Learning N-Best Correction Models from Implicit User Feedback in a Multi-Modal Local Search Application

Dan Bohus, Xiao Li, Patrick Nguyen, Geoffrey Zweig
Microsoft Research
One Microsoft Way
Redmond, WA, 98052
{dbohus, xiaol, panguyen, gzweig}@microsoft.com

Abstract

We describe a novel n-best correction model that can leverage implicit user feedback (in the form of clicks) to improve performance in a multi-modal speech-search application. The proposed model works in two stages. First, the n-best list generated by the speech recognizer is expanded with additional candidates, based on confusability information captured via user click statistics. In the second stage, this expanded list is rescored and pruned to produce a more accurate and compact n-best list. Results indicate that the proposed n-best correction model leads to significant improvements over the existing baseline, as well as other traditional n-best rescoring approaches.

1 Introduction

Supported by years of research in speech recognition and related technologies, as well as advances in mobile devices, speech-enabled mobile applications are finally transitioning into day-to-day use. One example is Live Search for Windows Mobile (2008), a speech-enabled application that allows users to get access to local information by speaking a query into their device. Several other systems operating in similar domains have recently become available (TellMeByMobile, 2008; Nuance Mobile Search, 2008; V-Lingo Mobile, 2008; VoiceSignal Search, 2008.)

Traditionally, multi-modal systems leverage the additional input channels such as text or buttons to compensate for the current shortcomings of speech recognition technology. For instance, after the user speaks a query, the Live Search for Windows Mobile application displays a confirmation screen that contains the n-best recognition results. The user selects the correct hypothesis using the buttons on the device, and only then the system displays the corresponding search results (see Figure 1.)

We argue that ideally multi-modal systems could use the additional, more accurate input channels not only for confirmation or immediate correction, but also to learn from the interaction and improve their performance over time, without explicit human supervision. For example, in the interaction paradigm described above, apart from providing the means for selecting the correct recognition result from an n-best list, the user click on a hypothesis can provide valuable information about the errors made by system, which could be exploited to further improve performance.

Consider for instance the following numbers from an analysis of logged click data in the Live Search for Windows Mobile system. Over a certain period of time, the results *Beer* and *Gear* were displayed together in an n-best list 122 times. Out of these cases, *Beer* was clicked 67% of the time, and *Gear* was never clicked. In 25% of the cases when *Beer* was selected, *Gear* was incorrectly presented above (i.e. higher than) *Beer* in the n-best list. More importantly, there are also 82 cases in which *Gear* appears in an n-best list, but *Beer* does not. A manual inspection reveals that, in 22% of these cases, the actual spoken utterance was indeed *Beer*. The clicks therefore indicate that the engine often misrecognizes *Gear* instead of *Beer*.

Proceedings of the 9th SIGdial Workshop on Discourse and Dialogue, pages 21–28,
Columbus, June 2008. ©2008 Association for Computational Linguistics

Ideally, the system should be able to take advantage of this information and use the clicks to create an automatic positive feedback loop. We can envision several ways in which this could be accomplished. A possible approach would be to use all the clicked results to adapt the existing language or acoustic models. Another, higher-level approach is to treat the recognition process as a black-box, and use the click feedback (perhaps also in conjunction with other high-level information) to post-process the results recognition results.

While both approaches have their merits, in this work we concentrate on the latter paradigm. We introduce a novel n-best correction model that leverages the click data to improve performance in a speech-enabled multi-modal application. The proposed model works in two stages. First, the n-best list generated by the speech recognizer is expanded with additional candidates, based on results confusability information captured by the click statistics. For instance, in the 82 cases we mentioned above when *Gear* was present in the n-best list but *Beer* was not, *Beer* (as well as potentially other results) would also be added to form an expanded n-best list. The expanded list is then rescored and pruned to construct a corrected, more accurate n-best list.

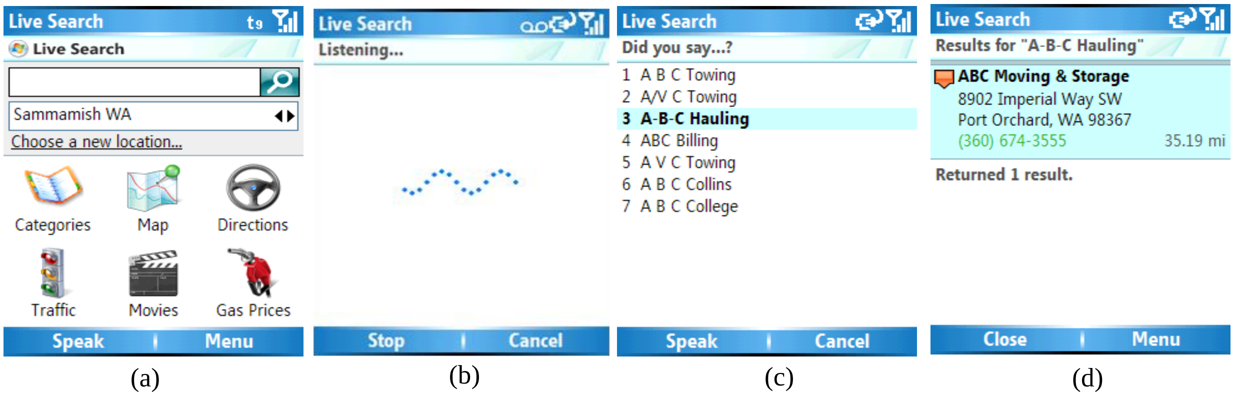
The proposed approach, described in detail in Section 3, draws inspiration from earlier work in post-recognition error-correction models (Ringger and Allen, 1996; Ringger and Allen, 1997) and n-best rescoring (Chotimongkol and Rudnicky, 2001; Birkenes et al., 2007). The novelty of our approach lies in: (1) the use of user click data in a deployed multi-modal system for creating a positive feedback loop, and (2) the development of an n-best correction model based on implicit feedback that outperforms traditional rescoring-only approaches. Later on, in Section 5, we will discuss in more detail the relationship of the proposed approach to these and other works previously reported in the literature.

Before moving on to describe the n-best correction model in more detail, we give a high-level overview of Live Search for Windows Mobile, the multi-modal, mobile local search application that provided the test-bed for evaluating this work.

2 Live Search for Windows Mobile

Live Search for Windows Mobile is an application that enables local web-search on mobile devices. In its current version, it allows users to find information about local businesses and restaurants, to obtain driving directions, explore maps, view current traffic, get movie show-times, etc. A number of screen-shots are illustrated in Figure 1.

Recently, Live Search for Windows Mobile has been extended with a speech interface (notice the Speak button assigned to the left soft-key in Figure 1.a.) The speech-based interaction with the system proceeds as follows: the user clicks the Speak button and speaks the name of a local business, for instance *A-B-C Hauling*, or a general category such as *Vietnamese Restaurants*. The application endpoints the audio and forwards it over the data channel to a server (Figure 1.b.) Recognition is performed on the server side, and the resulting n-best list is sent back to the client application, where it is displayed to the user (Figure 1.c.) The user can select the correct item from the n-best list, re-speak the request, or abandon the interaction altogether by pressing Cancel. Once the user selects an item in the n-best list, the corresponding search results are displayed (Figure 1.d.)



Did you say...?

1 A B C Towing
2 A/V C Towing
3 A-B-C Hauling
4 ABC Billing
5 A V C Towing
6 A B C Collins
7 A B C College

Results for "A-B-C Hauling"

ABC Moving & Storage
8902 Imperial Way SW
Port Orchard, WA 98367
(360) 674-3555 35.19 mi

Returned 1 result.

(a) (b) (c) (d)

Figure 1. Windows Live Search for Mobile. (a) initial screen; (b) user is speaking a request; (c) n-best list is presented; (d) final search results are displayed

Apart from business names, the system also handles speech input for addresses, as well as compound requests, such as *Shamiana Restaurant in Kirkland, Washington*. For the latter cases, a two-tier recognition and confirmation process is used. In the first stage a location n-best list is generated and sent to the client for confirmation. After the user selects the location, a second recognition stage uses a grammar tailored to that specific location to re-recognize the utterance. The client then displays the final n-best list from which the user can select the correct result.

Several details about the system architecture and the structure of the recognition process have been omitted here due to space considerations. For the interested reader, a more in-depth description of this system is available in (Acero et al., 2008).

3 Approach

We now turn our attention to the proposed n-best correction model

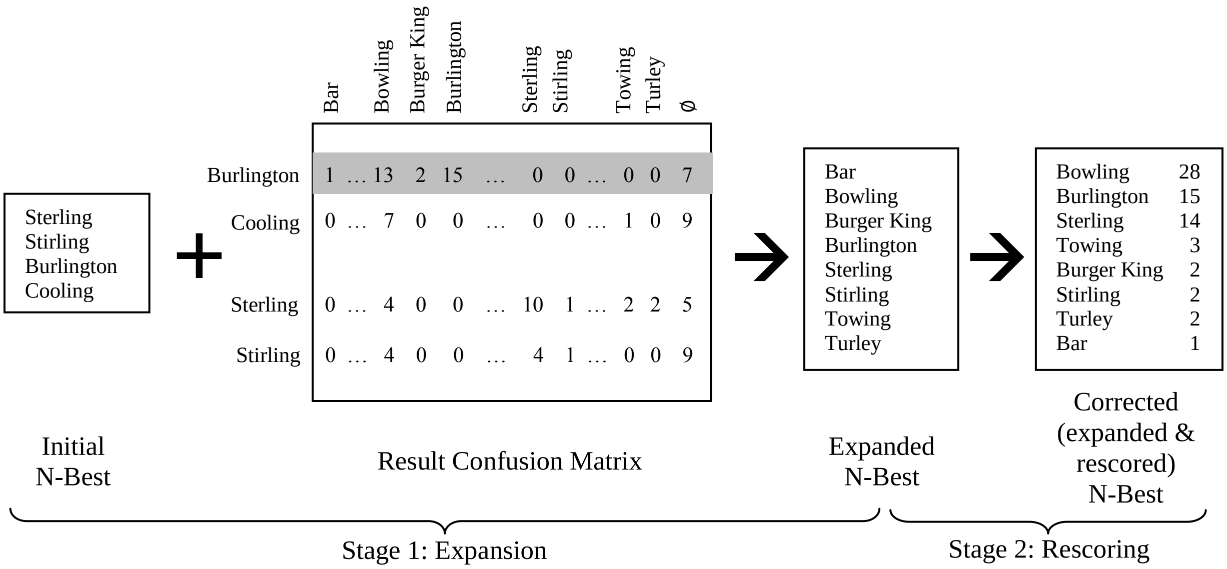
3.1 Overview

The model works in two stages, illustrated in Figure 2. In the first stage the n-best list produced by the speech recognizer is expanded with several alternative hypotheses. In the second stage, the expanded n-best list is rescored to construct the final, corrected n-best list.

The n-best expansion step relies on a result confusion matrix, constructed from click information. The matrix, which we will describe in more detail in the following subsection, contains information about which result was selected (clicked) by the user when a certain result was displayed. For instance, in the example from Figure 2, the matrix indicates that when *Burlington* appeared in the n-best list, *Bar* was clicked once, *Bowling* was clicked 13 times, *Burger King* was clicked twice, and *Burlington* was clicked 15 times (see hashed row in matrix.) The last element in the row indicates that there were 7 cases in which *Burlington* was decoded, but nothing (∅) was clicked. Essentially, the matrix captures information about the confusability of different recognition results.

The expansion step adds to an n-best list generated by the recognizer all the results that were previously clicked in conjunction with any one of the items in the given n-best list. For instance, in the example from Figure 2, the n-best list contains *Sterling, Stirling, Burlington* and *Cooling.* Based on the confusion matrix, this list will be expanded to also include *Bar, Bowling, Burger King, Towing,* and *Turley.* In this particular case, the correct recognition result, *Bowling,* is added in the expanded n-best list.

In the final step, the expanded list is rescored. In the previous example, for simplicity of explanation, a simple heuristic for re-scoring was used: add all the counts on the columns corresponding to each expanded result. As a consequence, the cor-



	Bar	…	Bowling	Burger King	Burlington	…	Sterling	Stirling	…	Towing	Turley	∅
Burlington	1	…	13	2	15	…	0	0	…	0	0	7
Cooling	0	…	7	0	0	…	0	0	…	1	0	9
Sterling	0	…	4	0	0	…	10	1	…	2	2	5
Stirling	0	…	4	0	0	…	4	1	…	0	0	9

Bowling	28
Burlington	15
Sterling	14
Towing	3
Burger King	2
Stirling	2
Turley	2
Bar	1

Figure 2. A confusion-based n-best correction model

rect recognition result, *Bowling*, was pushed to the top of the n-best list.

We begin by formally describing the construction of the results confusability matrix and the expansion process in the next two sub-sections. Then, we describe three rescoring approaches. The first one is based on an error-correction model constructed from the confusion matrix. The other two, are more traditional rescoring approaches, based on language model adaptation.

3.2 The Result Confusion Matrix

The result confusion matrix is computed in a simple traversal of the click logs. The rows in the matrix correspond to decoded results, i.e. results that have appeared in an n-best list. The columns in the matrix correspond to clicked (or intended) results, i.e. results that the user has clicked on in the n-best list. The entries at the intersection of row d and column c correspond to the number of times result c was clicked when result d was decoded:

$$m_{d,c} = \#(decoded = d, clicked = c).$$

In addition, the last column in the matrix, denoted $\emptyset$ contains the number of times no result was clicked when result d was displayed:

$$m_{d,\emptyset} = \#(decoded = d, clicked = \emptyset).$$

The rows in the matrix can therefore be used to compute the maximum likelihood estimate for the conditional probability distribution:

$$P_{ML}(c|d) = \frac{m_{d,c}}{\sum_c m_{d,c}}.$$

The full dimensions of the result confusion matrix can grow very large since the matrix is constructed at the result level (the average number of words per displayed result is 2.01). The number of rows equals the number of previously decoded results, and the number of columns equals the number of previously clicked results. However, the matrix is very sparse and can be stored efficiently using a sparse matrix representation.

3.3 N-Best Expansion

The first step in the proposed n-best correction model is to expand the initial n-best list with all results that have been previously clicked in conjunction with the items in the current n-best list.

Let's denote by $N = \{d_r\}_{r=1..n}$ the initial n-best list produced by the speech recognizer. Then, the expanded n-best list EN will contain all d_r, as well as all previously clicked results c such that there exists r with $m_{d_r,c} > 0$.

3.4 Confusion Matrix Based Rescoring

Ideally, we would like to rank the hypotheses in the expanded list EN according to $P(i|a)$, where i represents the intended result and a represents the acoustics of the spoken utterance. This can be rewritten as follows:

$$P(i|a) = \sum_d P(i|d) \cdot P(d|a). \qquad [1]$$

The first component in this model is an error-correction model $P(i|d)$. This model describes the conditional probability that the correct (or intended) result is i given that result d has been decoded. While this conditional model cannot be constructed directly, we can replace it by a proxy - $P(c|d)$, which models the probability that the result c will be clicked, given that result d was decoded. As mentioned earlier in subsection 3.2, this conditional probability distribution can be computed from the result confusion matrix. In replacing $P(i|d)$ with $P(c|d)$, we are making the assumption that the clicks correspond indeed to the correct, intended results, and to nothing else[1].

Notice that the result confusion matrix is generally very sparse. The maximum likelihood estimator $P_{ML}(c|d)$ will therefore often be inappropriate. To address this data sparsity issue, we linearly interpolate the maximum likelihood estimator with an overall model $P_O(c|d)$:

$$P(c|d) = \lambda P_{ML}(c|d) + (1-\lambda) P_O(c|d).$$

The overall model is defined in terms of two constants, α and β, as follows:

$$P_O(c|d) = \begin{cases} \alpha, if\ c = d \\ \beta, if\ c \neq d \end{cases}$$

where α is the overall probability in the whole dataset of clicking on a given decoded result, and β is computed such that $P_O(c|d)$ normalizes to 1.

[1] While this assumption generally holds, we have also observed cases where it is violated: sometimes users (perhaps accidentally) click on an incorrect result; other times the correct result is in the list but nothing is clicked (perhaps the user was simply testing out the recognition capabilities of the system, without having an actual information need)

Finally, the λ interpolation parameter is determined empirically on the development set.

The second component in the confusion based rescoring model from equation [1] is $P(d|a)$. This is the recognition score for hypothesis d. The n-best rescoring model from [1] becomes:

$$P(c|a) = \sum_{d_r \in N} [\lambda P_{ML}(c|d_r) + (1-\lambda)P_O(c|d_r)] \cdot P(d_r|a)$$

3.5 Language Model Based Rescoring

A more traditional alternative for n-best rescoring is to adapt the bigram language model used by the system in light of the user click data, and re-rank the decoded results by:

$$P(i|a) \propto P(d_r|a) \propto P(a|d_r)P(d_r)$$

Here $P(a|d_r)$ is the acoustic score assigned by the recognizer to hypothesis d_r, and $P(d_r)$ is the adapted language model score for this hypothesis.

A simple approach for adapting the system's language model is to add the word sequences of the user-clicked results to the original training sentences and to re-estimate the language model $P(d)$. We will refer to this method as maximum likelihood (ML) estimation. A second approach, referred to as conditional maximum likelihood (CML) estimation, is to adapt the language model such as to directly maximize the conditional likelihood of the correct result given acoustics, *i.e.*,

$$P(i|a) = \frac{P(a|i)P(i)}{\sum_{d_r \in N} P(a|d_r)P(d_r)}$$

Note that this is the same objective function as the one used in Section 3.4, except that here the click data is used to estimate the language model instead of the error correction model. Again, in practice we assume that users click on correct results, i.e. $i = c$.

4 Experiments

We now discuss a number of experiments and the results obtained using the proposed n-best correction approach.

4.1 Data

For the purposes of the experiments described below we extracted just over 800,000 queries from the server logs in which the recognizer had generated a simple n-best list[2]. For each recognition event, we collected from the system logs the n-best list, and the result clicked by the user (if the user clicked on any result).

In addition, for testing purposes, we also make use of 11529 orthographically transcribed user requests. The transcribed set was further divided into a development set containing 5680 utterances and a test set containing 5849 utterances.

4.2 Initial N-Best Rescoring

To tease apart the effects of expansion and rescoring in the proposed n-best correction model, we began by using the rescoring techniques on the initial n-best lists, without first expanding them.

Since the actual recognition confidence scores $P(d_r|a)$ were not available in the system logs, we replaced them with an exponential probability density function based on the rank of the hypothesis:

$$P(d_r|a) = 2^{-r}$$

We then rescored the n-best lists from the test set according to the three rescoring models described earlier: confusion matrix, maximum likelihood (ML), and conditional maximum likelihood (CML). We computed the sentence level accuracy for the rescored n-best list, at different cutoffs. The accuracy was measured by comparing the rescored hypotheses against the available transcripts.

Note that the maximum depth of the n-best lists generated by the recognizer is 10; this is the maximum number of hypotheses that can be displayed on the mobile device. However, the system may generate fewer than 10 hypotheses. The observed average n-best list size in the test set was 4.2.

The rescoring results are illustrated in Figure 3 and reported in Table 1. The X axis in Figure 3 shows the cutoff at which the n-best accuracy was computed. For instance in the baseline system, the correct hypothesis was contained in the top result in 46.2% of cases, in the top-2 results in 50.5% of the cases and in the top-3 results in 51.5% of the cases. The results indicate that all the rescoring models improve performance relative to the base-

[2] We did not consider cases where a false-recognition event was fired (e.g. if no speech was detected in the audio signal) – in these cases no n-best list is generated. We also did not consider cases where a compound n-best was generated (e.g. for compound requests like *Shamiana in Kirkland, Washington*)

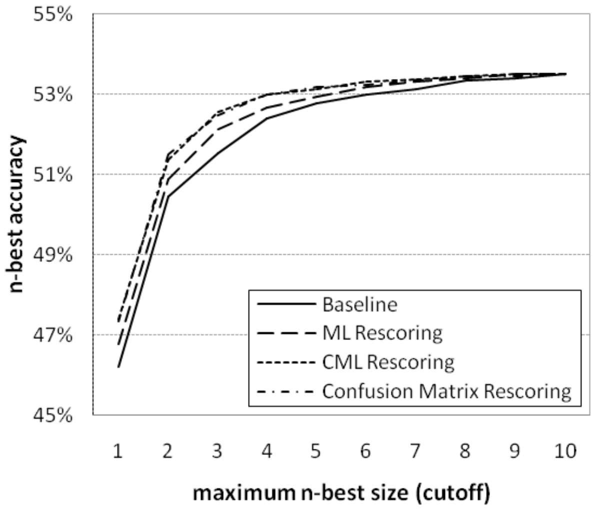


Figure 3. Initial n-best rescoring (test-set)

	Model	1-Best	2-Best	3-Best	10-Best
0	Baseline	46.2	50.5	51.5	53.5
1	ML Rescoring	46.8	50.9	52.1	53.5
2	CML Rescoring	47.4	51.4	52.6	53.5
3	Confusion Matrix Resc.	47.3	51.5	52.5	53.5
4	Expansion + Rescoring (size=7.09)	46.8	52.3	54.5	57.3
5	Expansion + Rescoring (size=4.15)	46.8	52.3	54.4	56.5

Table 1. Test-set sentence-level n-best accuracy;
(0) baseline; (1)-(3) initial n-best rescoring;
(4)-(5) expansion + rescoring

line. The improvement is smallest for the maximum likelihood (ML) language model rescoring approach, but is still statistically significant ($p = 0.008$ in a Wilcoxon sign-rank test.) The confusion-matrix based rescoring and the CML rescoring models perform similarly well, leading to a 1% absolute improvement in 1-best and 2-best sentence-level accuracy from the baseline ($p < 10^{-5}$). No statistically significant difference can be detected between these two models. At the same time, they both outperform the maximum likelihood rescoring model ($p < 0.03$).

4.3 N-Best Correction

Next, we evaluated the end-to-end n-best correction approach. The n-best lists were first expanded, as described in section 3.3, and the expanded lists were ranked using the confusion matrix based rescoring model described in Section 3.4.

The expansion process enlarges the original n-best lists. Immediately after expansion, the average n-best size grows from 4.2 to 96.9. The oracle performance for the expanded n-best lists increases to 59.8% (versus 53.5% in the initial n-best lists.) After rescoring, we trimmed the expanded n-best lists to a maximum of 10 hypotheses: we still want to obey the mobile device display constraint. The resulting average n-best size was 7.09 (this is lower than 10 since there are cases when the system cannot generate enough expansion hypotheses.)

The sentence-level accuracy of the corrected n-best lists is displayed in line 4 from Table 1. A direct comparison with the rescoring-only models or with the baseline is however unfair, due to the larger average size of the corrected n-best lists. To create a fair comparison and to better understand the performance of the n-best correction process, we pruned the corrected n-best lists by eliminating all hypotheses with a score below a certain threshold. By varying this rejection threshold, we can therefore control the average depth of the resulting corrected n-best lists. At a rejection threshold of 0.004, the average corrected n-best size is 4.15, comparable to the baseline of 4.2 .

The performance for the corresponding corrected (and pruned) n-best lists is shown in line 5 from Table 1 and illustrated in Figure 4. In contrast to a rescoring-only approach, the expansion process allows for improved performance at higher depths in the n-best list. The maximum n-best performance (while keeping the average n-best size at 4.15), is 56.5%, a 3% absolute improvement over the baseline ($p < 10^{-5}$).

Figure 5 provides more insight into the relationship between the sentence-level accuracy of the corrected (and pruned) n-best lists and the average n-best size (the plot was generated by varying the rejection threshold.) The result we discussed above can also be observed here: at the same average n-best size, the n-best correction model significantly outperforms the baseline. Furthermore, we can see that we can attain the same level of accuracy as the baseline system while cutting the average n-best size by more than 50%, from 4.22 to 2. In the opposite direction, if we are less sensitive to the number of items displayed in the n-best list (except for the 10-maximum constraint we already obey), we can further increase the overall performance by another 0.8% absolute to 57.3%; this overall accuracy is attained at an average n-best size of 7.09.

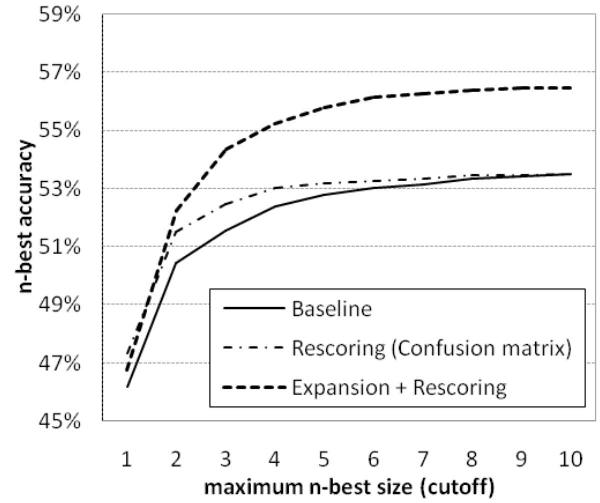


Figure 4. N-Best correction (test-set)

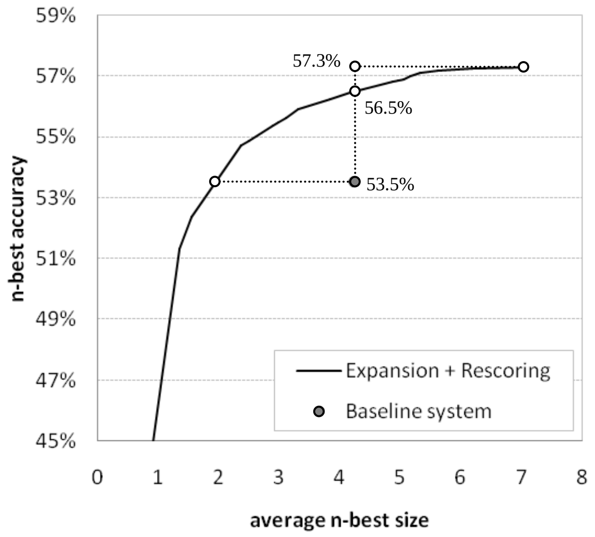


Figure 5. Overall n-best accuracy as a function of the average n-best size

Finally, we also investigated rescoring the expanded n-best lists using the CML approach. To apply CML, an initial ranking of the expanded n-best lists is however needed. If we use the ranking produced by the confusion-matrix based model discussed above, no further performance improvements can be observed.

5 Related work

The n-best correction model we have described in this paper draws inspiration from earlier works on post-recognition error correction models, n-best rescoring and implicitly supervised learning. In this section we discuss some of the similarities and differences between the proposed approach and previous work.

The idea of correcting speech recognition errors in a post-processing step has been proposed earlier by (Ringger and Allen, 1996; Ringger and Allen, 1997). The authors showed that, in the presence of transcribed data, a translation-based post-processor can be trained to correct the results of a speech recognizer, leading to a 15% relative WER improvement in a corpus of TRAINS-95 dialogues.

The n-best correction approach described here is different in two important aspects. First, instead of making use of transcripts, the proposed error-correction model is trained using implicit user feedback obtained in a multi-modal interface (in this case user clicks in the n-best list.) This is a less costly endeavor, as the system automatically obtains the supervision signal directly from the interaction; no transcripts are necessary. Second, the approach operates on the entire n-best list, rather than only on the top hypothesis; as such, it has additional information that can be helpful in making corrections. At Figure 2 illustrates, there is a potential for multiple incorrect hypotheses to point towards and reinforce the same correction hypothesis, leading to improved performance (in this example, *Burlington, Cooling, Sterling* and *Stirling* were all highly confusable with *Bowling*, which was the correct hypothesis).

The n-best correction model we have described includes a rescoring step. N-best rescoring approaches have been investigated extensively in the speech recognition community. In the dialog community, n-best rescoring techniques that use higher-level, dialog features have also been proposed and evaluated (Chotimongkol and Rudnicky, 2001). Apart from using the click feedback, the novelty in our approach lies in the added expansion step and in the use of an error-correction model for rescoring. We have seen that the confusability-based n-best expansion process leads to significantly improved performance, even if we force the model to keep the same average n-best size.

Finally, the work discussed in this paper has commonalities with previous works on lightly supervised learning in the speech community, e.g. (Lamel and Gauvain, 2002) and leveraging implicit feedback for learning from interaction, e.g. (Banerjee and Rudnicky, 2007; Bohus and Rudnicky, 2007). In all these cases, the goal is to minimize the need for manually-labeled data, and learn directly from the interaction. We believe that in the long term this family of learning techniques will play a key role towards building autonomous, self-improving systems.

6 Conclusion and future work

We have proposed and evaluated a novel n-best correction model that leverages implicit user feedback in a multi-modal interface to create a positive feedback loop. While the experiments reported here were conducted in the context of a local search application, the approach is applicable in any multi-modal interface that elicits selection in an n-best list from the user.

The proposed n-best correction model works in two stages. First, the n-best list generated by the speech recognizer is expanded with additional hypotheses based on confusability information captured from previous user clicks. This expanded list is then rescored and pruned to create a more accurate and more compact n-best list. Our experiments show that the proposed n-best correction approach significantly outperforms both the baseline and other traditional n-best rescoring approaches, without increasing the average length of the n-best lists.

Several issues remain to be investigated. The models discussed in this paper focus on post-recognition processing. Other ways of using the click data can also be envisioned. For instance, one approach would be to add all the clicked results to the existing language model training data and create an updated recognition language model. In the future, we plan to investigate the relationship between these two approaches, and to whether they can be used in conjunction. Earlier related work (Ringger and Allen, 1997) suggests that this should indeed be the case.

Second, the click-based error-correction model we have described in section 3.4 operates at the result level. The proposed model is essentially a sentence level, memory-based translation model. In the future, we also plan to investigate word-level error-correction models, using machine translation techniques like the ones discussed in (Ringger and Allen, 1997; Li et al., 2008).

Finally, we plan to investigate how this process of learning from implicit feedback in a multi-modal interface can be streamlined, such that the system continuously learns online, with a minimal amount of human intervention.

Acknowledgments

This work would have not been possible without the help of a number of other people. We would like to especially thank Oliver Scholz, Julian Odell, Christopher Dac, Tim Paek, Y.C. Ju, Paul Bennett, Eric Horvitz and Alex Acero for their help and for useful conversations and feedback.

References

Acero, A., N. Bernstein, et al. (2008). "Live Search for Mobile: Web Services by Voice on the Cellphone". ICASSP'08. Las Vegas, NV.

Banerjee, S. and A. Rudnicky (2007). "Segmenting Meetings into Agenda Items by Extracting Implicit Supervision from Human Note-Taking". IUI'2007. Honolulu, Hawaii.

Birkenes, O., T. Matsui, et al. (2007). "N-Best Rescoring for Speech Recognition using Penalized Logistic Regression Machines with Garbage Class". ICASSP'2007, Honolulu, Hawaii.

Bohus, D. and A. Rudnicky (2007). "Implicitly-supervised learning in spoken language interfaces: an application to the confidence annotation problem". SIGdial 2007, Antwerp, Belgium.

Chotimongkol, A. and A. Rudnicky (2001). "N-best Speech Hypotheses Reordering Using Linear Regression". Eurospeech'2001, Aalborg, Denmark.

Lamel, L. and J.-L. Gauvain (2002). "Lightly Supervised and Unsupervised Acoustic Model Training." Computer Speech and Language 16: 115-129.

Li, X., Y.-C. Ju, et al. (2008). "Language Modeling for Voice Search: a Machine Translation Approach". ICASSP'08, Las Vegas, NV.

Live Search for Windows Mobile (2008):
http://mobile.search.live.com

Nuance Mobile Search (2008):
http://www.nuance.com/mobilesearch.

Ringger, E. and J. Allen (1996). "Error Correction via Post-Processor for Continuous Speech Recognition". ICASSP'96, Atlanta, GA.

Ringger, E. and J. Allen (1997). "Robust Error Correction of Continuous Speech Recognition". ESCA-NATO Workshop on Robust Speech Recognition for Unknown Communication Channels, Pont-a-Mousson, France.

TellMeByMobile (2008):
http://www.tellme.com/products/tellmebymobile.

V-Lingo Mobile. (2008):
http://www.vlingomobile.com/downloads.html.

VoiceSignal Search. (2008):
http://www.voicesignal.com/solutions/vsearch.php.