Discovering Customer Intent in Real-time for Streamlining Service Desk Conversations

Ullas Nambiar, Tanveer Faruquie, L Venkata Subramaniam, Sumit Negi
IBM Research - India, New Delhi, India
{ubnambiar, ftanveer, lvsubram, sumitneg}@in.ibm.com

Ganesh Ramakrishnan
Dept of Computer Science and Engg, IIT Mumbai, India
ganesh@cse.iitb.ac.in

ABSTRACT

Businesses require the contact center agents to meet pre-specified customer satisfaction levels while keeping the cost of operations low or meeting sales targets, objectives that end up being complementary and difficult to achieve in real-time. In this paper, we describe a speech enabled real-time conversation management system that tracks customer-agent conversations to detect user intent (e.g. gathering information, likely to buy, etc.) that can help agents to then decide the best sequence of actions for that call. We present an entropy based decision support system that parses a text stream generated in real-time during a audio conversation and identifies the first instance at which the intent becomes distinct enough for the agent to then take subsequent actions. We provide evaluation results displaying the efficiency and effectiveness of our system.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Search process, Selection Process

General Terms

Algorithms, Design, Performance

Keywords

Real-time, Contact Center Conversation, Customer Intent Recognition

1. INTRODUCTION

Customer contact centers (also known as call centers) were started as a low-cost alternative to maintaining physical centers for serving existing customers of a business. However, with a view to increasing profits and to make maximum utilization of the agents time, other activities like lead generation, taking orders, etc. were slowly added to the tasks performed. In this paper, we present, *AgentAid* - a real-time audio conversation management system that will streamline service desk conversations by automatically recognizing customer intent regarding a service and immediately prompt the agent with relevant information. AgentAid is designed to help service desk agents in solving the customer's information need while ensuring operational effectiveness. AgentAid achieves this by prompting the agent, at appropriate time periods during the call, with information which is useful for successfully closing the call. This information could be in the form of queries which the agent can ask the customer to gain a better understanding of the customer's problem or product/service recommendations for effective cross-sell/up-sell. Two key aspects of the AgentAid system are *a)* The information which is displayed to the agent is generated in real time, i.e., as the agent-customer conversation is in progress *b)* The system, based on the conversation context, automatically generates this information without requiring any input from the agent. The AgentAid system consists of two main modules namely the *Decision Module* and the *Query Builder*.

The Decision Module's aim is to identify the *customer's intent* in real-time while an audio conversation between a customer and contact center agent is taking place. To do this, we start by converting the on-going call audio to text using an Automatic Speech Recognizer(ASR). We then segregate the streaming text into distinct portions and assign a set of predefined categories that reflect the intent of the customer. This is depicted in Figure 1 where the Decision Module has assigned call segment d_0 to d_1 to the *'Car unavailability issue'* category and d_2 to d_3 to the *'High rates'* category. The critical task for Decision Module is to identify *intent points* i.e. earliest point in the streaming conversation where the customer's intent becomes evident with reasonable degree of confidence. Once this is done and the call segment assigned a category (intent category) relevant information is fetched from the backend database using the *Query Builder* and presented to the agent. For instance, the moment the Decision Module detects (step 1 in Figure 1) that a particular call segment is of type *'Car unavailability issue'* it invokes the Query Builder (step 2 in Figure 1) and passes the relevant call context (i.e. portion of the ASR that was categorized as *'Car unavailability issue'* t_{25} to t_{42}). The Query Builder uses this call context to automatically build a query for the relevant backend (step 3 in Figure 1) database (e.g. database containing current fleet inventory at the location). The result of the query is then displayed to the agent in realtime (step 4 in Figure 1). In this paper, we describe in detail the *Decision Module* component of the AgentAid system. The *Query Builder* module of our system is explained in our previous work [21].

The intent identification is done by using a classifier that not only identifies customer's intent but also discovers the *intent points* - points at which intent changes. Typically classifiers make decisions after the entire document. However, a call center conversation can be visualized as a continuously evolving document that is stream-

CIKM'11, October 24–28, 2011, Glasgow, Scotland, UK.

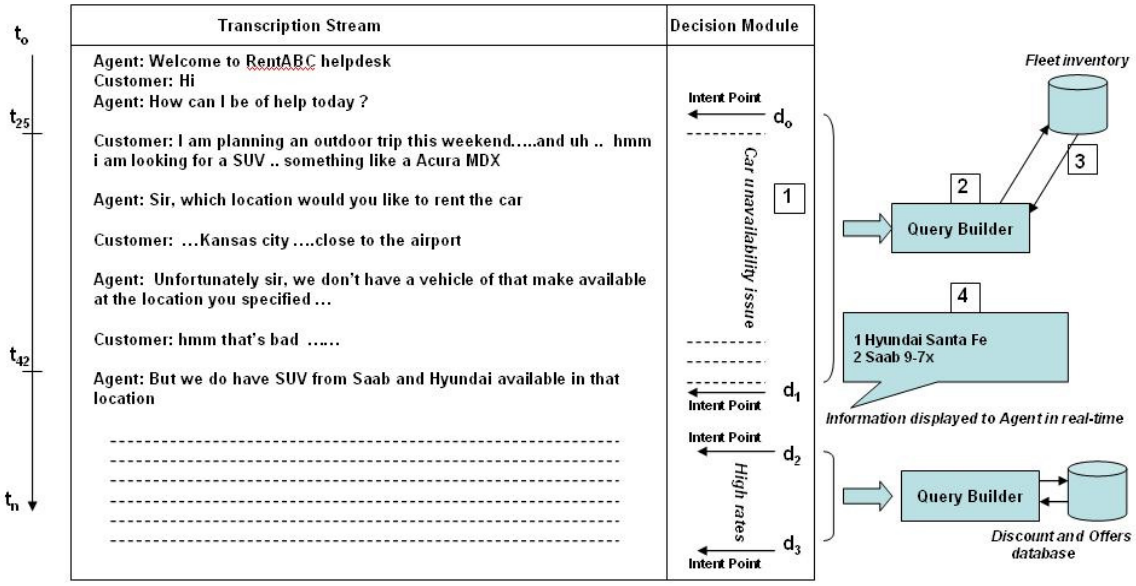

Figure 1: AgentAid system flow

ing into the system. Herein lies the *first challenge - The text stream is non-stationary i.e. distribution of features change with time and therefore the system must determine whether it has enough information to make the decision that an intent point has been reached and the call segment can be labeled with a pre-specified intent category.* Determining the customer's intent early in the call can help the agent take appropriate action (e.g. if the system prompts the agent that the customer's intent is to shop around for a good price it might be helpful to provide some kind of price comparison or offers upfront). Even if we can only narrow down to a broad intent category e.g. interested in *opening a checking account* and not to the particular type, *a student's checking*; the quicker mapping of information could help reduce the average time for the handling the call. The *second challenge* arises from the fact that the interactions between customer and agent is over a real-time audio call and therefore the customer expects satisfactory answers and suggestions in real-time. The *third challenge* is with respect to handling the high noise in audio to text translations done by most ASRs.

The remainder of the paper is organized as follows. In the next section, Section 2, we list related research efforts. In Section 3, we formulate the problem for efficiently finding customer intent in realtime conversation. We then show how we can assign intent categories and find intent points in Section 4. Section 5 presents detailed evaluation results over both synthetic and real-life data. We summarize our contributions in Section 6.

2. RELATED WORK

There exists prior work on building classifiers for call classification [16] and routing customer calls based on the customer response to an open ended system prompts such as, *"Welcome to xxx, How may I help you?"* [12]. In call classification and routing the complete call (document) is collected before a decision is made. In the case of online data streams where the individual class distribution does not remain stationary incremental learning is used [9]. The approach usually taken to capture non-stationarity is to take a time window or weigh the data depending on age or relevance. The size of the window tries to balance the adaptivity and generalization of the classifier [19]. The concept-adapting very fast decision tree learner [8] applies the very fast decision tree learner [4] to build the model incrementally using a sliding window of fixed size. Techniques to dynamically decide the window horizon to incorporate the relevance of the data stream have been proposed in [2]. Support Vector Machines (SVMs) have been used with a dynamic window size in [10]. The Incremental On Line Information Network [3] dynamically adjusts the window size and training frequency based on statistical measures.

In above methods the solution is to learn a classifier that forgets its past and learns the new distribution based on the changed concepts. This model is not suited in the call center conversation model where the document is evolving but a clear intent has not been established. Evolving the current state of classifier as more data streams in becomes necessary. To the best of our knowledge we are not aware of any prior work that looks at issues related to automatically finding intent points in real time conversations.

3. CLASSIFYING A CONVERSATION IN REAL-TIME

The primary challenge faced by AgentAid is to identify a particular conversational segment and classify it in realtime. Often the boundaries of these segments are fuzzy. For any real time call analysis, determining these boundaries quickly and effectively and making a classification decision is very important. The difficulty is compounded by the fact that segment boundaries are not available even at training time and often vary from call to call. In practice class labels for these particular segments are available only at the call level (i.e. entire document level) during training time. These labels are usually generated by contact center quality analysts as a post operational quality check exercise. The analysts after hearing portions of the call provide labeling at the call level only. In order to provide real time assistance to the agent we have to identify call segments and classify them as quickly as possible with high confidence. In the following section we describe an entropy based

approach that identifies call segment boundaries (intent points) at which customer's intent changes even when no segment boundary information is available at training time. We further propose to use the intent points as a feedback loop to retrain the classifier on partial call segments so as to improve the accuracy of the classifier.

The Problem: In our system conversations of contact center agents are passed in real time through an ASR system which outputs streaming text corresponding to a transcription of the conversation. At the end of every speaker utterance the ASR outputs the word uttered by the speaker [6]. Hence, at a given time t_i the transcript of the conversation till t_i is available. When such text is passed to a classifier at the end of every word the classifier potentially has a new decision. It can alter the estimated confidence about the customer's intent at t_{i-1} or it can detect a completely new intent. We assume that words in a conversation $w_1 \dots w_n$ are generated at time stamps $t_1 \dots t_n$. Our classifier makes the decision based on the conditional probability model $p(C_x/a_1, a_2, \dots, a_n)$ where $\hat{A} = a_1 \dots a_n$ are features derived from the incoming words $w_1 \dots w_n$ in the conversation and C_x takes on a value from the set of class labels $\mathcal{C}$. It is important to note that the features $\hat{A}$, necessary for classification, could all have been collected before the call reaches its end. In effect this means that even a partial call may be sufficient for the classifier to make a confident decision. Below we first define an *intent point* and then formally state the problem.

Intent Point: is the time instance t when the classifier has the least uncertainty $U(s_t)$ about the *intent category* to assign to the incoming conversation $s_t = w_1, w_2, \dots, w_t$. Any decision made before or after this point has non-decreasing uncertainty.

Problem Definition: In text classification for (streaming) conversational text the intent identification cannot wait till the complete document has been seen. Hence, in conversational text the classifier needs to take two decisions: (1) the classification decision to determine the customer's intent in the conversation (2) the point in time where to take the classification decision (determining the intent point).

4. ENTROPY BASED IDENTIFICATION OF INTENT POINTS

In information theory, the information entropy or Shannon entropy of a random variable X is the measure of uncertainty associated with X. The information entropy $H(X)$ of a discrete random variable X, that can take values in the set $\{x_1, x_2, \dots, x_n\}$ is defined to be

$$H(X) = E(I(X)) = \sum_{i=1}^{n} p(x_i) log_2 \left(\frac{1}{p(x_i)}\right) = -\sum_{i=1}^{n} p(x_i) log_2(p(x_i)) \quad (1)$$

where, $I(X)$ is the self information of X, which is itself a random variable and $p(x_i)$ is the probability mass function of X.

Let C_x be the class predicted for an instance x by a probabilistic classifier such as logistic regression [13]. C_x takes on a value from the set of class labels $\mathcal{C}$. The classifier associates a distribution with C_x. We use entropy of the class random variable C_x to measure the uncertainty $U(x)$ associated with the classification of x.

$$U(x) = -\sum_{C_x \ in \ \mathcal{C}} p(C_x) log_2(p(C_x)) \quad (2)$$

where, $p(C_x)$ is the posterior probability of class C_x for the instance x as predicted by the probabilistic classifier. A high value of $U(x)$ implies that the classifier is uncertain about the classification of instance x.

We make use of this uncertainty measure for determining the text segment that must be chosen from the test document to determine its class. To make computations feasible, we restrict the choice of text segments to multiples of units that are sentences. Further, we restrict the choice of segments to those that start at the beginning of the conversation. We refer to the segment comprising the first k sentences from a conversation S as s_k. For instance, given the conversation in Figure 1 we consider the following as candidate segments

$s_1 =$
<text name="Agent">Welcome to RentABC helpdesk</text>
$s_2 =$
<text name="Agent">Welcome to RentABC helpdesk< /text>
<text name="Customer">Hi</text>
$s_3 =$
<text name="Agent">Welcome to RentABC helpdesk</text>
<text name="Customer">Hi</text>
<text name="Agent">How can i be of help today ?</text>

Given a test conversation S consisting of N sentences, we consider two methods for determining its true class $C(S)$ on online streaming text

GlobalEntropy: This method is applicable when the entire conversation is made available before-hand, and the task is to determine segment that best determines the class label for the conversation. The best segment is determined by computing the uncertainty $U(S_i)$ of every segment s_i, $1 \leq i \leq N$ and finding the segment that has the lowest uncertainty.

$$C(S, GlobalEntropy) = C(s_k) \quad where \ s_k = \underset{s_i, \ 1 \leq i \leq N}{argmin} \ U(s_i) \quad (3)$$

where, $C(S, GlobalEntropy)$ is the class determined for the test instance S using the $GlobalEntropy$ method and $C(s_k)$ is the class determined for segment s_k using the probabilistic classifier that is being used. Further we define the quantity *Global Test Ratio* for a call as the ratio of $\frac{k}{N}$.

LocalEntropy: This method is applicable when the conversation is made available in a streaming manner and the task is to classify the conversation as quickly as possible. This method computes the uncertainty for the segment that has been seen so far and as soon as the uncertainty increases, it determines the class label based on the segment till the previous time instance.

$$C(S, LocalEntropy) = C(s_k) \quad where \ U(s_k) < U(s_{k+1}) \quad (4)$$

where, $C(S, LocalEntropy)$ is the class determined for the test instance S using the $LocalEntropy$ method. Further we define the quantity *Local Test Ratio* for a call as the ratio of $\frac{k}{N}$.

4.1 Incremental Classification

With streaming text the classifier incrementally updates its decisions. At every segment, the classifier has a decision, $p(C_x)$, and an associated Entropy value, $U(x)$. At some point into the stream the classifier has to decide on the "true" class. The relative entropy over the stream determines the point at which to make this decision. While we have presented our explanation based on a segment level

stream it also holds for a word stream. It is possible that the LocalEntropy based segment detection may get stuck in a local minima, which can be determined by checking whether the LocalEntropy again dips after n words. If it does not dip we declare the decision at minima point. If a new minima is encountered in the look ahead zone we update the minima point and again look ahead.

4.2 Designing Segment Classifiers

We have so far taken the approach of building a classifier on the complete call (i.e. document) and then using the entropy to make decisions on partial documents. Below we consider an alternate model where the classifier learnt using partial data itself with the hope that it will perform better. Basically, we argue that depending on the class labels different portions of the call may be important. To classify whether the agent opens the call in a proper way, a classifier built on just the first portion of the call is sufficient. However, for separating calls where credit card details are asked versus those where the customer agrees to pay on delivery, the middle segment of the call may be important. So the same call may need to be classified into these different classes, where different classifiers can come into play on different segments of the call.

4.2.1 Order of Appearance of Features

The text stream for a conversation is non-stationary therefore features for specific tasks are limited to portions of the call. So for example, in most calls the features relating to greeting will occur at the beginning of the call and features relating to types of payment may appear in another portion of the call. The classifier makes a decision based on the conditional model $p(C_x/a_1, \ldots, a_n)$.

4.2.2 Incremental Learning

Learning a classifier on streaming text would mean that the classifier is updated with each new feature that appears. Earlier we hypothesized that a classifier built using the whole call reaches a "true" decision after seeing just a fraction of the call. This implies that the features needed for the classifier have all been collected at this point in the call which could be a fraction of the complete call. Given a collection T consisting of m conversations, we use the following method for determining the segment s_k that should be used for training the classifier:

TrainingCutOffPoint: The cut off segment is determined by computing the *Global Test Ratio*, G, for each call in the collection T. Based on this ratio we select $s_k = G \times N$ from each call to train the classifier.

4.3 Overcoming Errors in Transcript

The ability of our solution to suggest entities of interest is therefore dependent on how well it can handle the noise in the input. Generally, ASR systems are known to have 30-40% error in transcription of service desk conversations. ASRs have three components - *An acoustic model* that works at the level of phonemic sounds (sub word), a *lexical model* that defines the vocabulary and models how each word is formed from the phonetic sounds comprising it and a *language model* that organizes the words to model the structure of the language. A speech signal is converted into a sequence of words based on a search that is constrained using these three components [17]. ASR systems typically use a lexicon of over 60,000 words and the models are trained using a few hundred hours of speech data. Also the language models are modified to take into account the domain specific features [18]. For example, in calls relating to an IT help desk, there will be a lot of computer specific words, and these words are given higher weights.

Since, AgentAid is only focussing on the noun phrases, only errors introduced in the noun phrase transcription would affect the system. Moreover, as explained above, a domain specific vocabulary or controlled vocabulary can be used to tune the language model of the ASR to perform better for words appearing in the given vocabulary. Since, we are only interested in noun-phrases appearing in the transactional database of a particular business, we can build a business specific controlled vocabulary and tune the ASR systems to reduce error. Such vocabularies can be modified in time to reflect popular terms based on call transcription logs.

An added advantage of using a controlled vocabulary is the ability to overcome spelling errors in the transcript. Spelling errors can be reduced by consulting the controlled vocabulary and checking for similarity between the transcribed term and any element of the vocabulary. In our implementation, we use a graded *Levenshtein distance* [1] measure to map terms in transcript to those in the vocabulary. The grading is dependent on the length of the word.

5. RESULTS AND DISCUSSION

In the following, we present results of evaluations done to assess the Decision Module's effectiveness in detecting intent points during real-time audio conversations. Below we first describe the dataset used for evaluation and then present results of various evaluations done using it.

5.1 Datasets

We obtained automatic transcriptions of contact center dialogs from a car rental process using an ASR.The speech recognition system was trained on 40 hours of data comprising of calls sampled at 8KHz. The resultant transcriptions had a word error rate of about 40%. We used 527 calls from the car rental process for our experiments. These calls were labeled according to two criteria - 1) based on the reason for *non-booking* by a customer and 2) based on reason for call or *call-opening*. Non-booking here relates to a call where the customer didn't rent a car. A total of 9 labels such as *shopping around, high rates etc.* were assigned by contact center quality analyst as reasons for non booking. We call this the *non-booking* data-set. The second criteria was a *binary labeling* based on whether call was made for booking or to find out the rates. We refer to this as the *call-opening* data-set.

We also used a publicly available data-set, the *20NewsGroups* data-set to simulate a real-time stream and test our methods in a non car rental setting. We chose two classes, *talk.politics.mideast* and *talk.politics.guns*, and picked all the postings within these classes that had approximately 30 to 50 sentences. This resulted in 285 files being used for 'mideast' class and 327 files for 'guns' class. We appended 65 to 85 sentences to these postings by randomly picking these sentences from the remaining 18 classes of 20newsgroup data. The sentences for the postings are treated as being generated in a streaming manner. Construction of dataset in this way results in the decision boundary of the 'mideast' and 'guns' classes to be between 0.26 (30/115) and 0.43 (50/115) of the normalized duration of the call.

5.2 Entropy Based Selection of Intent Point

We conducted classification experiments for the tasks of *call opening intent point identification* using the *call-opening* data-set and *non-booking intent point identification* using the *non-booking* data-set. The objective was to evaluate how the different methods proposed in the paper perform for the intent point detection and intent classification task i.e. categorizing the intent behind non-booking (high rates, unavailability etc) and for the call-opening in-

[1]The Levenshtein distance between two strings is given by the minimum number of operations needed to transform one string into the other, where an operation is an insertion, deletion, or substitution of a single character.

tent detection i.e. whether the customer's intent is to do a booking or find out rates. We trained a logistic regression classifier[2] on complete call transcripts of all the 527 calls for each task. During the testing phase instead of classifying the complete call we classified the call incrementally where each unit of increment was a sentence as described in Section 4. For each such incremental segment, $s_1 \ldots s_k$, we computed the entropy.

Table 1: Accuracies for classifiers trained on different fractions of the call for call-opening intent classification

Call Fraction	Baseline Accuracy	Global Accuracy	Local Accuracy	Best Accuracy
0.2	70.96	80.64	80.64	93.54
0.4	70.96	87.09	87.09	96.77
0.6	74.19	80.64	77.42	93.55
0.8	70.97	80.64	77.42	93.54
1.0	77.41	80.64	80.64	83.87

5.3 Intent Classifier Design

In Tables 1, 2 and 3 we show the various accuracies for classifiers that have been trained on fractions of the call for call-opening intent classification, non-booking intent classification and 20newsgroup classification respectively. *Baseline accuracy* represents the case when the testing is done on the complete call. *Global accuracy* represents the accuracy when the segment is selected based on GlobalEntropy. The average global test ratio represents the fraction of the call where the global minima of the entropy lies on an average for the whole set. *Local accuracy* is what is actually used in real-time classification of conversational text. It is the accuracy when the segment is selected based on LocalEntropy. The average local test ratio is the fraction of the call one needs to see before making a classification decision. The *best accuracy* is the maximum classification accuracy for different segment lengths. The best accuracy is arrived at by computing the classification scores for different segment lengths and selecting that segment which gives the highest accuracy over the whole collection. The best accuracy is the upper bound that can be obtained for the given task using a given classifier. In all the tables we see that the best accuracy peaks when the classifier is trained on a fraction of the call.

Table 2: Accuracies for classifiers trained on different fractions of the call for non-booking intent classification

Call Fraction	Baseline Accuracy	Global Accuracy	Local Accuracy	Best Accuracy
0.2	27.50	42.5	42.5	67.5
0.4	40.00	55.00	50.00	70.00
0.6	52.50	55.00	55.00	77.50
0.8	45.00	50.00	50.00	67.50
1.0	52.50	55.00	55.00	67.50

5.4 Comparison with Classification by Humans

We also compared the entropy based intent point selection with the human decision making process. Given the streaming text output from the conversation, a human listener would perform the classification after she has gathered enough evidence. For the call-opening intent classification task this decision point is at 0.25 and for non-booking intent classification task it is at 0.5. The human reaches close to 100% classification accuracy at this intent point. Also after this the classification accuracy remains at 100%. This is because once the human believes she has all the evidence required to reach a decision she does not change her mind. However, as shown in the results earlier our classifier achieves a high classification accuracy close to the intent point but after this intent point the accuracy actually goes down. In Table 4 we show the comparison between the intent point for classification done by a human and that determined by our method. Several observations can be made, namely - 1) A human classifier needs to see less of the call to make a final decision regarding the intent (i.e. the human intent point is earlier than that of our method). 2) For our method the classification accuracy is highest near the intent point and lesser before and after the intent point and 3) For a human classifier the classification accuracy reaches 100% at the intent point and remains at 100% even as more of the conversation is seen.

[2]We chose logistic regression as a representative since it performs very well in practice and has been shown to be closely related to SVM [20] and AdaBoost [5].

Table 3: Accuracies for classifiers trained on different fractions of 20 NewsGroups Data

Call Fraction	Baseline Accuracy	Global Accuracy	Local Accuracy	Best Accuracy
0.2	87.50	91.07	91.07	96.42
0.4	89.28	92.85	91.07	96.42
0.6	94.64	96.42	96.42	98.21
0.8	96.42	96.42	96.42	98.21
1.0	89.28	92.85	91.07	96.42

Table 4: Comparing Intent Point Determination by Humans and Entropy Technique

Task	Av. Global Test Ratio	Human Test Ratio
call-opening intent Classification	0.47	0.25
non-booking intent Classification	0.60	0.5

5.5 Higher Accuracy of ASR system with Controlled Vocabulary

The accuracy of a transcript is characterized by its *word error rate (WER)*. WER is measured as $\frac{S+D+I}{N} \times 100$ where N is the total number of words in the test set, and S, I and D are the total number of substitution, insertion, and deletion errors respectively. We also defined two other metrics, *Keyword Error Rate (KER)* and *Controlled Keyword Error Rate (CKER)*. Both KER and CKER are estimated similar to WER but differ over the set of words used for the measurement. KER is measured using all noun phrases appearing in the test set while CKER is measured using the noun phrases that appear in the controlled vocabulary. For our evaluation we randomly selected 501 call transcripts from the initial set of 527 call transcripts obtained from the car rental process. We looked at the Yahoo Auto Database [3] to determine a possible set of 300 words that could appear in a controlled vocabulary about cars.

We then measured, WER, KER and CKER using the automatically and manually generated transcripts. In addition, we also measured WER– where all stopwords were removed and CKER+ where we considered terms with a graded Levenshtein distance of 1 from the controlled vocabulary as similar to term in the vocabulary.

[3]http://autos.yahoo.com

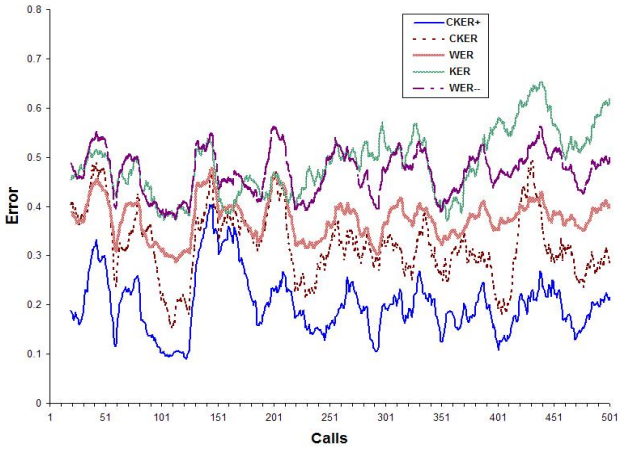

Figure 2: Comparison of WER, KER and CKER

Figure 2 compares the various error measurements for all the calls in our dataset. As expected the WER rate is around 30-35% for all the calls. However, CKER performs much better (25-30% error) than all the above and validates the assertion that ASR systems when tuned to domain specific words do improve the transcription accuracy.

6. CONCLUSION

In this paper we identify the need for identifying customer intent during a real-time call center conversation. We then presented details of the *AgentAid* conversation management system which prompts agents, at appropriate points during a real-time audio call, with information which is useful for successfully closing the call. Evaluation results demonstrate the ability of our system to efficiently and effectively identify customer intent, extract relevant information from backend database and prompt the agent about next steps. The results show our approach is able to overcome the errors introduced by Automatic Speech Recognition systems and is able to perform under strict response time constraints. To the best of our knowledge, AgentAid is the only conversation management system currently available for helping an agent during a real-time conversation. It can be implemented with minimal domain-specific setup and used by an agent with minimal training leading to reduced overall agent training costs.

7. REFERENCES

[1] S. Agarwal, S. Godbole, D. Punjani, S. Roy. 2007. How Much Noise Is Too Much: A Study in Automatic Text Classification. *Proceedings of IEEE International Conference on Data Mining*, Omaha, Nebraska, 3–12.

[2] C. Aggarwal, J. Han, J. Wang, P. S. Yu. 2004. On Demand classification of data streams. *Proceedings of the International Conferece on Knowledge Discovery and Data Mining*, Seattle, USA, 503–508.

[3] L. Cohen, G. Avrahami, M. Last, A. Kandel, O. Kipersztok. 2005. Incremental classification of nonstationary data streams. *ECML/PKDD-2005 International Workshop on Knowledge Discovery from Data Streams*, Portugal.

[4] P. Domingos, G. Hulten. 2000. Mining High-Speed Data Streams. *Proceedings of International Conference on Knowledge Discovery and Data Mining*, 71–80.

[5] Y. Freund, R. E. Schapire. 1996. Experiments in new boosting algorithm. *Prodeedings of the Thirteenth International Conference on Machine Learning*, 148–156.

[6] J. R. Glass, T. J. Hazen, I. L. Hetherington. 1999. Real-time telephone-based speech recognition in the Jupiter domain. *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, Phoenix, Arizona, 61–64.

[7] M. A. Hearst. 1994. Multi-paragraph segmentation of expository text. *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, Phoenix, Arizona, 61–64.

[8] G. Hulten, L. Spencer, P. Domingos. 2001. Mining Time-Changing Data Streams. *Proceedings of the Annual Meeting of the Association for Computer Linguistics (ACL)*, 9–16.

[9] I. Katakis, G. Tsoumakas, I. Vlahavas. 2006. Dynamic feature space and incremental feature selection for the classification of textual data streams. *ECML/PKDD-2006 International Workshop on Knowledge Discovery from Data Streams*, Berlin, Germany, 107–116.

[10] R. Klinkenberg, T. Joachims. 2000.Detecting concept drift with support vector machines. *Proceedings of International Conference on Machine Learning*, 487–494.

[11] K. Kummamuru, Deepak P., S. Roy, L. V. Subramaniam. 2008.Unsupervised Segmentation of Conversational Transcripts. *Proceedings of SIAM International Conference on Data Mining*, 834–845.

[12] H.-K. J. Kuo, C.-H. Lee. 2003. Discriminative training of natural language call routers. *IEEE Trans. on Speech and Audio Processing*, 11(1), 24–35.

[13] N. Landwehr, M. Hall, F. Eibe. 2003. Logistic Model Trees. *Proceedings of European Conference on Machine Learning*, Cavtat-Dubrovnik, Croatia, 241–252.

[14] Y. Park. 2007. Automatic call section segmentation for contact-center calls. *Proceedings of International Conference on Knowledge Discovery and Data Mining*, Lisbon, Portugal, 117–126.

[15] H. Takeuchi, L. V. Subramaniam, T. Nasukawa, S. Roy. 2007. Automatic Identification of Important Segments and Expressions for Mining of Business-Oriented Conversations at Contact Centers. *Proceedings of Empirical Methods on Natural Language Processing*, Prague, Czech Republic.

[16] M. Tang, B. Pellom, K. Hacioglu. 2003. Calltype classification and unsupervised training for the call center domain. *Proceedings of the Automatic Speech Recog. and Understanding Workshop*, St. Thomas, U S Virgin Islands, 204–208.

[17] M. Padmanabhan, G. Saon, J. Huang, B. Kingsbury, and L. Mangu. Automatic Speech Recognition Performance on a Voicemail Transcription Task. *IEEE Trans. on Speech and Audio Processing*, 10(7):433-442, 2002.

[18] H. Chevalier, C. Ingold, C. Kunz, C. Moore, C. Roven, J. Yamron, B. Baker, P. Bamberg, S. Bridle, T. Bruce, and A. Weader. Large-vocabulary speech recognition in specialized domains. *ICASSP-95*, May 9-12, 1995, Detroit, MI, USA.

[19] G. Widmer, M. Kubat. 1996. Learning in the presence of concept drift and hidden contexts. *Machine Learning*, 23(1), 69–101.

[20] V. N. Vapnik. 1995. *The nature of statistical learning theory*. Springer-Verlag, New York.

[21] U. Nambiar, H. Gupta, R. Balakrishnan and M. Mohania. 2008. Helping Satisfy Multiple Objectives during a Service Desk Conversation. *SIGMOD*, June 9-12, 2008, Vancouver, BC, Canada.

[22] T. Mitchell. 1997 Machine Learning. *McGraw Hill*.