QUERY LANGUAGE MODELING FOR VOICE SEARCH

C. Chelba, J. Schalkwyk, T. Brants, V. Ha, B. Harb, W. Neveitt, C. Parada, P. Xu*

Google, Inc.,
1600 Amphiteatre Pkwy,
Mountain View, CA 94043, USA

ABSTRACT

The paper presents an empirical exploration of `google.com` query stream language modeling. We describe the normalization of the typed query stream resulting in out-of-vocabulary (OoV) rates below 1% for a one million word vocabulary. We present a comprehensive set of experiments that guided the design decisions for a voice search service. In the process we re-discovered a less known interaction between Kneser-Ney smoothing and entropy pruning, and found empirical evidence that hints at non-stationarity of the query stream, as well as strong dependence on various English locales—USA, Britain and Australia.

Index Terms— language modeling, voice search, query stream

1. INTRODUCTION

A typical voice search language model used in our system for the US English query stream is trained as follows:

- vocabulary size: 1M words, OoV rate 0.57%
- training data: 230B words, a random sample of anonymized queries that did not trigger spelling correction

The resulting size, as well as its performance on unseen query data (10k queries) when using Katz smoothing is shown in Table 1. We note a few key aspects:

- the first pass LM (15 million n-grams) requires very aggressive pruning—to about 0.1% of its unpruned size—in order to make it usable in static FST-based ASR decoders
- the perplexity hit taken by pruning the LM is significant, 50% relative; similarly, the 3-gram hit ratio is halved
- the impact on WER due to pruning is significant, yet lower in relative terms—10% relative, as we show in Section 6
- the unpruned model has excellent n-gram hit ratios on unseen test data: 77% for $n = 5$, and 97% for $n = 3$
- the choice of $n = 5$ is because using higher n-gram orders yields diminishing returns: a 7-gram LM is four times larger than the 5-gram LM trained from the same data and using the same vocabulary, at no gain in perplexity.

The paper attempts to explain our design choices. The next section describes the text normalization that allows us to use a one million word vocabulary and obtain out-of-vocabulary (OoV) rates lower than 1%, as well as the excellent n-gram hit ratios presented in Table 1.

The use of Katz smoothing in our language model is not accidental: we examined the interaction between Stolcke pruning and various n-gram LM smoothing techniques for aggressive pruning regimes which cut the original LM to under 1% of its original size. The main finding is that the increasingly popular family of Kneser-Ney [1] smoothing methods is in fact poorly suited for such aggressive pruning regimes, as explained in Section 3. When evaluated in terms of both perplexity and ASR word error rate, the more traditional ones, e.g. Katz/Good-Turing [2] perform significantly better after pruning. We wish to emphasize that this is not a new result, [3] also pointed out this behavior of Kneser-Ney models [1] and proposed a solution that alleviates the problem by growing the LM, instead of pruning it. [4] also suggests a variation of Kneser-Ney smoothing more suitable for pruning.

We then present experiments that show the temporal and spatial dependence of the English language models. Somewhat unexpectedly, using more training data does not result in an improved language model despite the fact that it is extremely well matched to the unseen test data. The English language models built from training data originating in three locales (USA, Britain, and Australia) exhibit strong locale-specific behavior, both in terms of perplexity and OoV rate.

The last section presents speech recognition experiments on a voice search test set. We conclude by highlighting our main findings.

2. TEXT NORMALIZATION

In order to build a language model for spoken query recognition we boot-strap from written queries to `google.com`. Written queries provide a data rich environment for modeling of queries. This requires robustly transforming written text into spoken form.

Table 2 lists a couple of example queries and their corresponding spoken equivalents. Written queries contain a fair number of cases which require special attention to convert to spoken form. Analyzing the top million vocabulary items before text normalization we see approximately 20% URLs and 20+% numeric items in the query stream. Without careful attention to text normalization the vocabulary of the system will grow substantially.

We adopt a finite state approach to text normalization. Let $T(written)$ be an acceptor that represents the written query. Conceptually the spoken form is computed as follows

$$T(spoken) = \text{bestpath}(T(written) \circ N(spoken))$$

where $N(spoken)$ represents the transduction from written to spoken form. Note that composition with $N(spoken)$ might introduce

*The author performed the work as a summer intern at Google, NYC. Her affiliation is with the CLSP, The Johns Hopkins University, 3400 N. Charles St., Baltimore, MD 21218, USA

[1] Alas, we were unaware of this at the time of our experiments and rediscovered this on our own.

Order	no. n-grams	pruning	PPL	n-gram hit-ratios
3	15M	entropy (Stolcke)	190	47/93/100
3	7.7B	none	132	97/99/100
5	12.7B	cut-off (1-1-2-2-2)	108	77/88/97/99/100

Table 1. Typical voice search LM, Katz smoothing: the LM is trained on 230 billion words using a vocabulary of 1 million words, achieving out-of-vocabulary rate of 0.57% on test data.

Written Query	**Spoken Query**
weather scarsdale, ny	weather scarsdale new york weather in scarsdale new york
bankofamerica.com	bank of america dot com
81 walker rd	eighty one walker rd
10:30am	ten thirty A M
at&t	A T and T
espn	E S P N

Table 2. Example written queries and their corresponding spoken form.

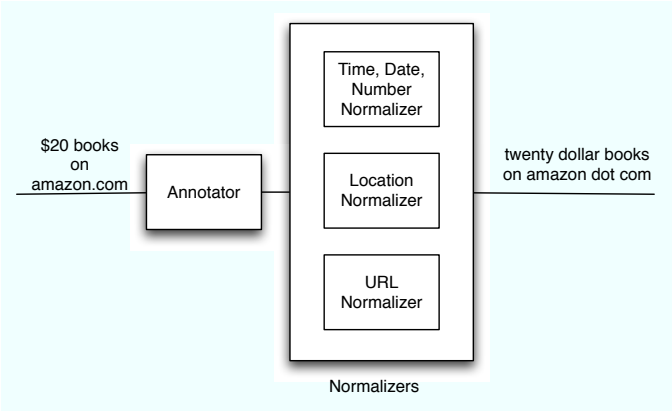

Fig. 1. Block diagram for context aware text normalization.

multiple alternate spoken representations of the input text. For the purpose of computing n-grams for spoken language modeling of queries we use the $\mathrm{bestpath}$ operation to select a single most likely interpretation.

The text normalization is run in multiple phases. Figure 1 depicts the text normalization process. In the first step we annotate the data. In this phase we categorize parts (sub strings) of queries into a set of known categories (e.g time, date, url, location).

Since the query is annotated, it is possible to perform context-aware normalization on the substrings. Each category has a corresponding text normalization transducer $N_{cat}(spoken)$ that is used to normalize the substring. Depending on the category we either use rule based approaches or a statistical approach to construct the text normalization transducer. For numeric categories like date, time and numbers it is easy enough to describe $N(spoken)$ using context dependent rewrite rules. For the URL normalizer $N_{url}(spoken)$ we train a statistical word decompounder that segments the string into its word constituents. For example, one reads the URL `cancercentersofamerica.com` as "cancer centers of america dot com". The URL decompounding transducer (decompounder) is built from the annotated data. Let Q be the set of queries in this table, and let U be the set of substrings of these queries that are labeled URLs.

For a string s of length k let $I(s)$ be the transducer that maps each character in s to itself; i.e., the i-th transition in $I(s)$ has input and output label $s(i)$. $I(s)$ represents the word segmented into characters. Further, let $T(s)$ be the transducer that maps the sequence of characters in s to s; i.e., the first transition in $T(s)$ has input $s(1)$ and output s, and the i-th transition, where $i \neq 1$, has input $s(i)$ and output ϵ. $T(s)$ represents the transduction of the spelled form of the word to the word itself. For a set of strings S, we define

$$T(S) = \bigoplus_{s \in S} T(s)$$

where $\bigoplus$ is the union operation on transducers. $T(S)$ therefore represents the transduction of the spelling of the word to the word itself for the whole vocabulary. Figure 2 illustrates the operation of $T(\cdot)$.

The queries in Q and their frequencies are used to train an LM L_{BASE}. Let V_{BASE} be its vocabulary. We build the decompounder as follows:

1. For each $u \in U$, define $N(u)$ as,

$$N(u) = \mathrm{bestpath}\left(I(u) \circ T^*(V_{\text{BASE}}) \circ L_{\text{BASE}}\right) \qquad (1)$$

   where '*' is the Kleene Closure, and '$\circ$' is the composition operator.

2. $N(U) = \bigoplus_{u \in U} N(u)$ is the URL decompounder.

The transducer $I(u) \circ T^*(V_{\text{BASE}})$ in (1) represents the lattice of all possible segmentations of u using the words in V_{BASE}, where each path from the start state to a final state in the transducer is a valid segmentation. The composition with the LM L_{BASE} scores every path. Finally, $N(u)$ is the path with the highest probability; i.e. the most likely segmentation.

As an example, Figure 3 depicts $I(u) \circ T^*(V_{\text{BASE}})$ for $u =$ *myspacelayouts*. Each path in this lattice is a valid decomposition, and in Table 3 we list a sample of these paths. After scoring all the paths via the composition with L_{BASE}, we choose the best path to represent the spoken form of the URL.

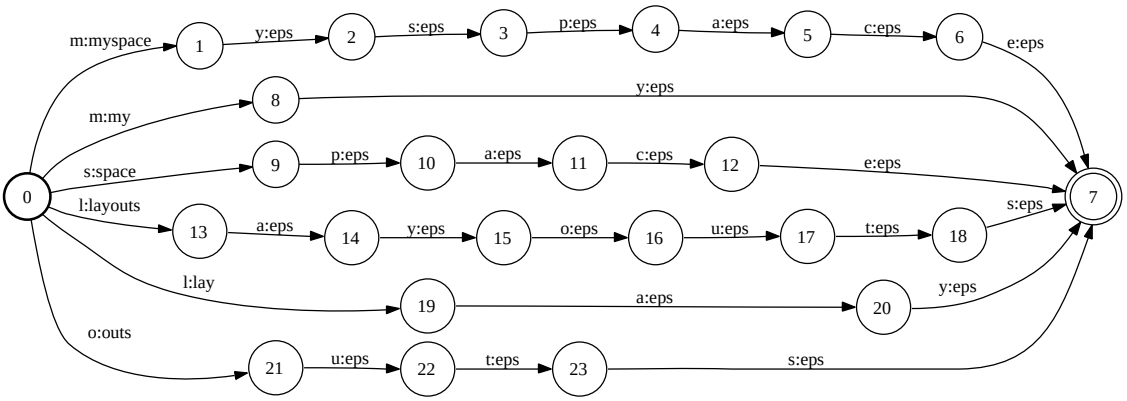

Fig. 2. $T(S)$ for the set of words $S = \{my, space, myspace, lay, outs, layouts\}$ where 'eps' denotes ϵ.

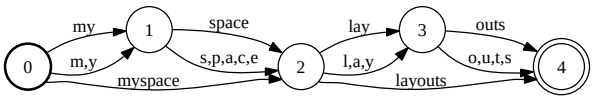

Fig. 3. The lattice $I(u) \circ T^*(V_{\text{BASE}})$ of all possible segmentations for $u =$ *myspacelayouts* using words in V_{BASE}.

Possible Segmentations
myspace layouts
my space layouts
my space lay outs
my space l a y outs

Table 3. Sample segmentations from Fig. 3. The one in bold represents the highest probability path as determined by the composition with L_{BASE}.

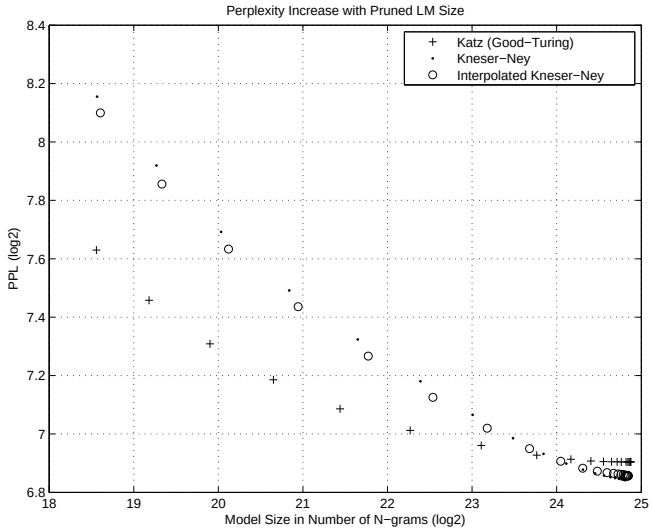

Fig. 4. Stolcke pruned 4-gram model perplexity as a function of model size (no. n-grams) for Katz, Kneser-Ney and Interpolated Kneser-Ney models as implemented by the SRILM toolkit. The Interpolated Kneser-Ney model is estimated by turning the *-interpolate* option on in the SRILM toolkit.

3. PRUNING INTERACTION WITH SMOOTHING

We examined the interaction between Stolcke pruning [5] and various n-gram LM smoothing techniques for aggressive pruning regimes which cut the original LM to under 1% of its original size. The main finding is that the increasingly popular family of Kneser-Ney [1] smoothing methods is in fact poorly suited for such aggressive pruning regimes.

Seymore-Rosenfeld pruning [6] is an alternative to Stolcke pruning that relies on the relative frequency in the training data for a given context $f(h)$ instead of the probability $P(h)$ computed from lower order estimates. For Kneser-Ney models this eliminates one source of potential problems in pruning: since the $P(h)$ calculation involves only lower order n-gram estimates it will use the diversity based estimates, which are quite different from the relative frequency ones.

Figure 4 shows the the change in perplexity with the number of n-grams in the entropy pruned model. We experimented with 4-gram models built on Broadcast News data, as reported in [7]. Although the unpruned Kneser-Ney models start from a slightly lower perplexity than the Katz model, they degrade faster with pruning.

In experiments for voice search we observed large relative differences in perplexity between Kneser-Ney/Katz models—after pruning them to 0.1% of their original size. Aggressive Stolcke pruning for a Kneser-Ney 4-gram model can lead to a relative increase in perplexity that is twice as large as for the other smoothing techniques evaluated—135% vs. 65% relative increase. The differences were also found to impact speech recognition accuracy significantly, approximately 10% relative.

Since the difference between Katz and Kneser-Ney is very small on unpruned models and significant on pruned models we chose to use Katz smoothing when building a LM for voice search. A thorough analysis on the interaction between LM smoothing and aggressive pruning in this context is presented in [7].

Training Set	Test Set PPL	
	Unpruned	Pruned
230B	121	205
BIG	132	209

Table 4. Pruned and unpruned 3-gram language model perplexity when trained on the most recent 230 billion words, and a much larger amount of training data prior to test data, respectively.

4. QUERY STREAM NON-STATIONARITY

Our first attempt at improving the language model was to use more training data: we used a significantly larger amount of training data (BIG) vs. the most recent 230 billion (230B) prior to September 2008. The 230B corpus is the most recent subset of BIG. As test data we used a random sample consisting of 10k queries from Sept-Dec 2008.

The first somewhat surprising finding was that this had very little impact in OoV rate for 1M word vocabulary: 0.77% (230B vocabulary) vs. 0.73% (BIG vocabulary). Perhaps even more surprising however is the fact that the significantly larger training set did not yield a better language model, despite the training data being clearly well matched, as illustrated in Table 4. In fact, we observed a significant reduction in PPL (10%) when using the more recent 230B data. Pruning masks this effect, and the differences in PPL and WER become insignificant after reducing the language model size to approximately 10 million 3-grams.

Since the vocabulary, and training data set change between the two rows, the PPL differences need to be analyzed in a more careful experimental setup.

A superficial interpretation of the results seems to contradict the *"there's no data like more data"* dictum, recently reiterated in a somewhat stronger form in [8], [9] and [10].

Our experience has been that supply of *"more data"* needs to be matched with increased demand on the modeling side, usually by increasing the model capacity—typically achieved by estimating more parameters. Experiments reported in Section 6 improve performance by *keeping the amount of training data constant* (albeit very large), and *increasing the n-gram model size* by adding more n-grams at fixed n, as well as increasing the model order n. As such, it may well be the case that the increase in PPL for the BIG model is in fact due to limited capacity in the 3-gram model.

More investigation is needed to disentangle the effects of query stream non-stationarity from possible mismatched model capacity issues. A complete set of experiments needs to:

- let the n-gram order grow as large as the data allows;
- build a sequence of models trained on exactly the same amount of data obtained by sliding a time-window of varying length over the query stream, and control for the ensuing vocabulary mismatches.

5. LOCALE MATTERS

We also built locale specific English language models using training data prior to September 2008 across 3 English locales: USA (USA), Britain (GBR, about a quarter of the USA amount) and Australia (AUS, about a quarter of the GBR amount). The test data consisted as before of 10k queries for each locale sampled randomly from Sept-Dec 2008.

Tables 5, 6, 7 show the results. The dependence on locale is surprisingly strong: using an LM on out-of-locale test data doubles the OoV rate and perplexity, either pruned or unpruned.

Training Locale	Test Locale		
	USA	GBR	AUS
USA	**0.7**	1.3	1.6
GBR	1.3	**0.7**	1.3
AUS	1.3	1.1	**0.7**

Table 5. Out of Vocabulary Rate: locale specific vocabulary halves the OoV rate

Training Locale	Test Locale		
	USA	GBR	AUS
USA	**132**	234	251
GBR	260	**110**	224
AUS	276	210	**124**

Table 6. Perplexity of unpruned LM: locale specific LM halves the PPL of the unpruned LM

We have also build a combined model by pooling data across locales, with the results shown on the last row of Table 7. Combining the data negatively impacts all locales, in particular the ones with less data. The farther the locale from USA (as seen on the first line, GBR is closer to USA than AUS), the more negative the impact of clumping all the data together, relative to using only the data from that given locale.

Training Locale	Test Locale		
	USA	GBR	AUS
USA	**210**	369	412
GBR	442	**150**	342
AUS	422	293	**171**
combined	227	210	271

Table 7. Perplexity of pruned LM: locale specific LM halves the PPL of the unpruned LM. Pooling all data is suboptimal.

6. EFFECT OF LANGUAGE MODEL SIZE ON SPEECH RECOGNITION ACCURACY

The work described in [11] and [12] enables us to evaluate relatively large query language models in the 1-st pass of our ASR decoder by representing the language model in the OpenFst [13] framework. Figures 5-6 show the PPL and word error rate (WER) for two language models (3-gram and 5-gram, respectively) built on the 230B training data, after entropy pruning to various sizes in the range 15 million - 1.5 billion n-grams. Perplexity is evaluated on the test set described in Section 4; word error rate is measured on another test set representative for the voice search task.

As can be seen, perplexity is very well correlated with WER, and the size of the language model has a significant impact on speech recognition accuracy: increasing the model size by two orders of magnitude reduces the WER by 10% relative.

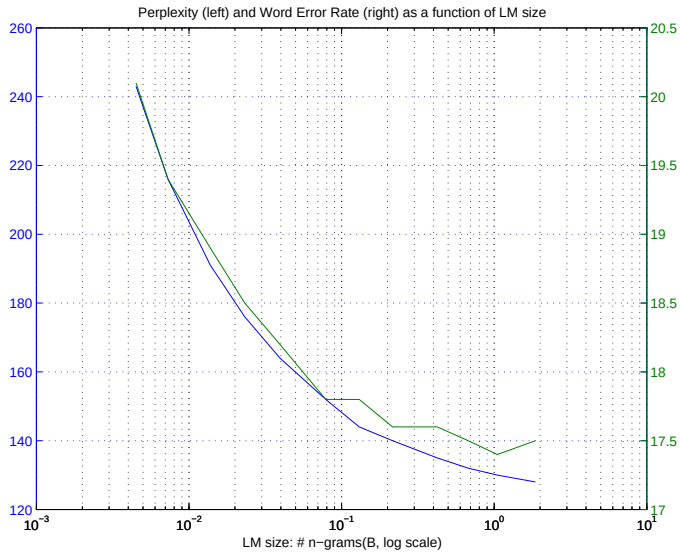

Fig. 5. 3-gram language model perplexity and word error rate as a function of language model size; lower curve is PPL.

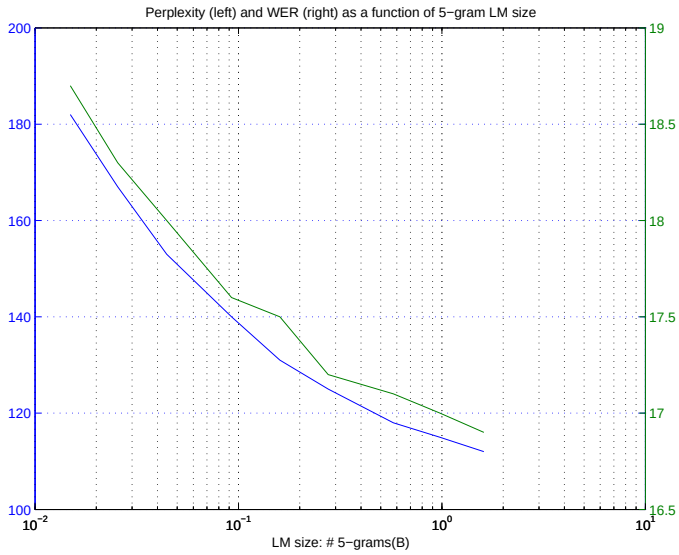

Fig. 6. 5-gram language model perplexity and word error rate as a function of language model size; lower curve is PPL.

We have also implemented lattice rescoring using the distributed language model architecture described in [14], see the results presented in Table 8. This enables us to validate empirically the fact that rescoring lattices generated with a relatively small 1-st pass language model (in this case 15 million 3-gram, denoted 15M 3-gram in Table 8) yields the same results as 1-st pass decoding with a large language model. A secondary benefit of the lattice rescoring setup is that one can evaluate the ASR performance of much larger language models.

Pass	Language Model	PPL	WER
1st	15M 3-gram	191	18.7
1st	1.6B 5-gram	112	16.9
2nd	15M 3-gram	191	18.8
2nd	1.6B 5-gram	112	16.9
2nd	12.7B 5-gram	108	16.8

Table 8. Speech recognition language model performance when used in the 1-st pass or in the 2-nd pass—lattice rescoring.

7. CONCLUSIONS

Our experiments show that with careful text normalization the query stream is not as "wild" as it seems at first sight. One can achieve excellent OoV rates for a one million word vocabulary, and n-gram hit ratios of 77/88% even at $n = 5/4$, respectively.

We have confirmed in a different experimental setup the less known fact that aggressive entropy pruning (in particular Stolcke pruning) significantly degrades language models built using Kneser-Ney smoothing, whereas Katz smoothing performs much better.

Experimental evidence suggests that the query stream is non-stationary, and that more data does not automatically imply better models even when the data is clearly matched to the test data. More careful experiments are needed to adjust model capacity and identify an optimal way of blending older and recent data—attempting to separate the stationary/non-stationary components in the query stream. Less surprisingly, we have shown that locale matters significantly for English query data across USA, Great Britain and Australia.

As a concluding remark, we generally see excellent correlation of WER with PPL under various pruning regimes, as long as the training set and vocabulary stays constant.

8. ACKNOWLEDGMENTS

Thanks Mark Paskin for providing the spelling correction data, and Zhongli Ding for test set query selection.

9. REFERENCES

[1] R. Kneser and H. Ney, "Improved backing-off for m-gram language modeling," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 1995, vol. 1, pp. 181–184.

[2] S. Katz, "Estimation of probabilities from sparse data for the language model component of a speech recognizer," in *IEEE Transactions on Acoustics, Speech and Signal Processing*, March 1987, vol. 35, pp. 400–01.

[3] V. Siivola, T. Hirsimaki, and S. Virpioja, "On Growing and Pruning Kneser–Ney Smoothed N-Gram Models," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 5, pp. 1617–1624, 2007.

[4] R. Kneser, "Statistical language modeling using a variable context length," in *Proc. ICSLP '96*, Philadelphia, PA, 1996, vol. 1, pp. 494–497.

[5] Andreas Stolcke, "Entropy-based pruning of back-off language models," in *Proceedings of News Transcription and Understanding Workshop*, Lansdowne, VA, 1998, pp. 270–274, DARPA.

[6] K. Seymore and R. Rosenfeld, "Scalable back-off language models," in *Proceedings ICSLP*, Philadelphia, 1996, vol. 1, pp. 232–235.

[7] C. Chelba, T. Brants, W. Neveitt, and P. Xu, "Study on interaction between entropy pruning and Kneser-Ney smoothing," in *Proceedings of Interspeech*, p. to appear. Makuhari, Japan, 2010.

[8] M. Banko and E. Brill, "Mitigating the paucity-of-data problem: Exploring the effect of training corpus size on classifier performance for natural language processing," in *Proceedings of the first international conference on Human language technology research*. Association for Computational Linguistics Morristown, NJ, USA, 2001, pp. 1–5.

[9] F.J. Och, "Statistical machine translation: Foundations and recent advances," in *Presentation at MT-Summit*, 2005.

[10] A. Halevy, P. Norvig, and F. Pereira, "The unreasonable effectiveness of data," *IEEE Intelligent Systems*, vol. 24, no. 2, pp. 8–12, 2009.

[11] B. Harb, C. Chelba, J. Dean, and S. Ghemawat, "Back-off language model compression," in *Proceedings of Interspeech*, pp. 325–355. Brighton, UK, 2009.

[12] Cyril Allauzen, Johan Schalkwyk, and Michael Riley, "A generalized composition algorithm for weighted finite-state transducers," in *Proc. Interspeech*, 2009, pp. 1203–1206.

[13] Cyril Allauzen, Michael Riley, Johan Schalkwyk, Wojciech Skut, and Mehryar Mohri, "OpenFst: A general and efficient weighted finite-state transducer library," in *Proceedings of the Ninth International Conference on Implementation and Application of Automata, (CIAA 2007)*. 2007, vol. 4783 of *Lecture Notes in Computer Science*, pp. 11–23, Springer, `http://www.openfst.org`.

[14] T. Brants, A. C. Popat, P. Xu, F. J. Och, and J. Dean, "Large language models in machine translation," in *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, 2007, pp. 858–867.