Mining Spoken Dialogue Corpora for System Evaluation and Modeling

Frederic Bechet
LIA-CNRS
University of Avignon, France
frederic.bechet@
lia.univ-avignon.fr

Giuseppe Riccardi
AT&T Labs
Florham Park, NJ, USA
dsp3@
research.att.com

Dilek Hakkani-Tur
AT&T Labs
Florham Park, NJ, USA
dtur@
research.att.com

Abstract

We are interested in the problem of modeling and evaluating spoken language systems in the context of human-machine dialogs. Spoken dialog corpora allow for a multidimensional analysis of speech recognition and language understanding models of dialog systems. Therefore language models can be directly trained based either on the dialog history or its equivalence class (or cluster). In this paper we propose an algorithm to mine dialog traces which exhibit similar patterns and are identified by the same class. For this purpose we apply data clustering methods to large human-machine spoken dialogue corpora. The resulting clusters can be used for system evaluation and language modeling. By clustering dialog traces we expect to learn about the behavior of the system with regards to not only the automation rate but the nature of the interaction (e.g. easy *vs* difficult dialogs). The equivalence classes can also be used in order to automatically adapt the language model, the understanding module and the dialogue strategy to better fit the kind of interaction detected. This paper investigates different ways for encoding dialogues into multidimensional structures and different clustering methods. Preliminary results are given for cluster interpretation and dynamic model adaptation using the clusters obtained.

1 Introduction

The deployment of large scale automatic spoken dialog systems, like *How May I Help You?*^{SM} (HMIHY) (Gorin et al., 1997), makes available large corpora of *real* human-machine dialog interactions. Traditionally, this data is used for supervised system evaluation. For instance, in (Kamm et al., 1999) they propose a static analysis aimed at measuring the performance of a dialog system, especially in an attempt to automatically estimate user satisfaction. In (Hastie et al., 2002), a dynamic strategy in the error handling process is proposed. In all these studies, supervised learning techniques are used in order to classify dialogs to predict user satisfaction or dialog failures.

A novel approach to the exploitation of dialog corpora is for speech recognition and language understanding modeling. In fact, such corpora allow for a multidimensional analysis of speech and language models of dialog systems. Our work differs from previous studies in the algorithmic approach and learning scenario.

First we use unsupervised speech mining techniques. We apply data clustering methods to large spoken dialog corpora. Two kinds of clustering methods are used: a hierarchical one based on decision trees and the optimization of a statistical criterion; a partitional one based on a k-means algorithm applied to vectors representing the dialogs. We *interpret* the clusters obtained and define a label for each of them.

Second we perform our analyses on large corpora of real dialogs collected from deployed systems. These log files contain a trace of the interaction between the users and a particular system at a certain point in time. Our goal is to highlight the structures behind these traces.

Lastly, we investigate several ways of encoding a dialog into a multidimensional structure in order to apply data clustering methods. Information about the system channel and the user channel are discussed and two ways of encoding are proposed, one for hierarchical clustering and the other for partitional clustering.

The clusters obtained can be used to learn about the behavior of the system with regards to the automation rate and the type of interaction (e.g. easy *vs* difficult dialog). Moreover, the clusters can be used on-the-fly to automatically adapt the language model, the understanding module and even the dialog strategy to better fit the kind of interaction detected.

In this study, we present two levels of clustering: clustering at the utterance level and the

- **System:** How may I help you?
- **User:** Hello?
- *Call-type: Hello*
- **System:** Hello, how may I help you?
- **User:** I have a question.
- *Call-type: Ask(Info)*
- **System:** OK, What is your question?
- **User:** I would like to know my account balance.
- *Call-type: Request(Account_Balance)*
- **System:** I can help you with that. What is your account number?
- **User:** ...

Figure 1: An example dialog from our corpus.

dialog level. Clustering at the utterance is for modeling the language people use in a specific dialog context; clustering at the dialog level allows us to characterize the whole interaction between the users and a system. In the next section we describe the corpora data structure. In section 3 we describe the clustering algorithms. In sections 4 and 5 we report on experiments and results for utterance-based and dialog clustering, respectively.

2 Dialog corpora

The corpora is collected from an automatic call routing system where the aim is to transfer the user to the right route in a large call center. An example dialog from a customer care application is given in Figure 1. After the greeting prompt, the speaker's response is recognized using an automatic speech recognizer (ASR). Then, the intent of the speaker is identified from the recognized sequence, using a spoken language understanding (SLU) component. This step can be framed as a classification problem, where the aim is to classify the intent of the user into one of the predefined call-types (Gorin et al., 1997). Then, the user would be engaged in a dialog via clarification or confirmation prompts until a final route is determined.

3 Hierarchical and Partitional clustering

The goal of clustering is to reduce the amount of data by categorizing or grouping similar data items together. Clustering methods can be divided into two basic types: hierarchical and partitional clustering. A lot of different algorithms have been proposed for both types of clustering methods in order to split a data set into clusters.

Hierarchical clustering proceeds successively by either merging smaller clusters into larger ones, or by splitting larger clusters. The result of the algorithm is a tree of clusters. By cutting the tree at a desired level a clustering of the data items into disjoint groups is obtained. We use in this study a decision-tree based clustering method.

Partitional clustering, on the other hand, attempts to directly decompose the data set into a set of disjoint clusters. A criterion function is used in order to estimate the distance between the samples of the different clusters. By minimizing this function between the samples of the same clusters and maximizing it among the different clusters, the clustering algorithm iteratively finds the best cluster distribution according to the criteria used. We use in this study a k-means algorithm applied to vectors encoding the dialogs. The number of clusters is fixed in advance.

4 Clustering at the utterance level

Performing clustering at the utterance level in a dialog corpus aims to capture different kinds of language that people would use in a specific dialog context. This is a way of grouping together turns of dialogs belonging to completely different requests but sharing some common properties according to their dialog contexts.

The immediate application of such a study can be the training of specialized Language Models (LMs) that can be used in replacement of a generic one once a specific situation is detected.

4.1 Decision-tree based clustering

In order to obtain utterance clusters from which we can build LMs we use a decision tree method based on an optimization criterion that has a direct influence on the recognition process: the *perplexity* measure of the Language Model on the manually transcribed training corpus. We decide to use this criterion because even if there is no evidence that a gain in perplexity results in a Word Error Rate (WER) reduction, these two quantities are generally related.

The clustering algorithm chosen is based on a decision-tree approach inspired by the Semantic Classification Tree method proposed in (Kuhn and Mori, 1995) and already used for corpus clustering in (Esteve et al., 2001) and (Bechet et al., 2003). One originality of this kind of decision tree is the dynamic generation of questions during the growing process of a tree.

4.2 Decision tree features

Each turn of the spoken dialog corpus used for the clustering process is represented by a multidimensional structure. Two kinds of channel can be considered in order to define the features: the system channel (which contains all the information managed by the system like the prompts or the states of the dialog) and the user channel (which contain the utterances uttered by the user with all their characteristics: length, vocabulary, perplexity, semantic calltypes, etc.). Because the clusters obtained are going to be used dynamically by training specific LMs on each of them, we used mostly system channel features in these experiments. On the HMIHY corpus we used the following features:

- **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);
- **dialog state**: a label given by the Dialog Manager characterizing the current state of the dialog;
- **dialog history**: the string of dialog state labels given by the Dialog Manager during the previous turns of the same dialog
- **utterance lengths**: the utterance lengths are estimated on the transcriptions (manual 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);

The only feature that does not belong to the system channel is the *utterance lengths*. This feature is part of the user channel but it can be estimated rather accurately from the word graph produced during the speech recognition process.

4.3 Results on the hierarchical clustering

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

After the clustering process we obtained the 6 clusters represented in table 1.

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 cluster sizes are not homogeneous. Indeed more than 70% of the words of the training corpus are in the same cluster. This result is not surprising: 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 the features considered relevant in the tree splitting of the training corpus. These 6 clusters contain the following type of 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 phoning 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 interaction 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 4.2.

It is interesting to notice that first it is always a specific prompt and not the prompt category *numeric*, *conf* or *other* which is chosen by the tree, and second 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.

Utterance lengths are very strong and reliable indicators for characterizing an answer to a given prompt. Cluster 1 contains mostly phone numbers, cluster 2 contains confirmation answers with explanation (mostly negative), and cluster 3 contains confirmation answers without explanation (mostly positive). We observe that no dialog state label or dialog history label was chosen as feature by the tree in the clustering process. One possible explanation is the limited length of the dialogs in the HMIHY corpus (4 turns on average). Therefore the dialog context and history are negligible compared to the system prompt alone.

		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: Results for each cluster obtained with the hierarchical clustering method, at the utterance level, on the HMIHY corpus

By using these clusters for training specific LMs and by dynamically choosing a specific LM according to the dialog context for performing LM rescoring, we obtain the perplexity and Word-Error-Rate (WER) results of table 1 on the HMIHY test corpus. Significant perplexity improvement can be seen for all the clusters between the first and the second pass. On the whole test corpus, the perplexity drops from 25.3 to 18.5, so a relative improvement of 26.8%. On the speech recognition side, even if the decrease in WER is not as significant, we obtain a gain for all of them.

4.4 K-means clustering

In order to split the utterances into clusters more equally, we decided to use another clustering method based on a partitional k-means algorithm. In these experiments we do not try to explicitly optimize a specific criterion, like the perplexity, but we just want to group together utterances sharing common properties and put utterances that are very dissimilar into different clusters. Perplexity measures obtained with LMs trained on such clusters will tell us if this method splits the utterances according to the language used. The first step in this process is to encode the dialogs into vectors, one vector for each dialog.

4.5 Representing utterances as feature vectors

The decision-tree method used symbolic features in order to generate the questions that split the corpus. The k-means clustering method is purely numerical therefore we need here to encode dialog turns into numerical feature vectors. According to what we learned from the previous experiments we decide to put in the vectors only the semantic calltype labels. Indeed as 70% of the utterances share the same prompt and the same utterance length category, it did not seem relevant to us to put these features in the vectors as we are looking for clusters that are quite balanced in size. The calltype labels are relevant as we are interested here in clusters that can be semantically different.

Therefore, a feature vector representing an utterance is a first order statistic on the calltypes. A component value is the number of occurrences of the corresponding calltype within the utterance. We kept the 34 most frequent calltype labels in the training corpus in order to build the vectors. They cover over 96% of the calltype occurrences on the HMIHY training corpus.

4.6 Results on the partitional clustering

This experiment is made on the same application than the last one (HMIHY) but on another data set. The training corpus contains 10k dialogs and 35.5k utterances and the test corpus contains 1.4k dialogs and 5k utterances. The number of clusters is set to 5. This value has the advantages of both providing a good convergence to the k-means process and splitting rather equally the training corpus.

Table 2 illustrates the partitional clustering of utterances on this corpus. As we can see, the distribution of the total amount of words of the corpus among the clusters is much more even than the one obtained with the hierarchical clustering. The largest cluster contains only 38.5% of the words compared to 73.9% previously. To check if these clusters can be useful for training specific LMs, we first split the test corpus according to the clustering model estimated on the training data. We use here, to build the vectors encoding the test corpus, automatic calltypes calculated by the Spoken Language Understanding Module. Then we compare the perplexity measures obtained with a general LM trained on the whole training corpus and the one obtained with a specific LM adapted on the corresponding cluster. Table 2 shows these results: the gain in perplexity obtained is smaller than the one obtained with the other method. Indeed the total perplexity on the test corpus is 21.3 and the one obtained with the specific LMs is 17.8, so a gain of 16% compared to the 26.8% obtained previously. However this result is not surprising as the previous method was designed for specifically decreasing the perplexity measure.

			Perplexity	
C	**% words**	**utt. length**	**1-pass**	**2-pass**
1	8.6	5.5	16.9	12.6
2	20.3	17.8	25.5	21.0
3	14.5	11.6	21.6	18.2
4	38.5	7.5	19.6	17.2
5	18.1	8.6	22.8	18.6

Table 2: Results for each cluster obtained with the partitional clustering method, at the utterance level, on the HMIHY corpus

5 Clustering at the dialog level

As we have seen in the previous section, specific dialog situations (like those obtained with the hierarchical clustering) proved to be more efficient than the semantic channel (represented by the calltype labels) for clustering utterances in relation with the language used, at least from the perplexity point of view.

However, for clustering dialogs rather than utterances, the semantic channel is the main channel that we are going to use because in this case we want to characterize the whole interaction between a system and a user rather than just the language used.

For example, an utterance like *I want to pay my bill* can be used in very different dialog contexts: in a *standard* interaction if this request is expressed just after the opening prompt or at a different stage of the dialog. To capture this kind of phenomena, we have to cluster at the dialog level rather than the utterance level.

After describing how we encode dialogs into feature vectors in the next section, we present in section 5.2 some preliminary work on the interpretation of the clusters obtained.

5.1 Representing dialogs as feature vectors

We use here the partitional clustering method. Each dialog is represented by a vector containing three kinds of information representing the interaction:

1. the number of turns in the dialog: associated to other features this parameter can be a strong indicator that the dialog is going fine (associated with a lot of confirmations or item values) or that the interaction is difficult (lot of repetitions for example).
2. first order statistics data on the calltype labels: these are the 34 calltypes presented in section 2 and representing both application-specific requests (*Pay_Bill*) and dialog-based concepts like *Yes*, *No*, *I want to talk to somebody*, *Help*, etc....
3. second order statistics data on the calltype labels: we chose the bigrams of the previous calltypes that had the highest weighted Mutual Information and we store in the vectors their frequencies. These features allow us to observe certain patterns like *repetition of a request*, *request followed by a confirmation*, *people asking twice for a representative*, etc....

5.2 Analyzing the clusters obtained

The experiment reported here is made on the HMIHY corpus. The vectors used contain 55 components: 1 for the number of turns, 34 for the calltypes and 20 for the bigrams on the calltypes. The number of clusters to be found by the k-means clustering algorithm was set to 5. Firstly because as we want to give an interpretation to each cluster we need to keep a relatively small number of them. Secondly because this number leads to a fast convergence of the k-mean clustering process. The clustering model is obtained on the training corpus and applied to the test corpus. Table 3 shows the distribution of the dialogs among the clusters, on the training corpus, ranging from 35.4% for C_1 to 7% for C_2.

There are two ways of analyzing the clusters obtained: from the language point of view and from the semantic point of view. Table 3 illustrates the language channel features with the average amount of turns (**#turn**), the average utterance lengths (**utt. length**) and the perplexity measure (**pplex**). This perplexity measure is obtained with a 3-gram LM trained on the whole training corpus and applied to the manual transcriptions of each cluster, of the test and the training corpus.

The differences in utterance lengths are not as big as those observed in section 4.6. This is an indicator that, unlike the clustering at the utterance level, the clusters obtained represent similar dialog situation. It is the way the dialog progresses, the dialog pattern, rather than the theme of the dialog which distinguishes the clusters. The lengths of the dialogs and the perplexity measures are indicators of these different dialog patterns.

The results on the test corpus presented in Table 3 are obtained with automatic calltypes, completely unsupervised. The F-measure score on the detection of these calltypes on the test corpus is 75%. As one can see, having errors in the calltypes detected does not affect too much the characteristics of the clusters obtained.

Training corpus				
C	**% dialogs**	**#turn**	**utt. length**	**pplex**
1	35.4%	3.9	7.3	7.32
2	7%	3.2	11.7	9.9
3	22.3%	3.5	9	7.3
4	10.4%	3.9	12.4	9.5
5	25%	2.9	9.3	7.9

Test corpus				
C	**% dialogs**	**#turn**	**utt. length**	**pplex**
1	32.4%	4.0	8.5	17.2
2	7.4%	3.3	11	30.2
3	20.8%	3.5	7.5	13.2
4	12.0%	3.8	11.6	28.2
5	27.5%	2.8	8.3	17

Table 3: Language features attached to the dialog clusters obtained with the partitional method, at the dialog level, on the HMIHY corpus

In order to analyze the clusters on the semantic channel we plot the weighed Mutual Information, $wMI(c_i;t_j)$, between each cluster, c_i, and each vector component, t_j. This measure is estimated in the following way:

$$wMI(c_i;t_j) = P(c_i;t_j)log\frac{P(c_i;t_j)}{P(c_i)P(t_j)}$$

This plot is shown on Figure 2 for the HMIHY training corpus. By grouping together components corresponding to a phenomenon we want to observe we are able to characterize more closely each cluster. In the color-coded graph, x-axis corresponds to call-type unigrams or selected call-type pairs, y-axis corresponds to each cluster. The color of each rectangle shows the degree of correlation, determined by the weighted mutual information between call-type and the cluster. As also can be seen from the color spectrum on the right hand side, dark red means high correlation (top), and dark blue means reverse correlation (bottom).

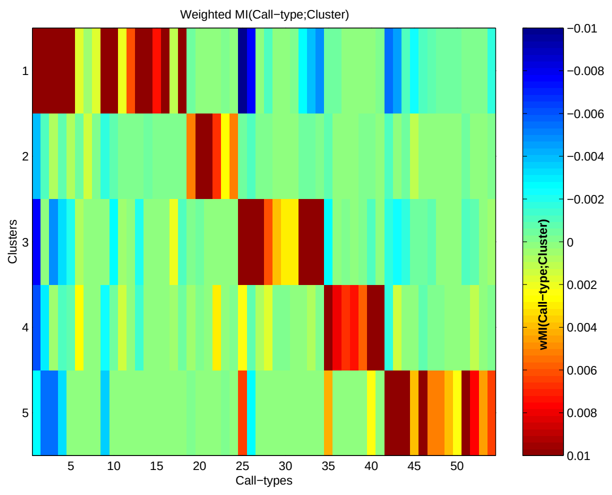

Figure 2: Weighted Mutual Information measures between and clusters on the HMIHY training corpus. The color of each rectangle shows the degree of correlation, determined by the weighted mutual information between the vector components and the cluster. As can be seen from the color spectrum on the right, dark red means high correlation, and dark blue means reverse correlation.

We chose to analyze the clusters according to 3 dimensions:

1. **Request** = the kind of request expressed by the user. We split all the request calltypes into two categories: the *easy* requests that correspond to the calltypes well detected by the SLU module and the *difficult* requests that contain all the calltypes that are often badly recognized.
2. **Understanding** = to try to characterize the *understanding* of a user by the system we use two features: the bigrams `request + yes` (**conf**) that can be indicators that the request is understood because the following concept expressed by the user is `yes`; the bigrams `request + request` (**repeat**) which indicate that the same request is repeated twice in a row, which can indicate that the system misunderstood it.
3. **Problems** = we grouped in this category the features that can be related to some problems the user have during the interaction. These features are: request for help

(**help**), two requests in a row for a representative (**rep**) and a calltype indicating that no meaningful information is extracted from the user's utterance (**null**).

	Request		**Underst.**		**Problems**		
C	**easy**	**dif**	**conf**	**repeat**	**help**	**rep**	**null**
1	+	−	+	=	−	−	−
2	−	+	−	+	+	−	+
3	+	−	=	=	=	+	=
4	−	+	+	+	+	+	+
5	−	+	+	+	=	=	−

Table 4: Interaction features attached to the dialog clusters obtained with the partitional method, at the dialog level, on the HMIHY corpus

Table 4 represents the 5 clusters from figure 2, ternary encoded according to these semantic dimensions. A $'+'$ means a higher value in the wMI scale, $'-'$ means a lower value in wMI scale and $'='$ stands for 0 in the wMI scale. According to the results presented in Tables 3 and 4, the following description can be given to the 5 clusters:

- Cluster C_1 : *easy* requests, expressed clearly (low perplexity, short utterances), understood correctly (`conf='+'`), and with no evidence of a problem.
- Cluster C_2 : *difficult* requests, with confusing formulation (high perplexity, long utterances), often misunderstood (`repeat='+'`), leading users to have problems (request for help, no info).
- Cluster C_3 : rather similar to C_1 except that people ask more often for a representative, maybe because the requests are not always recognized immediately (`conf='='`).
- Cluster C_4 : *difficult* requests, with confusing formulation (high perplexity, long utterances), leading users to have problems (request for help, no info), but unlike C_2 eventually leading to a success (high number of turns and `conf='+'`).
- Cluster C_5 : *difficult* requests, expressed clearly (low perplexity, rather short utterances), understood correctly (`conf='+'`) even after some repetition because these are difficult requests, and with no evidence of a problem.

Figure 3 gives portions of two example dialogs from clusters C_1 and C_2. As can be seen, in the first dialog, the user's responses are very clear, whereas in the second one they are very vague and complicated.

Example dialog from C_1:

- **System:** How may I help you?
- **User:** I want to pay a bill.
- *System Call-type: Make(Payment)*
- **System:** Do you want to arrange a bill payment?
- **User:** Yes.
- *System Call-type: Yes*

Example dialog from C_2:

- **System:** How may I help you?
- **User:** Yes somebody just called me from I don't know it's collect or something like that and it had to do with A T and T and when...
- *System Call-type: Explain(Bill)*
- **System:** Do you have a question about a specific charge on your bill?
- **User:** No
- *System Call-type: No*
- **System:** I'm sorry . How may I help you?
- **User:** It just said call you guys for help about trying to get through to a long distant number and it's not getting through when I called the long distant ...
- *System Call-type: Other*

Figure 3: Two example dialogs from our corpus.

6 Using clusters for Speech Mining

In the previous section we have interpreted the clusters obtained with the respech to the dialog contexts. Each cluster is identified by various features computed, characterizing the language and the interaction. These features can be seen as a set of tools allowing the designer of a Spoken Dialog System to monitor the behavior of the system. Indeed, once a clustering model has been build on a manually labeled training corpus, this model can be applied in a fully unsupervised way to non transcribed data and all the features presented in Figure 2 and Tables 3 and 4 can be extracted. Even if mistakes in the calltype detection occur, the general structure of the clusters is stable as shown on the HMIHY test corpus in Table 3 and in Figure 4 that plot the same parameters for the test corpus with automatic calltypes that Figure 2 has for the training corpus with manual labels.

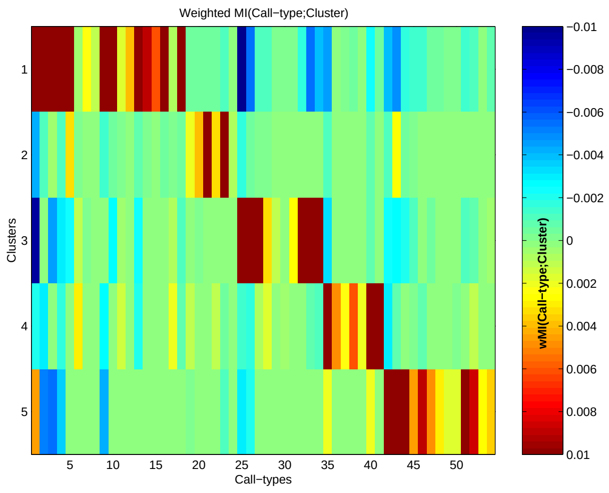

Figure 4: Weighted Mutual Information measures between calltype labels and clusters on the HMIHY test corpus with automatic calltype labels

7 Conclusions

We presented in this paper an application of data clustering methods to large Human-Computer spoken dialog corpora. Different ways for encoding dialogs into multidimensional structures (symbolic and numerical) and different clustering methods have been proposed. Preliminary results are given for cluster interpretation and dynamic model adaptation using the clusters obtained.

Acknowledgements

The authors would like to thank Driss Matrouf for providing the k-means clustering tools used in this study.

References

Frederic Bechet, Giuseppe Riccardi, and Dilek Hakkani-Tur. 2003. Multi-channel sentence classification for spoken dialogue language modeling. In *Proceedings of Eurospeech'03*, Geneve, Switzerland.

Yannick Esteve, Frederic Bechet, Alexis Nasr, and Renato De Mori. 2001. Stochastic finite state automata language model triggered by dialogue states. In *Proceedings of Eurospeech'01*, volume 1, pages 725–728, Aalborg, Danemark.

A.L. Gorin, G. Riccardi, and J.H. Wright. 1997. How May I Help You? *Speech Communication*, 23:113–127.

Helen Wright Hastie, Rashmi Prasad, and Marilyn A. Walker. 2002. What's the trouble: Automatically identifying problematic dialogs in darpa communicator dialog systems. In *Proceedings of the Association of Computational Linguistics Meeting*, Philadelphia, USA.

Candace Kamm, Marilyn A. Walker, and Diane Litman. 1999. Evaluating spoken language systems. In *Proceedings of American Voice Input/Output Society AVIOS*.

R. Kuhn and R. De Mori. 1995. The application of semantic classification trees to natural language understanding. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 17(449-460).