INTERSPEECH 2009 BRIGHTON

Language Modeling for What-with-Where on GOOG-411

Charl van Heerden [1*], *Johan Schalkwyk* [2], *Brian Strope* [2]

[1] HLT Research group, Meraka Institute, CSIR, South Africa
[2] Google Inc., USA

Abstract

This paper describes the language modeling architectures and recognition experiments that enabled support of 'what-with-where' queries on GOOG-411. First we compare accuracy trade-offs between a single national business LM for business queries and using many small models adapted for particular cities. Experimental evaluations show that both approaches lead to comparable overall accuracy. Differences in the distributions of errors also lead to improvements from a simple combination. We then optimize variants of the national business LM in the context of combined business and location queries from the web, and finally evaluate these models on a recognition test from the recently fielded 'what-with-where' system.

Index Terms: Language modeling, directory assistance, voice search, speech recognition

1. Introduction

Today successful commercial speech recognition systems typically depend on limited domains and strong language models in order to reach usable accuracy. For example most deployed dialog systems will prompt callers for what they can say (e.g. "What city and state?"). As computational capacity, available data, and model sizes have grown, systems are providing usable accuracy for increasingly open tasks, which in turn provide increasing value to users.

Most directory-assistance applications make use of speech recognition (e.g. 800-555-TELL, 800-FREE-411, 800-CALL-411, http://www.livesearch411.com). Some detail about approaches considered has been presented [1, 2, 3]. Most of these systems have leveraged the existing directory assistance UI-model that starts with "What city and state?" This is in contrast to more general call-routing applications, "How may I help you?" [4], and increasingly in contrast to web search engines for local information (e.g. http://maps.google.com) which have migrated to a single text-input box.

800-GOOG-411 is an automated system that uses speech recognition and web search to help people find and call businesses. By first asking callers for the city and state, the system can make the probabilities for the expected business queries conditional on each city. When we open the dialog by moving to a new domain of combined 'what-with-where' queries, we both confound the recognition problem, and lose the ability to have explicit models for different cities. This direction raises two experimental questions: 1) Can a single national business LM compensate for what might be lost when city-specific information is not available with an open dialog? and 2) Can models derived from separate sub-dialog states predict natural combinations of those states?

Section 2 describes the data sources and text processing used to estimate the language models. Section 3 describes the basic design of two language modeling approaches considered for business queries; a single national business LM approach (section 3.1) and city specific business language models (section 3.2). Section 4 compares experimental results for these two approaches, and for a simple combination strategy. Section 5 describes the extension of the national business LM to a system that predicts less restrictive business queries, and shows the recognition performance of these systems on a transcribed test set from production data.

* Work performed during Google intership.

2. Data sources and text processing

Table 1 summarizes the three data sources used to train our language models. The GOOG-411 transcripts were collected over a little more than one year, and the sample of Google Maps queries considered business queries, collected over the last two years. Both of these datasets are anonymous query counts. The business databases are a compilation of commercially available US business listings. Each of these data sources has its own normalization challenges.

Previously [5] we found that the US business listings were not very helpful, and did not consider that source for the national business LM described below. Note that others have proposed strategies to get more value from similar databases [6].

Source	Relevance	Uniq #	Total #
GOOG-411 trans.	highest	2M	10M
Google Maps	high	20M	2B
Business DBs	low	20M	20M

Table 1: Data sources for language modeling.

Fig. 1 shows an overview of the two-step data preparation phase used to process the data sources. After initial text normalization, an annotation step identifies substrings of a given query that are either a business/category or a location.

For example, consider the query "looking for restaurants in Portland Oregon." A business annotator identifies "restaurants" as a business/category, while the location annotator identifies "portland Oregon" as a city-state. This separation enables both task-specific data selection (location and business), and task-specific text normalization.

The output of this second step is used to build the business and location language models described in Section 3.

3. Language Modeling Alternatives

In this section we consider two types of language models for business queries: a single national LM and a system built from many city-specific models.

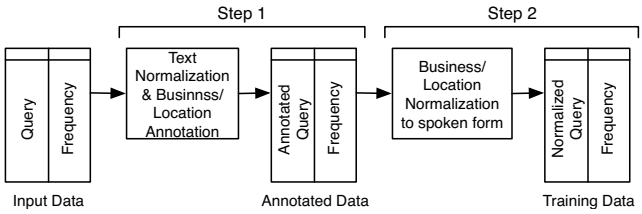

Figure 1: Steps of the data preparation phase.

3.1. National Business LM Estimation

To build a national business language model, we trained two individual tri-gram models from our normalized sources: one from Google Maps queries and one from transcribed GOOG-411 queries. The model estimated from Maps queries included words that occurred 3 or more times, while no vocabulary pruning was necessary for the model of GOOG-411 queries. Kneser-Ney smoothing [7] was used for both models.

Interpolation weights were estimated by evaluating perplexity (PPL) on a development set from the GOOG-411 queries. Empirically, a weight of 0.8 for the GOOG-411 model was best for this evaluation. The interpolated model was then entropy pruned [8] to fit into the memory of a single machine.

The resulting N-gram sizes for the two models, and their combination are shown in Table 2. A similar national N-gram was also estimated for locations, using the GOOG-411 data from that dialog state together with the location information extracted from Maps queries. The resulting model, referred to as the citystate-LM, is smaller than the national business-LM and is described in more detail in Section 5.

Model	# 1-grams	# 2-grams	# 3-grams
GOOG-411	0.11M	0.93M	1.8M
Maps	0.50M	5.9M	13M
Interpolated	0.51M	6.1M	14M
Pruned interpolated	0.51M	5.9M	8.2M

Table 2: N-gram sizes for individual & interpolated models.

3.2. City-Specific LM Estimation

The city-specific system uses a hybrid LM that combines full-query lists in parallel with a de-weighted unigram. Individual unigram and query weights were estimated from frequency counts in the training data. The weighting of the unigram with respect to the full queries was optimized on development tests. Initially the unigram was employed as a decoy path for rejection, but because high-confidence values through the unigram were often correct, we optimized independent confidence thresholds for the distinct grammar paths.

To compensate for data sparcity, the city-specific models were smoothed against increasingly general models of region, state, and country. Interpolation weights for data sources and regionalization were optimized on a development set. For these experiments, the size of the city-specific models was held constant across cities. Because of the smoothing with increasingly less specific models, smaller cities with fewer queries ended up with less city-specific data in their models.

The city-specific system also includes a semantic stage for inverse text normalization. This stage maps the query variants like “comp usa” and ”comp u s a,” to the most common web-text query form “compusa”.

The city-specific system was designed earlier and there were a few implementation differences from the national-LM system: it used a different text normalization framework without explicit spell checking; the data sources were sampled at slightly different times; and all optimizations minimized “sentence semantic error“ which considered the mis-match of the output of the inverse text normalization.

4. Experimental Results

4.1. Tests and Measures

The recognition tests were held out and manually transcribed GOOG-411 calls. The tests were split into city-state (47K utterances) and business queries (56K). We report three types of measures: perplexity, word error, and sentence error. For sentence error, we ignored inconsistency in spaces and apostrophes (kinko’s vs kinkos). All systems were evaluated with the same fairly standard acoustic models (triphones with decision tree clustering, 16 Gaussians / state, 3 state units, using STC, and ML estimation followed by MMI, with a PLP-based front-end and LDA).

4.2. Overall Performance

The top line of Table 3 shows WER/SER for all data for the national business LM and city-specific business LMs. While the performance is close, each system does better with the measure more closely related to its structure. There are at least two factors to consider in the evaluation. First, more of the word-errors in the city-specific system can be tracked to difference in compounding and apostrophes, which are ignored by the semantic measure used to optimize that system. Second, by using full-sentence query models without a smooth N-gram backoff, the city-specific system is more likely to get the utterance completely right or completely wrong.

The accuracy of the national business LM, which consumes only a couple GBs of memory, is very close to the city-specific system that includes 100s of GBs of memory in total, and requires the explicit city-state question in the dialog. More directly, because it uses no city-specific information, the national business LM opens the possibility of a new combined state where callers provide both the business and the city simultaneously.

4.3. Error Analysis and Combination

Table 3 shows word error and sentence error for different subsets of the data on three systems: the national-LM system, the city-specific system, and a simple combination scheme described below. Each system uses a different approach to cover the tail of the distribution. The national business LM covers the tail with a large N-gram, while the city-specific LM tries to cover the tail of the national distribution by having a list of the common queries for each specific city.

The “national” lines in the table show accuracy numbers for increasingly less common queries. The “head” contains queries that are in the most common 10K, the “body” contains queries from the next 290K, and the “tail” is everything else. From these lines, it’s clear that the city-specific system provides a sentence error improvement as we move toward the tail of the distribution.

For the “city-size” lines in the table we’ve grouped all utterances by city and sorted those cities into three groups based on

Data group	% of queries	National LM	City-Specific LM	Combination
all	100	21.8 / 29.0	23.1 / 26.8	20.3 / 24.3
national head	41	10.5 / 11.6	12.5 / 11.4	12.2 / 11.0
national body	25	19.4 / 25.4	20.8 / 22.7	18.9 / 20.5
national tail	34	31.5 / 52.8	32.1 / 48.5	26.9 / 43.3
big cities	33	23.1 / 31.2	27.0 / 31.8	22.6 / 27.8
medium cities	33	20.9 / 27.7	21.3 / 24.5	19.1 / 22.8
small cities	33	21.5 / 28.1	20.6 / 23.8	18.8 / 22.3
city head	64	14.5 / 17.9	11.3 / 11.3	11.1 / 11.0
city body	14	26.0 / 38.3	21.6 / 29.5	19.8 / 27.7
city tail	21	34.9 / 56.4	48.2 / 71.6	39.5 / 62.1

Table 3: Error analysis: WER/SER (%) across different data subsets.

their frequency, with each group representing equal numbers of utterances. For this test, we can cover about a third of the data with the most common 70 cities, while it takes thousands of cities to cover the least common third. The result is expected, the national business LM system does the best with the large cities that dominated our data, while the city-specific system does increasingly better with smaller cities.

The last lines of the analysis table use a city-specific reference for the definition of head and tail. The "city head" is test utterances that are in the most common 10K query list for each city, the "city body" is utterances covered by the rest of the city-specific query lists, and the "city tail" is utterances not modeled by the city-specific query lists. With this last set the difference is more plain: both systems are poor, but the national N-gram does much better with the city-tail than the city-specific unigrams. For the rest of the queries, the city-specific query lists do much better than the national N-gram.

Based on these observations we investigated a simple combination scheme where we use the result from the city-specific system unless that result came through the unigram. In that case, we take the result from the national business LM. This gives us city-specific query models together with a smooth national back-off mechanism. Without optimizing the combination, we see large gains in both word error and sentence error. Most notably, on the tail of the national distribution, the combination improved the sentence error of the national business LM by 9.5% and the sentence error of the city-specific system by 5.2% absolute.

5. Predicting a New Domain

5.1. Text Data and an Unseen Testset

Having a single language model was necessary to have a more open dialog by asking for both "what and where" at the same time. To predict these new queries, we considered two types of language models.

The first LM was an interpolation between the citystate-LM and the national business-LM described in section 3.1. The interpolation weights were picked to reduce the total error on both business and city-state recognition tests.

The second LM combined the same two LM pieces using a hierarchical language model (HLM) [9] which explicitly includes paths for a business query, a city-state query, or multiple combinations of the two in a single utterance. Having an HLM therefore allows us to estimate explicit priors for different types of combined queries, while still re-using LM pieces that were already highly-optimized for existing GOOG-411 data. The HLM also enables rapid adaptation of these priors. The topology for the HLM, shown in Fig. 2 was constructed manually. All paths starting at node 0 and ending at node 5 represent possible queries. We estimated the initial topology and the associated arc weights from a small "onebox" (text) evaluation set built from Google web queries that get a Maps business result.

We chose to use the HLM in the first "what-with-where" production system, and then transcribed a new test set from production data. Table 4 summarizes errors across all models and across four test sets. The citystate and business test sets contain transcribed GOOG-411 queries from the original system that first asked for a city and then a business. The "what-with-where" test set contains the transcribed queries from the recently fielded "what-with-where" system. Looking across the data we had before fielding the "what-with-where" system, the HLM results are comparable to that of the national business LM as well as the interpolated business and city-state LM. The production "what-with-where" recognition test shows about a 35% relative reduction in both WER and SER by the HLM compared to simple interpolation. Looking across all models, we also see a strong correlation between perplexity on onebox text data, and recognition error rates on spoken "what-with-where" data.

Given that strong correlation, we should note that to date, we've been unable to do better on this recognition task using an LM derived more directly from the web text queries used to optimize the HLM. This suggests that the HLM is allowing us to combine spoken-domain knowledge from GOOG-411 queries together with text knowledge from web queries, to help predict a new spoken domain.

5.2. Future Work and Dialog Considerations

These data and the general structure of the HLM suggest many future directions. The arc weights and maybe even the topology will most likely change as we obtain examples of spoken queries in this combined domain. More basically, we may expect that the same HLM structure that enabled an efficient migration path to a more natural combined domain, eventually also limits the modeling potential in that new domain. For example, with enough data in the new domain, we might expect a simple N-gram to start to model many of the city-specific dependencies that the initial HLM structure intentionally removes.

Allowing callers to say what and where together also may not be an obvious improvement for directory assistance. To a first degree, if callers can give a natural response that provides the information for two dialog states in one, then we've saved time, and given ourselves a more interesting recognition challenge. But the new prompt may confuse some callers, it adds more semantic complexity in interpretting the caller's response (was it what-only, where-only, or what-with-where?),

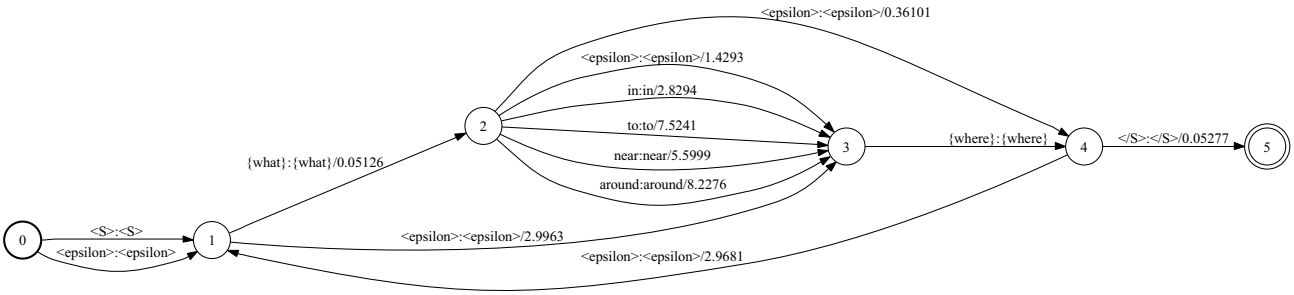

Figure 2: An HLM "What/Where" model.

Model	# of N-grams	citystate WER / SER	business WER / SER	onebox PPL	what-with-where WER / SER
citystate-LM	2.4M	5.8 / 8.3	-	-	-
business-LM	14M	9.4 / 12.7	21.8 / 29.0	441	19.1 / 40.9
bus-city-interp	14M	6.7 / 9.5	21.9 / 29.0	284	15.8 / 35.6
bus-city-HLM	14M	8.0 / 11.0	22.1 / 29.4	186	10.4 / 22.7

Table 4: Perplexity, accuracy, combined domain (onebox), and unseen data (what-with-where).

and given the recognition complexity of the new task, the recovery paths of the dialog are now complicated by an increase in initial recognition errors.

To a second degree, allowing for the mixing of what-with-where may also give the caller a more natural path for queries like "Metropolitan Museum of Art," "Stanford University Golf Course." or even "American Airlines," where location can be implicit or unknown.

Lastly, optimizations of dialog recovery strategies for the full system, which includes both an initial "what-with-where" question followed by an optional city-specific recovery mechanism, suggest that having two paths for the caller to get information might provide another opportunity for combination wins. If the errors of the two paths are decorrelated, then when one path isn't working for the caller, it's a better bet to try the other. The details of the dialog optimizations needed to get to comparable performance in task completion and total time using "what-with-where" recognition will be described in a future paper.

6. Conclusion

This paper described the development of the language modeling that enabled recognizing "what-with-where" queries for a commercial business directory assistance system. We showed that using a single national business LM reached similar accuracy levels as a city-specific system on a business query recognition task, and that additional wins are obtained with combinations of the two approaches. Then we showed that an HLM composition of national business and location LMs can be optimized to predicted a naturally combined domain of more open business text queries. Finally we showed that the HLM optimized to predict the combined text data performed best on an unseen recognition test set collected from the production "what-with-where" system.

7. Acknowledgements

We thank Boulos Harb, Will Neveitt, Vida Ha, Carolina Parada, Mike Schuster, Francoise Beaufays, Ciprian Chelba, and Greg Morris for providing tools, infrastructure, and consultation for the experiments reported in this paper.

8. References

[1] N. Gupta, G. Tur, D. Hakkani-Tur, S. Bangalore, G. Riccardi, and M. Gilbert, "The AT&T spoken language understanding system," in *Proc. ASLP*, January 2006, pp. 213–222.

[2] S. Chang, S. Boyce, K. Hayati, I. Alphonso, and B. Buntschuh, "Modalities and demographics in voice search: Learnings from three case studies," in *Proc. ICASSP*, April 2008, pp. 5252–5255.

[3] A. Acero, N. Bernstein, R. Chambers, Y.C. Ju, X. Li, J. Odell, P. Nguyen, O. Scholz, and G. Zweig, "Live search for mobile: Web services by voice on the cellphone," in *Proc. ICASSP*, April 2008, pp. 5256–5259.

[4] A. Gorin, G. Riccardi, and J Wright, "How may I help you?," *Speech Communication*, vol. 23, pp. 113–127, October 1997.

[5] M. Bacchiani, F. Beaufays, J. Schalkwyk, M. Schuster, and B. Strope, "Deploying GOOG-411: Early Lessons in Data, Measurement and Testing," in *Proc. ICASSP*, April 2008, pp. 5260–5263.

[6] X. Li, Y.C. Ju, G. Zweig, and A. Acero, "Language modeling for voice search: A machine translation approach," in *Proc. ICASSP*, April 2008, pp. 4931–4916.

[7] R. Kneser and H. Ney, "Improved backing-off for m-gram language modeling," in *Proc. ICASSP*, May 1995, pp. 181–184.

[8] A. Stolcke, "Entropy-based pruning of backoff language models," in *Proc. DARPA Broadcast News Transcription and Understanding Workshop*, February 1998, pp. 270–274.

[9] L. Galescu and J. Allen, "Hierarchical Statistical Language Models: Experiments on In-Domain Adaptation," in *Proc. ICSLP*, October 2000, pp. 186–189.