because each perplexity (even along the same curve) is computed with respect to a different vocabulary, resulting in a different out-of-vocabulary (OOV) rate. OOV tokens in the test data are excluded from the perplexity computation, so the perplexity measurements are not strictly comparable.

Out of the 55566 test set tokens, the number of OOV tokens ranges from 418 (0.75%), for the smallest training set based on in-domain cross-entropy scoring, to 20 (0.03%), for training on the full Gigaword corpus. If we consider only the training sets that appear to produce the lowest perplexity for each selection method, however, the spread of OOV counts is much narrower, ranging 53 (0.10%) for best training set based on cross-entropy difference scoring, to 20 (0.03%), for random selection.

To control for the difference in vocabulary, we estimated a modified 4-gram language model for each selection method (other than random selection) using the training set that appeared to produce the lowest perplexity for that selection method in our initial experiments. In the modified language models, the unigram model based on the selected training set is smoothed by absolute discounting, and backed-off to an unsmoothed unigram model based on the full Gigaword corpus. This produces language models that are normalized over the same vocabulary as a model trained on the full Gigaword corpus; thus the test set has the same OOVs for each model.

Test set perplexity for each of these modifed language models is compared to that of the original version of the model in Table 2. It can be seen that adjusting the vocabulary in this way, so that all models are based on the same vocabulary,

yields only very small changes in the measured test-set perplexity, and these differences are much smaller than the differences between the different selection methods, whichever way the vocabulary of the language models is determined.

5 Conclusions

The cross-entropy difference selection method introduced here seems to produce language models that are both a better match to texts in a restricted domain, and require less data for training, than any of the other data selection methods tested. This study is preliminary, however, in that we have not yet shown improved end-to-end task performance applying this approach, such as improved BLEU scores in a machine translation task. However, we believe there is reason to be optimistic about this. When a language model trained on non-domain-specific data is used in a statistical translation model as a separate feature function (as is often the case), lower perplexity on in-domain target language test data derived from reference translations corresponds directly to assigning higher language model feature scores to those reference translations, which should in turn lead to translation system output that matches reference translations better.

References

Thorsten Brants, Ashok C. Popat, Peng Xu, Franz J. Och, and Jeffrey Dean. 2007. Large language models in machine translation. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, June 28–30, Prague, Czech Republic, 858–867.

François Denis, Remi Gilleron, and Marc Tommasi. 2002. Text classification from positive and unlabeled examples. In *The 9th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU 2002)*, 1927–1934.

Charles Elkin and Keith Noto. 2008. Learning classifiers from only positive and unlabeled data. In *KDD 2008*, August 24–27, Las Vegas, Nevada, USA, 213–220.

Jianfeng Gao, Joshua Goodman, Mingjing Li, and Kai-Fu Lee. 2002. Toward a unified approach to statistical language modeling for Chinese. *ACM Transactions on Asian Language Information Processing*, 1(1):3–33.

Dietrich Klakow. 2000. Selecting articles from the language model training corpus. In *ICASSP 2000*, June 5–9, Istanbul, Turkey, vol. 3, 1695–1698.

Philipp Koehn. 2005. Europarl: a parallel corpus for statistical machine translation. In *MT Summit X*, September 12–16, Phuket, Thailand, 79–86.

Sung-Chien Lin, Chi-Lung Tsai, Lee-Feng Chien, Ker-Jiann Chen, and Lin-Shan Lee. 1997. Chinese language model adaptation based on document classification and multiple domain-specific language models. In *EUROSPEECH-1997*, 1463–1466.

Hermann Ney, Ute Essen, and Reinhard Kneser. 1994. On structuring dependencies in stochastic language modelling. *Computer Speech and Language*, 8:1–38.