Distributed Language Modeling for N-best List Re-ranking

Ying Zhang **Almut Silja Hildebrand** **Stephan Vogel**
Language Technologies Institute, Carnegie Mellon University
5000 Forbes Ave. Pittsburgh, PA 15213, U.S.A.
{joy+, silja+, vogel+}@cs.cmu.edu

Abstract

In this paper we describe a novel distributed language model for N-best list re-ranking. The model is based on the client/server paradigm where each server hosts a portion of the data and provides information to the client. This model allows for using an arbitrarily large corpus in a very efficient way. It also provides a natural platform for relevance weighting and selection. We applied this model on a 2.97 billion-word corpus and re-ranked the N-best list from Hiero, a state-of-the-art phrase-based system. Using BLEU as a metric, the re-ranked translation achieves a relative improvement of 4.8%, significantly better than the model-best translation.

1 Introduction

Statistical language modeling has been widely used in natural language processing applications such as Automatic Speech Recognition (ASR), Statistical Machine Translation (SMT) (Brown et al., 1993) and Information Retrieval (IR) (Ponte and Croft, 1998).

Conventional n-gram language modeling counts the frequency of all the n-grams in a corpus and calculates the conditional probabilities of a word given its history of $n-1$ words $P(w_i|w_{i-n+1}^{i-1})$. As the corpus size increases, building a high order language model offline becomes very expensive if it is still possible (Goodman, 2000).

In this paper, we describe a new approach of language modeling using a distributed computing paradigm. Distributed language modeling can make use of arbitrarily large training corpora and provides a natural way for language model adaptation.

We applied the distributed LM to the task of re-ranking the N-best list in statistical machine translation and achieved significantly better translation quality when measured by the BLEU metric (Papineni et al., 2001).

2 N-best list re-ranking

When translating a source language sentence $\mathbf{f}$ into English, the SMT decoder first builds a translation lattice over the source words by applying the translation model and then explores the lattice and searches for an optimal path as the best translation. The decoder uses different models, such as the translation model, n-gram language model, fertility model, and combines multiple model scores to calculate the objective function value which favors one translation hypothesis over the other (Och et al., 2004).

Instead of outputting the top hypothesis $\mathbf{e}^{(1)}$ based on the decoder model, the decoder can output N (usually $N = 1000$) alternative hypotheses $\{\mathbf{e}^{(r)}|r = 1, \ldots, N\}$ for one source sentence and rank them according to their model scores.

Figure 1 shows an example of the output from a SMT system. In this example, alternative hypothesis $\mathbf{e}^{(2)}$ is a better translations than $\mathbf{e}^{(1)}$ according to the reference (Ref) although its model score is lower.

SMT models are not perfect, it is unavoidable to have a sub-optimal translation output as the model-best by the decoder. The objective of N-best list re-ranking is then to re-rank the translation hypotheses using features which are not used during decoding so that better translations can emerge as "optimal" translations. Our exper-

Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP 2006), pages 216–223,
Sydney, July 2006.

f: 自从 2001 年 美国 遭受 恐怖 攻击 的 事件 之后

Ref: Since the terrorist attacks on the United States in 2001

$\mathbf{e}^{(1)}$: since 200 year , the united states after the terrorist attacks in the incident

$\mathbf{e}^{(2)}$: since 2001 after the incident of the terrorist attacks on the united states

$\mathbf{e}^{(3)}$: since the united states 2001 threats of terrorist attacks after the incident

$\mathbf{e}^{(4)}$: since 2001 the terrorist attacks after the incident

$\mathbf{e}^{(5)}$: since 200 year , the united states after the terrorist attacks in the incident

Figure 1: An example of N-best list.

iments (section 5.1) have shown that the oracle-best translation from a typical N-best list could be 6 to 10 BLEU points better than the model-best translation.

In this paper we use the distributed language model on very large data to re-rank the N-best list.

2.1 Sentence likelihood

The goal of a language model is to determine the probability, or in general the "likelihood" of a word sequence $w_1 \dots w_m$ (w_1^m for short) given some training data. The standard language modeling approach breaks the sentence probability down into:

$$P(w_1^m) = \prod_i P(w_i|w_1^{i-1}) \tag{1}$$

Under the Markov or higher order Markov process assumption that only the closest $n-1$ words have real impact on the choice of w_i, equation 1 is approximated to:

$$P(w_1^m) = \prod_i P(w_i|w_{i-n+1}^{i-1}) \tag{2}$$

The probability of a word given its history can be approximated with the maximum likelihood estimate (MLE) without any smoothing:

$$P(w_i|w_{i-n+1}^{i-1}) \approx \frac{C(w_{i-n+1}^{i})}{C(w_{i-n+1}^{i-1})} \tag{3}$$

In addition to the standard n-gram probability estimation, we propose 3 sentence likelihood metrics.

- L_0: Number of n-grams matched.

  The simplest metric for sentence likelihood is to count how many n-grams in this sentence can be found in the corpus.

  $$L_0(w_1^m) = \sum_{\substack{i,j \\ i \le j}} \delta(w_i^j) \tag{4}$$

  $$\delta(w_i^j) = \begin{cases} 1 & : \quad C(w_i^j) > 0 \\ 0 & : \quad C(w_i^j) = 0 \end{cases} \tag{5}$$

  For example, L_0 for sentence in figure 2 is 52 because 52 n-grams have non-zero counts.

- L_1^n: Average interpolated n-gram conditional probability.

  $$L_1^n(w_1^m) = \left(\prod_{i=1}^{m} \sum_{k=1}^{n} \lambda_k P(w_i|w_{i-k+1}^{i-1}) \right)^{\frac{1}{m}} \tag{6}$$

  $P(w_i|w_{i-k+1}^{i-1})$ is approximated from the n-gram counts (Eq. 3) without any smoothing. λ_k is the weight for k-gram conditional probability, $\sum \lambda_k = 1$.

  L_1^n is similar to the standard n-gram LM except the probability is averaged over the words in the sentence to prevent shorter sentences being favored unfairly.

- L_2: Sum of n-gram's non-compositionality

  For each matched n-gram, we consider all the possibilities to cut/decompose it into two short n-grams, for example "the terrorist attacks on the united states" could be decomposed into ("the", "terrorist attacks on the united states") or ("the terrorist", "attacks on the united states"), ... , or ("the terrorist attacks on the united", "states"). For each cut, calculate the point-wise mutual information (PMI) between the two short n-grams. The one with the minimal PMI is the most "natural" cut for this n-gram. The PMI over the natural cut quantifies the *non-compositionality* I_{nc} of an n-gram w_i^j. The higher the value of $I_{nc}(w_i^j)$ the more likely w_i^j is a meaningful constituent, in other words, it is less likely that w_i^j is composed from two short n-grams just by chance (Yamamoto and Church, 2001).

  Define L_2 formally as:

  $$L_2(w_1^m) = \sum_{\substack{i,j \\ i \le j}} I_{nc}(w_i^j) \tag{7}$$

$$I_{nc}(w_i^j) = \begin{cases} \min\limits_k I(w_i^k; w_{k+1}^j) & : C(w_i^j) > 0 \\ 0 & : C(w_i^j) = 0 \end{cases} \quad (8)$$

$$I(w_i^k; w_{k+1}^j) = log \frac{P(w_i^j)}{P(w_i^k)P(w_{k+1}^j)} \quad (9)$$

3 Distributed language model

The fundamental information required to calculate the likelihood of a sentence is the frequency of n-grams in the corpus. In conventional LM training, all the counts are collected from the corpus $\mathcal{D}$ and saved to disk for probability estimation. When the size of $\mathcal{D}$ becomes large and/or n is increased to capture more context, the count file can be too large to be processed.

Instead of collecting n-gram counts offline, we index $\mathcal{D}$ using a suffix array (Manber and Myers, 1993) and count the occurrences of w_{i-n+1}^i in $\mathcal{D}$ on the fly.

3.1 Calculate n-gram frequency using suffix array

For a corpus $\mathcal{D}$ with $\mathcal{N}$ words, locating all the occurrences of w_{i-n+1}^i takes $O(log\mathcal{N})$. Zhang and Vogel (2005) introduce a search algorithm which locates all the $m(m+1)/2$ embedded n-grams in a sentence of m words within $O(m \cdot log\mathcal{N})$ time.

Figure 2 shows the frequencies of all the embedded n-grams in sentence "since 2001 after the incident of the terrorist attacks on the united states" matched against a 26 million words corpus. For example, unigram "after" occurs 4.43×10^4 times, trigram "after the incident" occurs 106 times. The longest n-gram that can be matched is the 8-gram "of the terrorist attacks on the united states" which occurs 7 times in the corpus.

3.2 Client/Server paradigm

To load the corpus and its suffix array index into the memory, each word token needs 8 bytes. For example, if the corpus has 50 million words, 400MB memory is required. For the English[1] GigaWord[2] corpus which has 2.7 billion words, the total memory required is 22GB. It is practically impossible to fit such data into the memory of any single machine.

To make use of the large amount of data, we developed a distributed client/server architecture for language modeling. Client/server is the most common paradigm of distributed computing at present (Leopold, 2001). The paradigm describes an asymmetric relationship between two type of processes, of which one is the client, and the other is the server. The server process manages some resources and offers a service which can be used by other processes. The client is a process that needs the service in order to accomplish its task. It sends a request to the server and asks for the execution of a task that is covered by the service.

We split the large corpus $\mathcal{D}$ into d non-overlapping chunks. One can easily verify that for any n-gram w_{i-n+1}^i the count of its occurrences in $\mathcal{D}$ is the sum of its occurrences in all the chunks, i.e.,

$$C(w_{i-n+1}^i)|\mathcal{D} = \sum_d C(w_{i-n+1}^i)|\mathcal{D}_d \quad (10)$$

Each server[3] loads one chunk of the corpus with its suffix array index. The client sends an English sentence $w_1 \ldots w_m$ to each of the servers and requests for the count information of all the n-grams in the sentence. The client collects the count information from all the servers, sums up the counts for each n-gram and then calculates the likelihood of the sentence.

The client communicates with the servers via TCP/IP sockets. In our experiments, we used 150 servers running on 26 computers to serve one client. Multiple clients can be served at the same time if needed. The process of collecting counts and calculating the sentence probabilities takes about 1 to 2 ms for each English sentence (average length 23.5 words). With this architecture, we can easily make use of larger corpora by adding additional data servers. In our experiments, we used all the 2.7 billion word data in the English Gigaword corpus without any technical difficulties.

[1] Though we used English data for our experiments in this paper, the approach described here is language independent.

[2] http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2005T12

[3] A server is a special program that provides services to client processes. It runs on a physical computer but the concept of server should not be confused with the actual machine that runs it. In practice, one computer usually hosts multiple servers at the same time.

n	since	2001	after	the	incident	of	the	terrorist	attacks	on	the	united	states
1	2.19×10^4	7559	4.43×10^4	1.67×10^6	2989	6.9×10^5	1.67×10^6	6160	9278	2.7×10^5	1.67×10^6	5.1×10^4	3.78×10^4
2		165	105	1.19×10^4	1892	34	2.07×10^5	807	1398	1656	5.64×10^4	3.72×10^4	3.29×10^4
3			6	56	106	6	3	162	181	216	545	605	2.58×10^4
4				0	0	0	1	0	35	67	111	239	424
5					0	0	0	0	0	15	34	77	232
6						0	0	0	0	0	10	23	76
7							0	0	0	0	0	7	23
8								0	0	0	0	0	7

Figure 2: Frequencies of all the embedded n-grams in sentence "since 2001 after the incident of the terrorist attacks on the united states."

4 "More data is better data" or "Relevant data is better data"

Although statistical systems usually improve with more data, performance can decrease if additional data does not fit the test data. There have been debates in the data-driven NLP community as to whether "more data is better data" or "relevant data is better data". For N-best list re-ranking, the question becomes: "should we use all the data to re-rank the hypotheses for one source sentence, or select some corpus chunks that are believed to be relevant to this sentence?"

Various relevance measures are proposed in (Iyer and Ostendorf, 1999) including content-based relevance criteria and style-based criteria. In this paper, we use a very simple relevance metric. Define corpora $\mathcal{D}_d$'s relevance to a source sentence $\mathbf{f}_t$ as:

$$R(\mathcal{D}_d, \mathbf{f}_t) = \sum_{r=1}^{N} L_0(\mathbf{e}_t^{(r)})|\mathcal{D}_d \qquad (11)$$

$R(\mathcal{D}_d, \mathbf{f}_t)$ estimates how well a corpus $\mathcal{D}_d$ can cover the n-grams in the N-best list of a source sentence. The higher the coverage, the more relevant $\mathcal{D}_d$ is.

In the distributed LM architecture, the client first sends N translations of $\mathbf{f}_t$ to all the servers. From the returned n-gram matching information, client calculates $R(\mathcal{D}_d, \mathbf{f}_t)$ for each server, and choose the most relevant (e.g., 20) servers for $\mathbf{f}_t$. The n-gram counts returned from these relevant servers are summed up for calculating the likelihood of $\mathbf{f}_t$. One could also assign weights to the n-gram counts returned from different servers during the summation so that the relevant data has more impact than the less-relevant ones.

5 Experiments

We used the N-best list generated by the Hiero SMT system (Chiang, 2005). Hiero is a statistical phrase-based translation model that uses hierarchical phrases. The decoder uses a trigram language model trained with modified Kneser-Ney smoothing (Kneser and Ney, 1995) on a 200 million words corpus. The 1000-best list was generated on 919 sentences from the MT03 Chinese-English evaluation set.

All the data from the English Gigaword corpus plus the English side of the Chinese-English bilingual data available from LDC are used. The 2.97 billion words data is split into 150 chunks, each has about 20 million words. The original order is kept so that each chunk contains data from the same news source and a certain period of time. For example, chunk *Xinhua2003* has all the Xinhua News data from year 2003 and *NYT9499_038* has the last 20 million words from the New York Times 1994-1999 corpus. One could split the data into larger(smaller) chunks which will require less(more) servers. We choose 20 million words as the size for each chunk because it can be loaded by our smallest machine and it is a reasonable granularity for selection.

In total, 150 corpus information servers run on 26 machines connected by the standard Ethernet LAN. One client sends each English hypothesis translations to all 150 servers and uses the returned information to re-rank. The whole process takes about 600 seconds to finish.

We use BLEU scores to measure the translation accuracy. A bootstrapping method is used to calculate the 95% confidence intervals for BLEU (Koehn, 2004; Zhang and Vogel, 2004).

5.1 Oracle score of the N-best list

Because of the *spurious ambiguity*, there are only 24,612 unique hypotheses in the 1000-best list, on average 27 per source sentence. This limits the potential of N-best re-ranking. *Spurious ambiguity* is created by the decoder where two hypotheses generated from different decoding path are considered different even though they have identical word sequences. For example, "the terrorist attacks on the united states" could be the output of decoding path [the terrorist attacks][on the united

states] and [the terrorist attacks on] [the united states].

We first calculate the oracle score from the N-best list to verify that there are alternative hypotheses better than the model-best translation. The oracle best translations are created by selecting the hypothesis which has the highest sentence BLEU score for each source sentence. Yet a critical problem with BLEU score is that it is a function of the entire test set and does not give meaningful scores for single sentences. We followed the approximation described in (Collins et al., 2005) to get around this problem. Given a test set with T sentences, N hypotheses are generated for each source sentence $\mathbf{f}_t$. Denote $\mathbf{e}_t^{(r)}$ as the r-th ranked hypothesis for $\mathbf{f}_t$. $\mathbf{e}_t^{(1)}$ is the model-best hypothesis for this sentence. The baseline BLEU scores are calculated based on the model-best translation set $\{\mathbf{e}_t^{(1)}|t = 1, \ldots, T\}$.

Define the BLEU sentence-level gain for $\mathbf{e}_t^{(r)}$ as:

$$\begin{aligned} G_{BLEU}\mathbf{e}_t^{(r)} = & \\ & BLEU\{\mathbf{e}_1^{(1)}, \mathbf{e}_2^{(1)}, \ldots, \mathbf{e}_t^{(r)}, \ldots, \mathbf{e}_T^{(r)}\} \\ - \quad & BLEU\{\mathbf{e}_1^{(1)}, \mathbf{e}_2^{(1)}, \ldots, \mathbf{e}_t^{(1)}, \ldots, \mathbf{e}_T^{(r)}\} \end{aligned}$$

$G_{BLEU}\mathbf{e}_t^{(r)}$ calculates the gain if we switch the model-best hypothesis $\mathbf{e}_t^{(1)}$ using $\mathbf{e}_t^{(r)}$ for sentence $\mathbf{f}_t$ and keep the translations for the rest of the test set untouched.

With the estimated sentence level gain for each hypothesis, we can construct the oracle best translation set by selecting the hypotheses with the highest BLEU gain for each sentence. Oracle best BLEU translation set is: $\{\mathbf{e}_t^{(r_t^*)}|t = 1, \ldots, T\}$ where $r_t^* = \arg\max_r G_{BLEU}\mathbf{e}_t^{(r)}$.

	Model-best		Oracle
	Score	Confidence Interval	
BLEU	31.44	[30.49, 32.33]	37.48

Table 1: BLEU scores for the model-best and oracle-best translations.

Table 1 shows the BLEU score of the approximated oracle best translation. The oracle score is 7 points higher than the model-best scores even though there are only 27 unique hypotheses for each sentence on average. This confirms our observation that there are indeed better translations in the N-best list.

5.2 Training standard n-gram LM on large data for comparison

Besides comparing the distributed language model re-ranked translations with the model-best translations, we also want to compare the distributed LM with the the standard 3-gram and 4-gram language models on the N-best list re-ranking task.

Training a standard n-gram model for a 2.9 billion words corpora is much more complicated and tedious than setting up the distributed LM. Because of the huge size of the corpora, we could only manage to train a test-set specific n-gram LM for this experiment.

First, we split the corpora into smaller chunks and generate n-gram count files for each chunk. Each count file is then sub-sampled to entries where all the words are listed in the vocabulary of the N-best list (5,522 word types). We merge all the sub-sampled count files into one and train the standard language model based on it.

We manage to train a 3-gram LM using the 2.97 billion-word corpus. Resulting LM requires 2.3GB memory to be loaded for the re-ranking experiment.

A 4-gram LM for this N-best list is of 13 GB in size and can not be fit into the memory. We split the N-best list into 9 parts to reduce the vocabulary size of each sub N-best list to be around 1000 words. The 4-gram LM tailored for each sub N-best list is around 1.5 to 2 GB in size.

Training higher order standard n-gram LMs with this method requires even more partitions of the N-best list to get smaller vocabularies. When the vocabulary becomes too small, the smoothing could fail and results in unreliable LM probabilities.

Adapting the standard n-gram LM for each individual source sentence is almost infeasible given our limited computing resources. Thus we do not have equivalent n-gram LMs to be compared with the distributed LM for conditions where the most relevant data chunks are used to re-rank the N-best list for a particular source sentence.

5.3 Results

Table 2 lists results of the re-ranking experiments under different conditions. The re-ranked translation improved the BLEU score from 31.44 to

32.64, significantly better than the model-best translation.

Different metrics are used under the same data situation for comparison. L_0, though extremely simple, gives quite nice results on N-best list re-ranking. With only one corpus chunk (the most relevant one) for each source sentence, L_0 improved the BLEU score to 32.22. We suspect that L_0 works well because it is inline with the nature of BLEU score. BLEU measures the similarity between the translation hypothesis and human reference by counting how many n-grams in MT can be found in the references.

Instead of assigning weights 1 to all the matched n-grams in L_0, L_2 weights each n-gram by its *non-compositionality*. For all data conditions, L_2 consistently gives the best results.

Metric family L_1 is close to the standard n-gram LM probability estimation. Because no smoothing is used, L_1^3 performance (32.00) is slightly worse than the standard 3-gram LM result (32.22). On the other hand, increasing the length of the history in L_1 generally improves the performance.

Figure 3 shows the BLEU score of the re-ranked translation when using different numbers of relevant data chunks for each sentence. The selected data chunks may differ for each sentences. For example, the 2 most relevant corpora for sentence 1 are *Xinhua2002* and *Xinhua2003* while for sentence 2 *APW2003A* and *NYT2002D* are more relevant. When we use the most relevant data chunk (about 20 million words) to re-rank the N-best list, 36 chunks of data will be used at least once for 919 different sentences, which accounts for about 720 million words in total. Thus the x-axis in figure 3 should not be interpreted as the total amount of data used but the number of the most relevant corpora used for each sentence.

All three metrics in figure 3 show that using all data together (150 chunks, 2.97 billion words) does not give better discriminative powers than using only some relevant chunks. This supports our argument in section 4 that relevance selection is helpful in N-best list re-ranking. In some cases the re-ranked N-best list has a higher BLEU score after adding a supposedly "less-relevant" corpus chunk and a lower BLEU score after adding a "more-relevant" chunk. This indicates that the relevance measurement (Eq. 11) is not fully reflecting the real "relevance" of a data chunk for a sentence. With a better relevance measurement one would expect to see the curves in figure 3 to be much smoother.

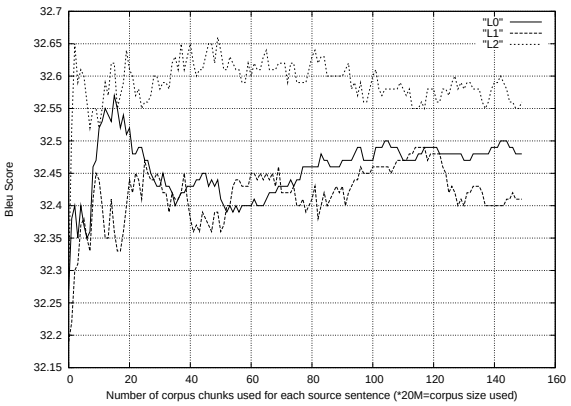

Figure 3: BLEU score of the re-ranked best hypothesis vs. the number of the most relevant corpus chunks used to re-rank the n-best list for each sentences. L_0: number of n-grams matched; L_1: average interpolated n-gram conditional probability; L_2: sum of n-grams' non-compositionality.

6 Related work and discussion

Yamamoto and Church (2001) used suffix arrays to compute the frequency and location of an n-gram in a corpus. The frequencies are used to find "interesting" substrings which have high mutual information.

Soricut et al. (2002) build a Finite State Acceptor (FSA) to compactly represent all possible English translations of a source sentence according to the translation model. All sentences in a big monolingual English corpus are then scanned by this FSA and those accepted by the FSA are considered as possible translations for the source sentence. The corpus is split into hundreds of chunks for parallel processing. All the sentences in one chunk are scanned by the FSA on one processor. Matched sentences from all chunks are then used together as possible translations. The assumption of this work that possible translations of a source sentence can be found as exact match in a big monolingual corpus is weak even for very large corpus. This method can easily fail to find any possible translation and return zero proposed translations.

Kirchhoff and Yang (2005) used a factored 3-gram model and a 4-gram LM (modified KN smoothing) together with seven system scores to re-rank an SMT N-best. They improved the translation quality of their best baseline (Spanish-

# of Relevant Chunks per. Sent	1	2	5	10	20	150
3-gram KN	32.22					32.08
4-gram KN	32.22					32.53
L_0	32.27	32.38	32.40	32.47	32.51	32.48
L_1^3	32.00	32.14	32.14	32.15	32.16	
L_1^4	32.18	32.36	32.28	32.44	32.41	
L_1^5	32.21	32.33	32.35	32.41	32.37	
L_1^6	32.19	32.22	32.37	32.45	32.40	32.41
L_1^7	32.22	32.29	32.37	32.44	32.40	
L_2	32.29	32.52	**32.61**	32.55	**32.64**	32.56

Table 2: BLEU scores of the re-ranked translations. Baseline score = 31.44

English) from BLEU 30.5 to BLEU 31.0.

Iyer and Ostendorf (1999) select and weight data to train language modeling for ASR. The data is selected based on its relevance for a topic or the similarity to data known to be in the same domain as the test data. Each additional document is classified to be in-domain or out-of-domain according to cosine distance with TF-IDF term weights, POS-tag LM and a 3-gram word LM. n-gram counts from the in-domain and the additionally selected out-of-domain data are then combined with an weighting factor. The combined counts are used to estimate a LM with standard smoothing.

Hildebrand et al. (2005) use information retrieval to select relevant data to train adapted translation and language models for an SMT system.

Si et al. (2002) use unigram distribution similarity to select the document collection which is most relevant to the query documents. Their work is mainly focused on information retrieval application.

7 Conclusion and future work

In this paper, we presented a novel distributed language modeling solution. The distributed LM is capable of using an arbitrarily large corpus to estimate the n-gram probability for arbitrarily long histories. We applied the distributed language model to N-best re-ranking and improved the translation quality by 4.8% when evaluated by the BLEU metric. The distributed LM provides a flexible architecture for relevance selection, which makes it possible to select data for each individual test sentence. Our experiments have shown that relevant data has better discriminative power than using all the data.

We will investigate different relevance weighting schemes to better combine n-gram statistics from different data sources. We are planning to integrate the distributed LM in the statistical machine translation decoder in the near future.

8 Acknowledgement

We would like to thank Necip Fazil Ayan and Philip Resnik for providing Hiero system's N-best list and allowing us to use it for this work.

References

Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: parameter estimation. *Comput. Linguist.*, 19(2):263–311.

David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of ACL 2005*, pages 263–270, Ann Arbor, MI, June 2005. ACL.

Michael Collins, Philipp Koehn, and Ivona Kucerova. 2005. Clause restructuring for statistical machine translation. In *Proceedings of ACL 2005*, pages 531–540, Ann Arbor, MI, June.

J. Goodman. 2000. A bit of progress in language modeling. Technical report, Microsoft Research, 56 Fuchun Peng.

Almut Silja Hildebrand, Matthias Eck, Stephan Vogel, and Alex Waibel. 2005. Adaptation of the translation model for statistical machine translation based on information retrieval. In *Proceedings of the 10th EAMT conference "Practical applications of machine translation"*, pages 133–142, Budapest, May.

R. Iyer and M. Ostendorf. 1999. Relevance weighting for combining multi-domain data for n-gram language modeling. *Comptuer Speech and Language*, 13(3):267–282.

Katrin Kirchhoff and Mei Yang. 2005. Improved language modeling for statistical machine translation. In *Proceedings of the ACL Workshop on Building and Using Parallel Texts*, pages 125–128, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Reinhard Kneser and Hermann Ney. 1995. Improved backing-off for m-gram language modeling. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, volume 1*, pages 181–184.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of EMNLP 2004*, Barcelona, Spain, July.

Claudia Leopold. 2001. *Parallel and Distributed Computing: A Survey of Models, Paradigms and Approaches*. John Wiley & Sons, Inc., New York, NY, USA.

Udi Manber and Gene Myers. 1993. Suffix arrays: a new method for on-line string searches. *SIAM J. Comput.*, 22(5):935–948.

Franz Josef Och, Daniel Gildea, Sanjeev Khudanpur, Anoop Sarkar, Kenji Yamada, Alex Fraser, Shankar Kumar, Libin Shen, David Smith, Katherine Eng, Viren Jain, Zhen Jin, and Dragomir Radev. 2004. A smorgasbord of features for statistical machine translation. In *Proceedings of the 2004 Meeting of the North American chapter of the Association for Computational Linguistics (NAACL-04)*, Boston.

K. Papineni, S. Roukos, T. Ward, and W. Zhu. 2001. Bleu: a method for automatic evaluation of machine translation. Technical Report RC22176(W0109-022), IBM Research Division, Thomas J. Watson Research Center.

Jay M. Ponte and W. Bruce Croft. 1998. A language modeling approach to information retrieval. In *Research and Development in Information Retrieval*, pages 275–281.

Luo Si, Rong Jin, Jamie Callan, and Paul Ogilvie. 2002. A language modeling framework for resource selection and results merging. In *CIKM '02: Proceedings of the eleventh international conference on Information and knowledge management*, pages 391–397, New York, NY, USA. ACM Press.

Radu Soricut, Kevin Knight, and Daniel Marcu. 2002. Using a large monolingual corpus to improve translation accuracy. In *AMTA '02: Proceedings of the 5th Conference of the Association for Machine Translation in the Americas on Machine Translation: From Research to Real Users*, pages 155–164, London, UK. Springer-Verlag.

Mikio Yamamoto and Kenneth W. Church. 2001. Using suffix arrays to compute term frequency and document frequency for all substrings in a corpus. *Comput. Linguist.*, 27(1):1–30.

Ying Zhang and Stephan Vogel. 2004. Measuring confidence intervals for the machine translation evaluation metrics. In *Proceedings of The 10th International Conference on Theoretical and Methodological Issues in Machine Translation*, October.

Ying Zhang and Stephan Vogel. 2005. An efficient phrase-to-phrase alignment model for arbitrarily long phrase and large corpora. In *Proceedings of the Tenth Conference of the European Association for Machine Translation (EAMT-05)*, Budapest, Hungary, May. The European Association for Machine Translation.