Multi-channel Sentence Classification for Spoken Dialogue Language Modeling

Frédéric Béchet †, *Giuseppe Riccardi* ‡, *Dilek Z. Hakkani-Tür* ‡

† LIA, University of Avignon BP1228 84911 Avignon Cedex 09, FRANCE
‡ AT&T Labs 180 Park Avenue, Florham Park, New Jersey 07932, USA

frederic.bechet@lia.univ-avignon.fr   dsp3@research.att.com   dtur@research.att.com

Abstract

In traditional language modeling word prediction is based on the local context (e.g. n-gram). In spoken dialog, language statistics are affected by the multidimensional structure of the human-machine interaction. In this paper we investigate the statistical dependencies of users' responses with respect to the system's and user's channel. The system channel components are the prompts' text, dialogue history, dialogue state. The user channel components are the Automatic Speech Recognition (ASR) transcriptions, the semantic classifier output and the sentence length. We describe an algorithm for language model rescoring using users' response classification. The user's response is first mapped into a multidimensional state and the state specific language model is applied for ASR rescoring. We present perplexity and ASR results on the *How May I Help You* $?^{sm}$ 100K spoken dialogs.

1. Introduction

In statistical language modeling for automatic speech recognition (ASR) words are effectively predicted by local lexical contexts (e.g. n-gram) of the user channel. In spoken dialog system, the generalization of lexical contexts extends to include lexical and syntactic features from the machine channel. In a human-machine interaction the machine asks questions to elicit a specific response carrying *expected* information. Within a human-machine dialog questions range from open ended (e.g. *How May I Help You?*) to narrow focus (*What is your telephone number?*) prompts. While traditionally language models have been trained by looking at within channel dependencies (user's n-gram statistics) in this paper we propose an algorithm to investigate cross channel dependencies between user's and machine's language channel.

From a language modeling perspective, each *state* s_i of the dialog should have associated a corresponding language model λ_i that has the lexical statistics skewed to account for the specific dialog contexts. While this is the optimal parameter setting, there is a serious data sparseness problem due to data fragmentation of spoken dialog corpora. In Fig. 1 we show the histogram of the prompts for the *How May I Help You?* corpus [1, 2]. On the x-axis we plot the bins for each system prompt. The data sparseness problem has been solved in the past by adapting *background* language models to context dependent features [3, 4, 5]. In these papers the user channel is the only source for word prediction in the users' responses. In this paper we propose a cross channel clustering algorithm to exploit both within and across channel language dependencies. In the next sections we review the background work on language clustering, the user and system channel features and in section 6 we report on the ASR rescoring experiments on the *How May I Help You?* spoken dialog corpus.
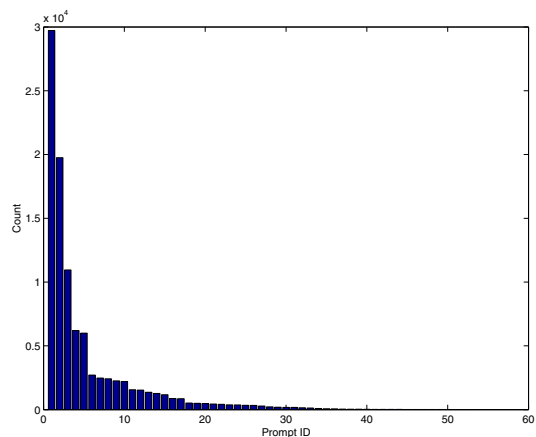

Figure 1: Histogram of users' responses in the "How May I Help You" spoken dialog system. Each bin (x-axis) corresponds to one of the system prompts.

2. The System Channel

These are the features, on the system channel side, used in this study and attached to each user's utterance:

- **prompt text**: this is the word string uttered by the system before each user's utterance;

- **prompt category**: prompt category according to the kind of information requested (`conf` if the prompt asks for a confirmation, `numerical` if the information requested is a numerical value like a phone number, `other` in all the other cases);

- **dialogue state**: a label given by the Dialogue Manager characterizing the current state of the dialogue;

- **dialogue history**: the string of dialogue state labels given by the Dialogue Manager during the previous turns of the same dialogue

3. The User Channel

The user channel features are:

- **transcriptions**: human transcriptions for the training corpus and ASR transcriptions for the test corpus;

- **utterance lengths**: the utterance lengths are estimated on the transcriptions (human or automatic) and represented by a set of discrete symbols `l0` for less than 5 words, `l1` between 5 and 10 words, `l2` between 10 and 15 and `l3` for more than 15 words);

- **Spoken Language Understanding tags** (SLU tags): the semantic tags used by the SLU module for representing the meaning of a sentence given by the semantic classifier or the human labelers.

4. The Multidimensional Clustering

Each turn of the spoken dialogue corpus used for the clustering process is represented by a multidimensional structure containing all the features previously presented both on the user and system channels. The clustering process consists of splitting the whole training corpus into groups of utterances sharing common properties on all the dimensions represented.

This process is performed by an unsupervised classification algorithm based on an optimization criterion that has a direct influence on the recognition process: the *perplexity* measure of the Language Model (LM) on the manually transcribed training corpus. We decide to use this criterion because even if there is no evidence that a gain in perplexity will result in a Word Error Rate (WER) reduction, these two quantities are generally related.

The multidimensional clustering algorithm chosen is based on a decision-tree approach inspired by the Semantic Classification Tree method proposed in [6] and already used for corpus clustering in [7]. One originality of this kind of decision tree is the dynamic generation of questions during the growing process of a tree. This is presented in the next section.

4.1. Question generation

The questions used in the growing process of the decision tree are related to the features contained in the multidimensional structure representing each user's utterance in the training corpus. There are two kinds of questions:

Firstly there are the *simple* questions that check if a given feature can apply to a user's utterance. Among these features one can find the prompt category, the dialogue state, the utterance length and the SLU tags.

Secondly, the *complex* questions are related to features that can be split into an ordered list of tokens. It's not the whole sequence of tokens that is going to be looked for in a user's utterance but a possible match with a regular expression dynamically built, on the whole sequence. The prompt text and the dialogue history belong to this category.

For example, prompt texts can be split into an ordered list of words. Because the dialogue corpus we use has been collected over a long period of time with several versions of the dialogue system, the texts of the different prompts have evolved. It is therefore interesting to try to match the various prompts between each other not just on their whole word strings but on some patterns (or regular expressions) characteristic of the queries addressed to the user.

The dynamic generation of these patterns is made as follows:

- at the initialization step, the *complex* questions contained in the root node are empty regular expressions coded `<+>` where the symbol + indicates a *gap* corresponding to the occurrence of at least one token;

- during the growing process of the tree, all regular expressions contained in a given node are augmented by replacing each symbol + by a token w| in four different contexts: w, +w, +w+, w+ (for example, if the regular expression contains 1 gap and if the corresponding feature has a vocabulary of n tokens, n*1*4 expressions are going to be generated);

- each regular expression generated at a given node in the tree corresponds to one question that can be applied to the corpus attached to the node and if one of these questions is selected as the best one according to the optimization criterion, then the regular expression attached to the node is replaced by this new one.

For example, if the regular expression of the feature *prompt text* at a given node is: `<Please +>` and if the question chosen is `<+ give +>` on the right gap of the expression, the new expression attached to the node will be: `<Please + give +>` which will match all the prompts that start with the word `Please` and have the word `give` in the rest of the sentence.

4.2. Tree growing algorithm

The tree growing algorithm consists of a top-down process that splits recursively a root node containing the whole training corpus into two nodes and two clusters according to the answer to a question chosen for maximizing the optimization criterion: the perplexity measure.

At the beginning a development corpus is extracted from the whole training corpus. This corpus contains about 20% of the original one. Then these two corpora are attached to the root node of the tree and a bigram Language Model is trained on the training corpus.

During the growing process, each simple or complex question q applied to the training and development corpora attached to a node i splits both of them into two sub-corpora:

- $Train(i, q, yes)$ and $Devt(i, q, yes)$ which contain all the utterances satisfying question q;

- $Train(i, q, no)$ and $Devt(i, q, no)$ containing the utterances which don't match question q.

Two new LM bigrams are then trained on $Train(i, q, yes)$ and $Train(i, q, no)$: $LM(i, q, yes)$ and $LM(i, q, no)$.

Let's note $LM(i)$ the LM trained on the training corpus attached to the node i, and $PP(LM, Devt)$ the perplexity measure of the language model LM on the development corpus $Devt$. The best question $Q(i)$ is then chosen according to equation 1.

$$Q(i) = \max_q(Gain(i, q, yes) + Gain(i, q, no)) \quad (1)$$

with:

$$Gain(i, q, a) = PP(LM(i), Devt(i, q, a)) - PP(LM(i, q, a), Devt(i, q, a))$$

Once the question that maximizes the gain in perplexity between the LM trained on the whole corpus and those trained on the sub-corpora is chosen, this question is attached to the node and the process is recursively applied to the two nodes corresponding to the two sub-corpora obtained. This process stops when no gain in perplexity can be achieved by specializing more the LMs or when the size of the sub-corpora is too small for training a new LM (according to a given threshold).

Figure 2 shows an example of such a tree. In this example, the only questions allowed were about the sentence lengths (user channel) and the text prompts (system channel). The top question chosen is `<+starting+>` which matches all the utterances of the training corpus uttered after a prompt containing the word `starting`. All the matching prompts request a phone
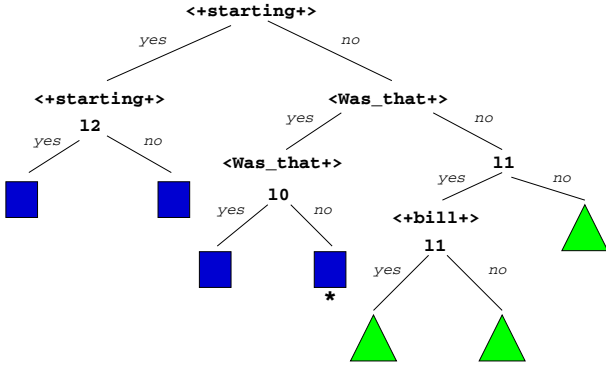
Figure 2: Example of clustering tree on the prompt texts and the sentence lengths

number starting with the area code, like: *Please give me your phone number starting with the area code*, or *I need the dialed number starting with the area code*. It's interesting to notice that this method groups together prompts corresponding to the same kind of request, regardless of the way the prompts are expressed, and in an automatic way. Moreover, if a new prompt is added to the system, it can be directly handled by the tree.

5. The Rescoring Paradigm

Once the clustering-tree is grown on the training corpus, each leaf (represented in figure 2 by a box) contains two corpora: a training and a development one. A new bigram LM is trained on both of them and attached to the leaf. Each LM corresponds to a specific dialogue context described by the list of questions (on the user channel and system channel features) found in the path from the root to the leaf.

The rescoring process of an utterance, illustrated by figure 3 consists of the following steps:

- the system channel information (prompt text, prompt type, dialogue state and dialogue history) is first attached to the utterance;

- the utterance is decoded by means of a general trigram LM in order to output a word graph as well as the 1-best word string;

- the sentence length is calculated on the 1-best string and the SLU tags are estimated from the word graph as presented in [8];

- starting at the root level, the clustering-tree is traversed by answering to all the questions contained in the nodes of the tree on a path from the root node to a leaf; for example the leaf of figure 2 which is marked with an * can be reached by the following dialogue context: all the utterances that have been uttered after a system prompt which don't contain the word *starting* but which starts with the words *Was that* and which have more than 5 words;

- finally, the LM attached to the leaf reached after traversing the tree is applied to the word graph and a new 1-best string is output.
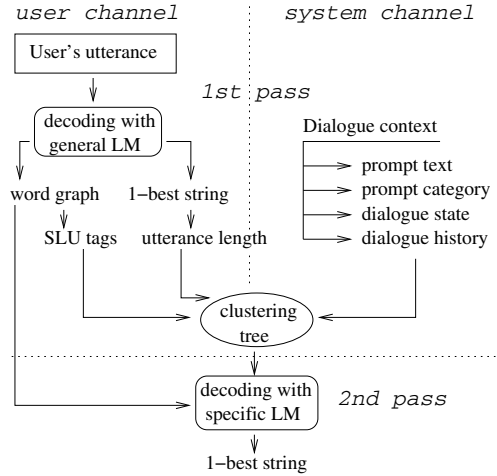

Figure 3: Rescoring process

6. Experiments

The training corpus used to grow the clustering-tree comprises about 102k utterances from live customer traffic of the HMIHY application. The test corpus was made of 7k utterances.

In the first experiment, the user channel does not contain any SLU tags. The system channel contains all the information presented in section 2. By keeping only the leaves in the tree with a corpus containing at least 14k words (representing about 1% of the words of the global training corpus) we obtained the 6 clusters represented in table 1.

6.1. Clusters description

The size of each cluster is calculated according to the number of words of all the utterances belonging to it. This number is expressed as a percentage of the total number of words of the training corpus (column *% words* of table 1). One can notice that the clusters size is not homogeneous. Indeed more than 70% of the words of the training corpus are in the same cluster. This result is not surprising according to figure 1. Indeed the open ended prompts like *How may I help you ?* represent a very large chunk of all the possible prompts and moreover most of the answers to these prompts are quite long with more than 15 words. It is therefore very difficult to split a chunk where all the utterances share the same characteristics.

It is interesting to see which information is considered relevant by the tree for splitting the training corpus. These 6 clusters contain the following utterances:

- **C1**: answers to a prompt asking for a phone number and containing between 10 and 15 words;

- **C2**: answers to the confirmation prompt *Are you calling from your home phone ?* containing between 5 and 10 words;

- **C3**: answers to the same confirmation prompt containing less than 5 words;

- **C4**: answers to other prompts and containing between 5 and 10 words;

- **C5**: answers to other prompts and containing between 10 and 15 words;

- **C6**: answers to other prompts and containing more than 15 words;

As we can see, 3 kinds of interactions are distinguished: request for a phone number, request for confirmation and other. These interactions correspond to the different types of system prompts defined in section 2.

However, it is interesting to notice that firstly it's always a specific prompt and not the global tag *numeric*, *conf* or *other* which is chosen by the tree, and secondly that an utterance length is systematically attached to each prompt. This means that these prompts (which are very frequent) have their own behaviors independently from the other prompts part of the same prompt category.

It means also that utterance lengths are very strong and reliable indicators for characterizing an answer to a given prompt. It is very likely that cluster 1 contains phone numbers, cluster 2 contains confirmation answers with explanation (mostly negative), and cluster 3 contains confirmation answers without explanation (mostly positive). It is also surprising to notice that no dialogue state label or dialogue history label was chosen by the tree in the clustering process. One possible explanation is the limited length of the dialogues in the HMIHY corpus (4 turns on average). Therefore the dialogue context and history are negligible compared to the system prompt alone.

6.2. ASR results

		Perplexity %		WER %	
C	% words	1-pass	2-pass	1-pass	2-pass
1	1.8	18.6	13.9	11.3	11.1
2	1.3	5.0	3.2	14.5	12.5
3	1.2	3.2	1.5	4.4	2.5
4	4.7	11	7.4	19.2	18
5	13.8	11.3	9.5	19.7	18.8
6	73.9	38.4	27.4	30.8	29.8

Table 1: Automatic clustering on both system and user channel without the SLU tags

The results presented in table 1 show significant perplexity improvement for all the clusters between the first and the second pass. Even if the decrease in WER is not as significant, we obtain a gain for all of them.

The second experiment consists of adding to the user channel the SLU tags. We decided, as a first step, to use manual tags in order to check the potential of this feature independently from the classification errors made by the SLU module.

The idea here is to check if knowing in advance what the user is going to say (at a semantic level) can help the clustering process. The results presented in Table 2 confirm this hypothesis: a decrease in perplexity and WER is observed for the 8 clusters obtained with this method. However, adding SLU tags didn't help to further split the main cluster containing all the remaining sentences (cluster #8).

Therefore, the global decrease of WER in table 2 is equal to the one obtained in table 1 without the SLU tags. Considering that the gain obtained by means of the SLU tags is not big enough to overcome the inevitable SLU misclassification that will occur by using an automatic tagging process, we don't consider keeping these tags in the features used in the clustering process.

7. Conclusions

In this paper we have addressed the problem of language modeling in the context of human-machine spoken dialog. We have

		Perplexity %		WER %	
C	% words	1-pass	2-pass	1-pass	2-pass
1	1.4	34.9	32.0	29.7	29.6
2	2.0	18.8	14.1	11.2	10.3
3	1.7	10.5	8.0	18.0	17.2
4	1.1	4.0	2.4	10.4	8.2
5	1.2	3.0	1.3	3.5	2.1
6	2.0	5.4	4.4	11.6	11.9
7	8.2	15.1	12.0	24.0	22.4
8	72.7	38.3	27.3	30.8	29.8

Table 2: Automatic clustering on all the features with manual SLU tags

investigated the statistical dependencies of users' responses with respect to the system and user language channel. The system channel features include the prompts' text, dialogue history, dialogue state. The user channel components are the Automatic Speech Recognition (ASR) transcriptions, the semantic classifier output and the sentence length. In order to exploit the joint channel language model, a two-step rescoring algorithm performs user response classification (first pass) and ASR decoding on refined language model (second pass). We have shown the perplexity and ASR results on the *How May I Help You ?^{sm}* 100K spoken dialogs.

8. References

[1] A. Gorin, J.H. Wright, G. Riccardi, A. Abella, and T. Alonso, "Semantic information processing of spoken language," in *Proc. of ATR Workshop on Multi-Lingual Speech Communication*, 2000.

[2] A.L. Gorin, G. Riccardi, and J.H. Wright, "How May I Help You?," *Speech Communication*, vol. 23, pp. 113–127, 1997.

[3] G. Riccardi, A. Potamianos, and S. Narayanan, "Language model adaptation for spoken dialog systems," in *Proceedings of the International Conference on Spoken Language Processing (ICSLP-98)*, Sidney, 1998.

[4] W. Xu and A. Rudnicky, "A. language modeling for dialog system," in *Proceedings of the International Conference on Spoken Language Processing (ICSLP-98)*, Beijing, 2000.

[5] G. Riccardi and A. G. Gorin, "Spoken language adaptation over time and state in a natural spoken dialog system," *IEEE Trans. on Speech and Audio Processing*, vol. 8, pp. 3–10, 2000.

[6] T. Kemp and A. Waibel, "Learning to recognize speech by watching television," *IEEE Intelligent Systems*, vol. 17, pp. 51–58, 1999.

[7] Y. Esteve, F. Bechet, A. Nasr, and R. De Mori, "Stochastic finite state automata language model triggered by dialogue states," in *Proc. of 7th European Conference on Speech Communication and Technology*, 2001, pp. 725–728.

[8] G. Tur, J. H. Wright, A. L. Gorin, G. Riccardi, and D. Hakkani-Tur, "Improving spoken language understanding using word confusion networks," in *Proceedings of the International Conference on Spoken Language Processing (ICSLP-98)*, Denver, 2002.