LEVERAGING MULTIPLE QUERY LOGS TO IMPROVE LANGUAGE MODELS FOR SPOKEN QUERY RECOGNITION

Xiao Li, Patrick Nguyen, Geoffrey Zweig, and Dan Bohus

Microsoft Research
One Microsoft Way, Redmond, WA, 98052, U.S.A.
{xiaol,panguyen,gzweig,dbohus}@microsoft.com

ABSTRACT

A *voice search* system requires a speech interface that can correctly recognize spoken queries uttered by users. The recognition performance strongly relies on a robust language model. In this work, we present the use of multiple data sources, with the focus on query logs, in improving ASR language models for a voice search application. Our contributions are three folds: (1) the use of text queries from web search and mobile search in language modeling; (2) the use of web click data to predict query forms from business listing forms; and (3) the use of voice query logs in creating a positive feedback loop. Experiments show that by leveraging these resources, we can achieve recognition performance comparable to, or even better than, that of a previously deploy system where a large amount of spoken query transcripts are used in language modeling.

Index Terms— language modeling, voice search, query log, click data

1. INTRODUCTION

Voice search [1] is an integration of speech recognition and search technologies that allows users to effectively access information using voice. The role of speech recognition therein is essentially one that transcribes a spoken query into text, either in form of one-best or n-best recognition results. The recognized queries are then used as input to an information retrieval system. Such speech interfaces for search applications have emerged into consumer market in recently years [2, 3, 4]. One example, which we focus on in this work, is *Live Search for Windows Mobile* (LS4WM) [1], a speech-enabled application operating in the domain of local search, whose goal is to return a user with the most relevant business listings given queries spoken by the user.

In no small part, the recognition performance of LS4WM (or similar systems) relies on a robust language model (LM), the construction of which is a challenging task due to the sheer volume of user queries and the lack of sufficient, in-domain training examples. The ideal training data for LS4WM, *i.e.*, transcripts of spoken local search queries, is costly to obtain at a large scale. Business listings, on the other hand, are easily accessible from a database, but listing forms are often different from how users formulate queries [3, 5]. In [5], we presented a machine translation approach that predicts query forms from listing forms. The performance of the prediction, however, still heavily depends on a substantial amount of transcribed data.

Fortunately, there are various data sources from relevant domains that we can leverage for our task, thanks to the exploding popularity of text search. In analogy to using web documents to model conversational speech [6], we exploit a variety of search query logs to model spoken queries. In addition to a business listing database, the resources of our interest include (1) web local search query log with click-through data; (2) LS4WM text query log; and (3) LS4WM voice query log. Note that throughout this paper all data sources are from the local search domain unless otherwise stated.

Given the above resources, our goal is to construct individual n-gram LMs from each data source, and to create a combined LM via model interpolation. In creating each individual LM, we investigate the following approaches:

- Use web and mobile search queries as training data for language modeling;
- Train a translation model [5] from web search click data. Then use the translation model to generate pseudo queries from listing names for language modeling.
- Use n-best recognition results, together with user confirmed ones, in maximum-likelihood and discriminative training of an LM.

Once individual LMs are trained, we create a single interpolated backoff model [7] in which each n-gram probability is linearly interpolated from the probabilities of these LMs, and the backoff weights are re-normalized accordingly. The rest of the paper is organized as follows: Section 2-4 will discuss the above three aspects respectively. Section 5 presents experiments and recognition results, followed by conclusions.

2. LANGUAGE MODELING USING TEXT QUERY LOGS

Since the training data that best fits our interest is transcripts of spoken queries, the first attempt one would make is to use text queries in similar domains as a substitute. In this work, we collect such data from two sources. The first is a text query log from local.live.com, a web-based local search service, and we refer to this dataset as *web text*. In addition to user search queries, this log also contains click-through information. Each click instance is recorded in form of a (query, listing) pair. To maximally filter out queries that are out of the local search domain, we extracted only queries that resulted in clicks. The intuition is that a query not meant for local search would probably trigger irrelevant search engine results from local.live.com, and the user is less likely to click on any of them. Our second data source is a text query log from LS4WM, which will be referred to as *mobile text*. This dataset contains queries that were *typed*, as opposed to spoken, to the local search application on *Windows Mobile*.

For text queries to be useful in language modeling, the foremost task is text normalization which is to convert query words into “canonical” forms consistent with those used in official business names. In both datasets, we observe many inconsistent representations of the same business name, mostly due to spelling errors. For example, the business *ABC Quick Mart* is mistakenly referred to as “ABC Quik Mart” in some queries. A global substitution of ”Quick” for ”Quik” is inappropriate because the word “Quik” does exist in other business names. It is also prohibitively time-consuming to manually specify what to do in each specific context. Therefore, we have developed an automated process based on transduction. This process uses the following steps:

1. A ”gold standard” language model is built from the business listing database that has been previously cleaned.
2. A set of possible word level substitutions is specified, *e.g.*, allowing ”Quick” to be substituted for ”Quik”.
3. A transducer is built such that the score assigned to any path is the gold-standard language model score of the output-side word sequence. Each input symbol is allowed to produce itself on the output side, as well as any substitutions specified in the previous step.

By transducing noisy text queries in this manner, we obtain on the output side the sequence of words that is consistent with the allowed substitutions, and that has the highest language model probability. A query is untransducable and hence discarded if it contains a word out of the business listing vocabulary and if the OOV word is not assigned any substitutions. Our method is described in more detail in [1] and [8].

After text normalization, query words in both datasets are confined to the business listing vocabulary but their query distributions are rather different. Table 1 shows the most frequency queries from the *web text* and *mobile text* datasets respectively, as well as those from a random set of LS4WM spoken query transcripts. Notice that *mobile text* seems to be more relevant to our target domain, which is not surprising since *mobile text* comes from the same mobile application as the spoken queries. On the other hand, *web text* is typically larger than *mobile text* if collected within the same time frame. In other words, while *mobile text* is closer to in-domain data, *web text* takes less time to collect to a desired size. In Section 5, we will show recognition performance of using these two data sources, with varied amounts, in language modeling.

Web text	**Mobile text**	**Mobile Voice**
Hotels	*Pizza*	*McDonald’s*
Restaurants	*Wal-Mart*	*Wal-Mart*
Real Estate	*Starbucks*	*Pizza*
Apartments	*Best Buy*	*Movies*
Motels	*Target*	*Starbucks*
Newspaper	*Pizza Hut*	*Mexican Restaurant*
Newspapers	*Home Depot*	*Target*
Hotels	*Motels & Hotels*	*Best Buy*
Churches	*Restaurants*	*Burger King*
News	*GameStop*	*Pizza Hut*

Table 1. Top ten local search queries (after text normalization) from three different data sources

3. TRANSLATION MODEL TRAINING WITH CLICK DATA

Using existing user queries as LM training data has the limitation that it does not generalize to business listings that have not been asked for. In fact, there are 20M entries in our business listing database; what users have asked for may only constitute a small portion of it. In [5], we presented a machine translation approach that “translates” business listings to pseudo queries. Our translation model is based on n-grams, where each “gram” is a pair of query word (or phrase) and listing word (or phrase). Given a corpus of parallel sentences in form of (query, listing) pairs, we discover the best alignments of the sentence pairs and estimate the n-gram model in an iterative fashion using the EM algorithm. Once trained, we “translate” all business listings into query forms, which are used as training data in language modeling. In [5], we observed significant perplexity reduction on a spoken query dataset by using such psuedo queries, in addition to business listings and spoken query transcripts, as LM training data.

The translation model used in our previous work [5] was trained on (query, listing) pairs where queries were manually transcribed from voice search data, and where listings were retrieved from a database based on *tf-idf* scores. This approach still depends on transcribed data which is again difficult to obtain in a substantial size. Moreover, since the listing counterpart of a query is heuristically discovered which may or may not be the *true* listing a user asks for, the sentence pairs collected in this way can be noisy.

With the availability of web click data, however, we are able to harvest sentence pairs for training a translation model in an automated fashion and presumably in a cleaner form. As mentioned in Section 2, the query log of local.live.com contains user click information in form of (query, listing) pairs. The click data is in essence implicit user feedback, in which each click instance is a weak indication of relevance. By inspecting the click data as a whole, we can discover strongly relevant (query, listing) pairs. For example, if most query instances “P. F. Chang’s Restaurant” clicked on the listing *P. F. Chang’s China Bistro*, it is likely that the latter listing is the most relevant one to the former query. When using these click instances as training sentence pairs, the translation model is able to learn that the phrase “China Bistro” can be translated to “Restaurant”. On the other hand, categorical queries such as “Restaurant” may click on various listings such as *Denny’s* and *P. F. Chang’s China Bistro*. Such sentence pairs are not desired since they often lack a strong correspondence between query and listing words/phrases, which would introduce noise in learning the translation model.

To select click instances that are most useful as training sentence pairs, we resort to the *point-wise mutual information* criterion [9]. For notational convenience, we use q to denote a query, d a listing, and $c_{q,d}$ the click count associated with (q, d). Furthermore, we use c_q and c_d to represent query and listing counts respectively, and N to represent the total number of click instances. The point-wise mutual information (weighted by frequency) is computed as

$$I(q,l) = p(q,l)\log\frac{p(q,l)}{p(q)p(l)} = \frac{c_{q,l}}{N}\log\frac{c_{q,l}N}{c_q c_l} \qquad (1)$$

We select a click instance $(q_i = q, l_i = l)$ as training data if its

$I(q,l) > \alpha$, where α is a threshold. All click instances satisfying this constraint will be used as training examples despite some of them may have the same (q,l) value. In other words, the value pair (q,l) is implicitly weighted by its count in our learning process.

Once the sentence pairs are selected, we normalize the queries by transduction as described in Section 2, and we follow our previous work [5] in training the n-gram based translation model. One key difference from [5] is that we can afford to create higher-order phrases in both source and target languages owing to the drastic increase in the amount of training sentence pairs. This gives the translation model the power to capture long-distance patterns that would be otherwise difficult to model. For example, it would be beneficial to treat *Jack In The Box* as a phrase and hence a single "gram" as opposed to a sequence of words, since it as a whole is a business entity name. Given sufficient data, we are able to automatically discover such phrases. This is achieved by computing the point-wise mutual information between words $I(w,w')$. A sequence of query/listing words $w_{m:n}$ is considered a phrase if all $I(w_i,w_{i+1})$, $i = m,\dots,n-1$, are above a threshold.

4. CREATING A FEEDBACK LOOP WITH VOICE QUERY LOG

Another data source we investigate in this work is the voice query log from a deployed LS4WM system [1]. Specifically, when user speakers a query to an LS4WM client, a confirmation screen will be displayed that contains the n-best recognition results. The user can select the correct hypothesis and then the search engine will be triggered by the user's selection. In this process, both the n-best hypotheses and users' confirmations are logged, which we can take advantage of to create a positive feedback loop.

On natural way of using this resource is to train an LM from user confirmed recognition results. Although such data represents what the deployed ASR system is able to recognize correctly, it provides information, at least partially, about the prior distribution of voice search queries. Notice that this prior information is different from that of web/mobile text queries, and is missing in translated pseudo queries. In this regard, using the confirmed recognition results, if available, can be a good complement to the other data sources we have explored.

Furthermore, we use this data source to train an n-gram LM discriminatively, where we assume that the recognition result clicked by the user is the ground truth while other n-best alternates are competitors. There are multiple ways that we can leverage such data for discriminative training *e.g.*, re-estimating n-gram probabilities with a discriminative criterion [10] or learning a discriminative model directly [11]. In this work, we adopt the former approach and we optimize n-gram LM parameters that maximize the conditional likelihood objective:

$$p(q_c|x) = \frac{p(x|q_c)p(q_c)}{\sum_{q\in Q} p(x|q)p(q)} \tag{2}$$

Here x denotes acoustics and q_c denotes user clicked hypothesis out of a list of n-best recognition results Q. In maximum conditional likelihood (MCL) training, we keep the acoustic model $p(x|q)$ fixed and only update parameters of the language model $p(q)$. In practice, we use an LM scaling factor β to adjust the importance of the LM with respect to the acoustic model. In other words, $p(q)$ is replaced by $p^{\beta}(q)$ in the above objective function. We apply stochastic gradient descent to update n-gram probabilities while keeping the backoff weights fixed. In application time, the MCL LM is used to rescore the n-best alternates generated by the same ASR engine.

5. EXPERIMENTS AND RESULTS

We evaluate our LMs by measuring speech recognition performance on a test set of 4388 real spoken queries collected from LS4WM. We use a fixed acoustic model and experiment with LMs trained from different data sources. There are two baseline LMs in this work: one is trained from 20M listing names from a business listing database; the other is the LM used in a previous deployed system which was trained from a combination of listing names and manually transcribed spoken queries [1]. The one-best and n-best accuracies as well as acceptance rates of these two baseline LMs are reported in the first two rows of Table 2. Note that an n-best list contains at most 10 hypotheses with a high enough confidence, and there can be $n < 10$ in some cases. An utterance is rejected (thereby not counted as "accepted") if no hypothesis satisfies the confidence constraint.

The focus of our work is to see the performance of LMs that are trained without using *any* spoken query transcripts. To this end, we collected the following datasets for language modeling.

- *Web text* (W): 10M normalized text queries from local.live.com that resulted in clicks. The data was collected from a continuous period of time.
- *Mobile text* (M): 5M normalized text queries from LS4WM (not necessarily with clicks). This was collected from roughly the same time period as the *web text* dataset.
- *Translation* (R): 21M pseudo queries generated from the business listing database based on the translation model described in Section 3. The translation model was trained on 8.5M $(q_i = q, l_i = l)$ pairs, with $I(q,l) > 7e-07$, selected from the 10M click instances of local.live.com. We chose such a threshold as to include those pairs where both the query and the listing appear exactly once. The discarded pairs mostly consist of categorical queries.
- *Confirmed voice* (C): 650K voice search queries corresponding to user clicked recognition results in the deployed LS4WM system [1]. There is also a list of n-best hypotheses associated with each confirmed result.

Notice that the amounts of data from the above resources were not intentionally chosen, but represent all data that is readily available to us at the time of our experiments.

In our first experiment, we trained LMs using each of the above data sources alone. The results are reported in the second section of Table 2. Interestingly, the performance of these LMs is comparable to that of the deployed one, and this is achieved without the need of any human transcribe data. This argument, however, does not apply to the case of *confirmed voice*, since the gerenation of such data depends on the deployed system which was trained on human transcribed data.

To illustrate the effectiveness of using *web text* and *mobile text* respectively for language modeling, we varied the amounts of training data in both cases. As shown in Figure 1, with a fixed training set size, the LM trained using *mobile text* alone yielded significantly higher recognition accuracy compared with that using *web*

LMs	1-best Accuracy	n-best Accuracy	% Accept
Listng (L)	48.44	58.81	69.71
Deployed (L+Manual)	53.01	60.62	69.12
Web text (W)	53.39	63.98	73.19
Mobile text (M)	52.60	63.08	73.04
Translation (R)	52.58	63.06	73.06
Confirmed voice (C)	53.10	63.70	72.90
W+M+R	54.57	64.93	73.25
W+M+R+C	55.83	67.62	74.66
L+W+M+R+C	54.79	65.15	73.13
MCL on W+M+R+C	55.95	(67.62)	(74.66)

Table 2. One-best accuracies, n-best accuracies where $n = 10$ (or $n < 10$ due to confidence thresholding), and acceptance rates on 4388 voice search queries.

text alone. However, the recognition accuracy of using *mobile text* appears to hit a plateau with 2.5M queries, while that of using *web text* keeps increasing, and is approximately linear in the log of training data size. In fact, *web text* data can be collected with a higher rate than *mobile text*, considering the fact that mobile search users are likely to be outnumbered by web search users.

Given the LMs trained separately on different resources, we created a single interpolated backoff model. We simply used equal weights for all LMs that participate in the interpolation, although fine-tuning the weights may further improve performance. For example, W+M+R in Table 2 means that the LM is an interpolation of W, M, and R with an equal weight $1/3$. As shown in the table, when all five individual LMs are combined, we achieve a one-best accuracy of 54.79%. The accuracy is increased to 55.83% when the listing names are excluded from training.

Finally, we conducted MCL training of the best LM we have, *i.e.*, W+M+R+C. We then used the MCL LM to rescore the test-set n-best hypotheses that were produced by an ASR system using the W+M+R+C LM. The one-best accuracy after rescoring is 55.95%, which is not a significant improvement. This is likely due to the mismatch between training and test data. Ideally, the training-set hypotheses should be generated from the same system, *i.e.* the engine using the W+M+R+C LM. In our case, however, n-best alternates as well as user clicked results were obtained via user interactions with a previously deployed system [1].

6. CONCLUSIONS AND FUTURE WORK

In this work, we studied the construction of LMs for spoken query recognition from different data sources. The most effective resources include text queries from web and mobile search applications, pseudo queries generated from a translation model that is trained on web click data, and user confirmed voice search queries. An interpolation of individual LMs trained on these data sources gives an additional boost to recognition accuracy, which outperforms a previous LM trained with transcribed data. In the future, we would like to refresh our data for discriminative training of the LM. The training data should be collected from user interactions with an LS4WM system with an updated LM.

The authors would like to thank Sarah Zhai for providing necessary data for this work.

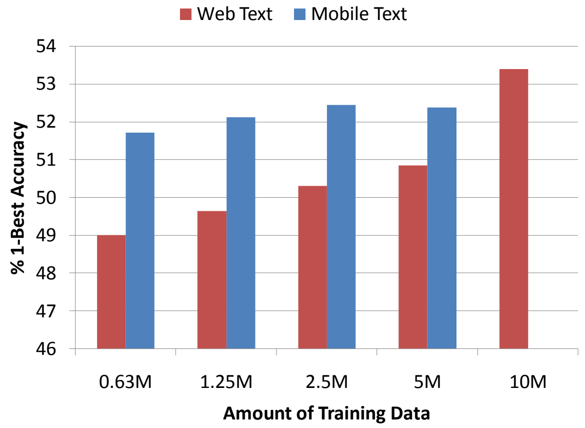

Fig. 1. One-best recognition accuracies using LMs trained with different amounts of web and mobile text queries

7. REFERENCES

[1] Acero et al., “Live search for mobile: Web services by voice on the cellphone,” in *Proc. of ICASSP*, 2008.

[2] Shuangyu Chang, Susan Boyce, Katia Hayati, Issac Alphonso, and Bruce Buntschuh, “Modalities and demographics in voice search: Learnings from three case studies,” in *Proc. of ICASSP*, 2008.

[3] Michiel Bacchiani, Francoise Beaufays, Johan Schalkwyk, Mike Schuster, and Brian Strope, “Deploying GOOG-411: Early lessons in data, measurement, and testing,” in *Proc. of ICASSP*, 2008.

[4] “VLingo FIND,” http://www.vlingomobile.com/downloads.html.

[5] Xiao Li, Yun-Cheng Ju, Geoffrey Zweig, and Alex Acero, “Language modeling for voice search: a machine translation approach,” in *Proc. of ICASSP*, 2008.

[6] Ivan Bulyko, Mari Ostendorf, Manhung Siu, Tim Ng, Andreas Stolcke, and Ozgur Cetin, “Web resources for language modeling in conversational speech recognition,” *ACM Transactions on Speech and Language Processing*, vol. 5, no. 1, 2007.

[7] Andreas Stolcke, “SRILM – an extensible language modeling toolkit,” in *Proc. of ICSLP*, 2002.

[8] Yun-Cheng Ju and Julian Odell, “A language-modeling approach to inverse text normalization and data cleanup for multimodal voice search applications,” in *Proc. of Interspeech*, 2008.

[9] Chris Manning and Hinrich Schutze, *Foundations of Statistical Natural Language Processing*, MIT Press, 1999.

[10] Hong-Kwang Jeff Kuo, Eric Fosler-Lussier, Hui Jiang, and Chin-Hui Lee, “Discriminative training of language models for speech recognition,” in *Proc. of ICASSP*, 2002.

[11] Dan Bohus, Xiao Li, Patrick Nguyen, and Geoffrey Zweig, “Learning n-best correction models from implicit user feedback in a multi-modal local search application,” in *Proc. of SIGDIAL’08*, 2008.