

# An Active Approach to Spoken Language Processing

DILEK HAKKANI-TÜR, GIUSEPPE RICCARDI, and GOKHAN TUR  
AT&T Labs—Research

---

State of the art data-driven speech and language processing systems require a large amount of human intervention ranging from data annotation to system prototyping. In the traditional *supervised passive approach*, the system is trained on a given number of annotated data samples and evaluated using a separate test set. Then more data is collected arbitrarily, annotated, and the whole cycle is repeated. In this article, we propose the *active approach* where the system itself selects its own training data, evaluates itself and re-trains when necessary. We first employ *active learning* which aims to automatically select the examples that are likely to be the most informative for a given task. We use active learning for both selecting the examples to label and the examples to re-label in order to correct labeling errors. Furthermore, the system automatically evaluates itself using *active evaluation* to keep track of the unexpected events and decides on-demand to label more examples. The active approach enables dynamic adaptation of spoken language processing systems to unseen or unexpected events for nonstationary input while reducing the manual annotation effort significantly. We have evaluated the active approach with the AT&T spoken dialog system used for customer care applications. In this article, we present our results for both automatic speech recognition and spoken language understanding.

Categories and Subject Descriptors: I.2.7 [Artificial Intelligence]: Natural Language Processing—*Speech recognition and synthesis*; I.5.1 [Pattern Recognition]: Models—*Statistical*

General Terms: Algorithms, Languages, Performance

Additional Key Words and Phrases: Passive learning, active learning, adaptive learning, unsupervised learning, active evaluation, spoken language understanding, automatic speech recognition, spoken dialog systems, speech and language processing

---

## 1. INTRODUCTION

State of the art data-driven speech and language processing systems are trained and tested using large amounts of randomly selected annotated data.

---

The authors are listed in alphabetical order.

Authors' present addresses: D. Hakkani-Tür, ICSI, University of California—Berkeley, Berkeley, CA 94704; email: dilek@icsi.berkeley.edu; G. Riccardi, Department of Information and Communication Technology, University of Trento, 38050 Povo di Trento, Italy; email: riccardi@dit.unitn.it; G. Tur, SRI International, Speech Technology and Research Lab, Menlo Park, CA 94025; email: gokhan@speech.sri.com.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or direct commercial advantage and that copies show this notice on the first page or initial screen of a display along with the full citation. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, to redistribute to lists, or to use any component of this work in other works requires prior specific permission and/or a fee. Permissions may be requested from Publications Dept., ACM, Inc., 2 Penn Plaza, Suite 701, New York, NY 10121-0701 USA, fax +1 (212) 869-0481, or [permissions@acm.org](mailto:permissions@acm.org).  
© 2006 ACM 1550-4875/06/1000-0001 \$5.00

In such a *passive approach*, the input channel is assumed to be stationary. Typical tasks include the part of speech tagging or syntactic parsing using the Penn Treebank [Marcus et al. 1993], text categorization using the Reuters documents [Reuters 2004], natural language understanding using the ATIS data [Price 1990], or speech recognition evaluations using the Switchboard data [Godfrey et al. 1990]. Such benchmark tests are sometimes repeated annually with newly annotated data.

Although keeping the training and test sets fixed may be a good idea to compare different approaches, for many data-driven real world applications, the passive approach has some drawbacks. First, it makes inefficient use of data. For example, same or very similar examples are generally labeled multiple times in the passive approach. Since annotation is an expensive, laborious and time consuming process,<sup>1</sup> the necessity to manually annotate similar training examples should be minimized. Second, it restricts the system behavior to adapt dynamically to nonstationary input channels. This is especially the case when the system is actually deployed, and needs to be updated in a dynamic manner.

In the literature, there are both model adaptation algorithms to be employed by nonstationary systems [Digalakis et al. 1995; Leggetter and Woodland 1995a; Federico 1996; Riccardi and Gorin 2000]. Even in the adaptation approach it is not determined *how* to track the time-varying statistics and *when* to update the parameters.

In this article, we propose an *active approach* where the system selects its own training data, re-trains, and evaluates itself when necessary. Human intervention is limited to the labeling of only the selectively sampled data. To this end, we employ *active learning* which aims to automatically select the examples that are likely to be the most informative for a given task. We use active learning for both selecting the examples to label and determining the misannotated ones. By this, we aim at decreasing the number of training examples to be labeled and checked for errors and inconsistencies. Active learning has the distinct advantage of efficiently exploiting annotated data and thus reduces human effort. Moreover, modeling under the active learning paradigm has the intrinsic capability to adapt to non-stationary events by means of a feedback mechanism in the training algorithm. The examples that are not selected by active learning are exploited using *semi-supervised learning* methods. Furthermore, the system automatically checks the performance using *active evaluation* to keep track of the unexpected events and decides on when to label data and re-train the models.

In our previous work, we have proposed methods for the application of active and unsupervised learning to both automatic speech recognition (ASR) [Hakkani-Tür et al. 2002, 2004; Riccardi and Hakkani-Tür 2003, 2005] and spoken language understanding (SLU) [Tur et al. 2003a, 2003b, 2005; Tur and Hakkani-Tür 2003]. In this article, we propose combining all these components along with a tracking and evaluation mechanism, in a single, complete, and task-independent framework, namely the *active approach*.

---

<sup>1</sup>In this article, we use the words “annotation” and “labeling” interchangeably.

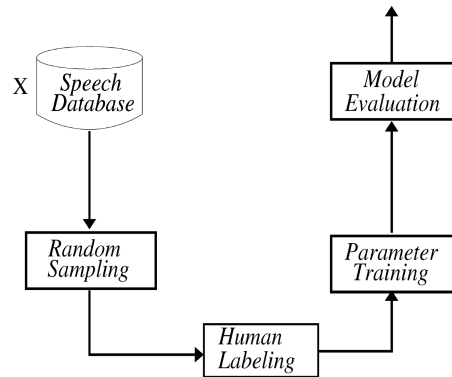


Fig. 1. Supervised passive learning.

The organization of this article is as follows: In the next section, we review the passive approach, then in Section 3, we explain the proposed active approach. We describe the application of active approach along with corresponding experiments to ASR in Section 4 and to SLU in Section 5.

## 2. THE PASSIVE APPROACH

In the standard maximum likelihood approach to train the statistical models, model parameters are estimated so that the training data probability is maximized [Jelinek 1997]:

$$\hat{\theta} = \operatorname{argmax}_{\theta} P(X|\theta),$$

where  $X = \{x_1, x_2, \dots, x_n\}$  is the available training set,  $x_i$  ( $i = 1, \dots, n, n = |X|$ ) is the  $i$ th labeled example, and  $P(X|\theta)$  is the probability of observing the set of training examples in  $X$ , for a given model parameter set  $\theta$ . Data is usually randomly collected, transcribed, and split into a training and a test set. In maximum likelihood formulation  $X$  is fixed and the identical-and-independent-distribution (i.i.d) assumption is made. The training examples,  $\mathbf{x}_i \in X$ , are drawn at random from the set  $\mathcal{X}$ , which has been selected a-priori and fixed in time.<sup>2</sup>

In Figure 1, we give the architecture of the learning process in the case of passive supervised learning. In this learning scheme, there is no relation between the expected error rate and the set of training examples  $X$ , for nonstationary distribution. In other words, if a new set of training examples  $X'$  is provided, it is not possible to predict if this set would decrease or increase the error rate estimated on  $X$ . Since training and test sets are sampled randomly from  $\mathcal{X}$ , *all* examples are considered equally *informative* for the purpose of learning. The test set matches the training set, hence the motto “There is no data like more data” holds for almost any statistical learning algorithm. No example is disregarded or generated automatically. In a passive supervised approach, no

<sup>2</sup>Similar arguments can be applied to training algorithms directly minimizing the error rate, such as in discriminative classifiers (e.g., Boosting [Schapire and Singer 2000]). These algorithms are not designed to adaptively select the training examples.

unlabeled data is exploited. The data is assumed to be stationary, and all the examples are assumed to be obtained and labeled at the same time. Typically, evaluations are made for comparing different models obtained with different methods but trained on the same given training set.

### 3. THE ACTIVE APPROACH

Passive approach delegates the burden of estimating *good* models to the estimation techniques. In contrast, active approach emphasizes the role of the input selection for the purpose of decreasing the expected error rate over time, hence adapting to the changes in the incoming data. The training and test sets are not fixed a-priori. We also do not assume that they are stationary. This means, the training set is not given, instead formed dynamically from a stream of incoming data using a buffering mechanism. The distribution of this data is also allowed to vary in time. Training set is not sampled randomly from  $\mathcal{X}$ . *Active learning* ensures the selection of the most informative examples. Test set is randomly selected to track the dynamic system performance, and to compare re-trained models with the already trained model. The models are updated regularly and automatically. *Active evaluation* keeps track of the performance and controls active and semi-supervised learning. There is no human involved in this process, except for labeling the selected data.

In the following sections, we present each of the components of the active approach:

- Active Learning for Labeling* decides on the examples to be labeled and sends them to human labelers.
- Active Learning for Labeling Error Correction* ensures the consistency and correctness of the manual labels by selecting the potentially problematic examples to re-label.
- Semi-Supervised Learning* exploits the unlabeled examples to get a better performance.
- Active Evaluation* evaluates the model automatically. The evaluation results are not only used for deciding the amount of data to label manually, and when to replace the existing model with a new model, but also for detecting the anomalies in the data, model, or the task itself.

Figure 2 presents a very high level description of the active approach. First a small amount of randomly selected data is labeled manually. This is used to train the core model to bootstrap. Then the system can begin buffering data from the incoming stream. A portion is then set aside for active evaluation. We then employ active learning for labeling using the remaining data to determine examples to add to the training data. Active learning for labeling error correction is used to select the problematic examples which are then checked manually. A new model is trained using this extended training set and the unlabeled examples using semi-supervised learning. Then active evaluation decides on deployment of the newly built model using the evaluation results on the test set.

One can tailor the proposed active approach for specific needs. For example, it is possible to exploit the test sets from previous iterations during training.

- (1) Train a model using small amount of randomly selected training data,  $S_{all\_labeled}$
- (2) Initiate processing the data stream using this model
- (3) While (labeler/data available) do
  - (a) Buffer data from the stream forming  $S_{all}$
  - (b) Set aside a randomly selected portion as the test set,  $S_{test}$ 

$$S_{all} = S_{all} - S_{test}$$
  - (c) Employ active learning for labeling on  $S_{all}$  to selectively sample extra training data,  $S_{selected}$
  - (d) Manually re-label and update  $S_{selected}$  which is selected by active learning for labeling error correction
  - (e)  $S_{all\_labeled} = S_{selected} \cup S_{all\_labeled}$
  - (f)  $S_{unlabeled} = S_{all} - S_{selected}$
  - (g)  $S_{all\_unlabeled} = S_{unlabeled} \cup S_{all\_unlabeled}$
  - (h) Train a new model using  $S_{all\_labeled}$  and  $S_{all\_unlabeled}$  via supervised and semi-supervised learning
  - (i) Employ active evaluation using  $S_{test}$

Fig. 2. The high-level algorithm of the active approach.

This approach is general enough to handle any task which includes model training and provides an incoming continuous data stream. In that sense, spoken language understanding or automatic speech recognition are just two example tasks for which the active approach can be used. As seen, in an active approach, active learning for labeling and active evaluation are the essential components, whereas active learning for labeling error correction and semi-supervised learning are employed for a shorter cycle time and to get better models with high-quality labeling.

### 3.1 Background

**3.1.1 Active Learning.** Previous work in active learning has concentrated on two approaches: committee-based methods and certainty-based methods. In the *committee-based methods*, a distinct set of learners is created using a small set of annotated examples. The unannotated instances whose outputs differ the most when presented to different learners are given to the labelers for annotation. Seung et al. [1992] have introduced this approach calling it *query by committee*. Freund et al. [1997] have provided an extensive analysis of this algorithm for neural networks. Liere and Tadepalli [1997] have employed committee-based active learning for text categorization and have obtained 2–30 fold reduction in the need of human labeled data depending on the size of labeled data. Argamon-Engelson and Dagan [1999] have formalized this algorithm for probabilistic classifiers introducing the metric *vote entropy* to compute the disagreement of the committee members. They have demonstrated its use for the task of part-of-speech tagging.

In the *certainty-based methods*, an initial system is trained using a small set of annotated examples. Then, the system examines and labels the unannotated examples and determines the certainties of its predictions on them. The examples with the lowest certainties are then presented to the labelers for annotation.

Cohn et al. [1994] have introduced certainty-based active learning for classification based on an earlier work on artificial membership queries [Angluin 1988]. Lewis and Catlett [1994] have used selective sampling for text categorization and have obtained 10-fold reduction in the need of labeled data using decision trees. Active learning has previously been applied to support-vector machines [Schohn and Cohn 2000; Tong and Koller 2001]. For language processing, certainty-based methods have been used for natural language parsing [Thompson et al. 1999; Tang et al. 2002; Baldrige and Osborne 2003] and information extraction [Thompson et al. 1999] and word segmentation [Sassano 2002]. One drawback with certainty-based methods is that it is hard to distinguish an informative example from an outlier. On the other hand, in order to train multiple classifiers required by committee-based active learning, one may divide the training data or feature set into multiple parts, and this may result in many weak classifiers instead of a single strong one.

In our previous work, we have presented certainty-based active learning methods for ASR [Hakkani-Tür et al. 2002] and SLU [Tur et al. 2003b]. In those studies, the goal was reducing the amount of labeled data needed to obtain better models in shorter time frames.

**3.1.2 Semi-Supervised Learning.** Compared to active learning, semi-supervised learning is a much more studied subject in machine learning and speech recognition literature. In speech recognition, it has been studied under the subject of *unsupervised adaptation*. Two well-known methods are *maximum a-posteriori* (MAP) and *maximum likelihood logistic regression* (MLLR) adaptation [Huang et al. 2001; Leggetter and Woodland 1995b; Bacchiani and Roark 2003]. Iyer et al. [2002] have proposed using ASR output of utterances, instead of their transcriptions, for training the ASR language models, without loss in performance. In our previous work [Hakkani-Tür et al. 2004], for the task of human-machine spoken dialog processing, we have evaluated multiple semi-supervised learning strategies.

In the machine learning literature, many semi-supervised learning algorithms have been proposed: Blum and Mitchell [1998] have proposed an approach, called Co-Training. For using Co-Training, the features in the problem domain should naturally divide into two sets. Then, the examples which are classified with high confidence scores with one view can be added to the training data of the other views. For example, for web page classification, one view can be the text in them and another view can be the text in the hyperlinks pointing to those web pages. For the same task, Nigam et al. [2000] have used an algorithm for learning from labeled and unlabeled documents based on the combination of the Expectation Maximization (EM) algorithm and a Naive Bayes classifier. Similar to the MAP adaptation in ASR, they first estimate the parameters of the Naive Bayes classifier,  $\hat{\Theta}$ , from a small amount of labeled examples. Then, this model is used to get the posterior probabilities  $P_{\hat{\Theta}}(c_j|d_i)$  for each class  $c_j$  given the example  $d_i$ . Next a new model is trained using both the manually and machine labeled data. These steps are iterated until the model performance converges. Nigam and Ghani [2000] have then combined Co-Training and EM, coming up with the Co-EM algorithm, which is

the probabilistic version of Co-Training. Ghani [2002] has later combined the Co-EM algorithm with Error-Correcting Output Coding (ECOC), to exploit the unlabeled data, in addition to the labeled data.

In our previous work, we have presented semi-supervised learning approaches for improving call classification accuracy using unlabeled data with a MAP-like adaptation technique [Tur and Hakkani-Tür 2003].

**3.1.3 Combining Active and Semi-Supervised Learning.** Combining active and semi-supervised learning has been first introduced by McCallum and Nigam [1998]. They combined the committee-based active learning algorithm with an EM-style semi-supervised learning to assign class labels of those examples that remain unlabeled. For the task of text categorization, using a Bayesian classifier, they have employed EM for each of the committee members. Muslea et al. [2002] have extended this idea to use Co-EM as the semi-supervised learning algorithm. They call this new algorithm Co-EMT, and have shown that exploiting multiple views for both active and semi-supervised learning is very effective. Their algorithm makes more sense since once the committee members are formed for active learning, it is straightforward to employ Co-Training or its variant Co-EM.

In our previous work, we have used a combination of active and unsupervised learning for ASR statistical language modeling [Riccardi and Hakkani-Tür 2003] and SLU [Tur et al. 2005].

**3.1.4 Labeling Error Detection.** Most of the work related to labeling error detection deals with the problem as a postprocessing step [Abney et al. 1999; Eskin 2000; Murata et al. 2002; van Halteren 2000]. These studies aim to detect a relatively small number of errors in the data after the labeling process. Other studies are either related to making the classifier robust to noisy data [Kearns 1993, among others] or they briefly mention that they have manually checked the data and corrected labeling errors [Hendrickx et al. 2002, among others].

## 3.2 Active Learning for Labeling

The purpose of active learning for labeling is to reduce the number of training examples to be labeled by selectively sampling a subset of the unlabeled data. This is done by inspecting the unlabeled examples, and selecting the most *informative* ones with respect to a given cost function for a human to label [Cohn et al. 1994]. In other words, the goal of the active learning algorithm is to select the examples which will bring the largest improvement on the system performance, hence reduce the amount of human labeling effort. Furthermore active learning provides robustness to changes in the incoming data distribution by adapting to them.

In Figure 3, we give the general scheme of active learning. The input set,  $X$  of Figure 1, is not shown, because it is assumed that there is a dynamic feed of unlabeled data, relaxing the stationarity assumption of the passive approach. Instead, we buffer the examples,  $X(t)$ , to form the pool that selective sampling will use.

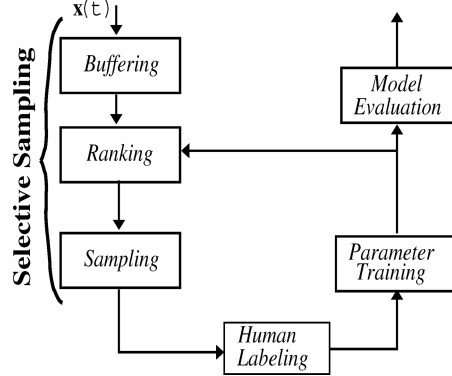


Fig. 3. Simplified structure of the Active Approach.

- (1) Given some amount of training data  $S_{all\_labeled}$ , and a larger amount of buffered unlabeled data in the pool  $S_{all} = \{s_1, \dots, s_n\}$
- (2) While (labelers/examples are available) do
  - 2.1 Train a model using the current training data  $S_{all\_labeled}$
  - 2.2 Compute the error estimations,  $\epsilon(s_i)$ ,  $i = 1, \dots, n$ , for the examples in  $S_{all}$  using this model,
  - 2.3 Manually label the set  $S_{selected} = \{s_i : \epsilon(s_i) > th\}$
  - 2.4  $S_{all\_labeled} = S_{all\_labeled} \cup S_{selected}$
  - 2.5 Get new unlabeled data into the pool,  $S_{all}$

Fig. 4. The certainty-based active learning algorithm.

Figure 4 summarizes the certainty-based active learning algorithm we propose. This method is independent of the task modeled. The error estimation is task dependent, and we propose methods for ASR and SLU in Sections 4.1 and 5.1 consecutively. The threshold,  $th$ , is mainly determined by the capacity of the manual labeling effort or by the performance of the current model. In accordance with the data properties in an active approach, this algorithm assumes a constant stream of incoming data traffic instead of a fixed given data set. At each iteration a new pool,  $S_{all}$ , is provided to the algorithm, which is the buffered data set. Then the aim of active learning is to come up with a smaller subset of examples,  $S_{selected}$ , collected from the field for human labeling.

More formally, the selection of training data  $\hat{X} \subset X$  is driven by a reward function,  $R(\bar{X})$ , which takes into account the total error rate  $\mathcal{E}(\bar{X})$  and the cost function of data labeling,  $f(|\bar{X}|)$ .  $\hat{X}$  is selected so that the following objective function is maximized:

$$R(\bar{X}) = [\mathcal{E}(\bar{X}) - f(|\bar{X}|)]$$

$$\hat{X} = \operatorname{argmax}_{X \in X} \{R(\bar{X}), 0\}$$

In the case that the maximum value is 0 (i.e.,  $R(\bar{X}) \leq 0$ ),  $\hat{X}$  is the empty set, and the reward is 0.  $\mathcal{E}(\bar{X})$  is the total error rate of the examples in  $X$ , and accounts for the benefit (e.g., error minimization) of having data labeled. To estimate

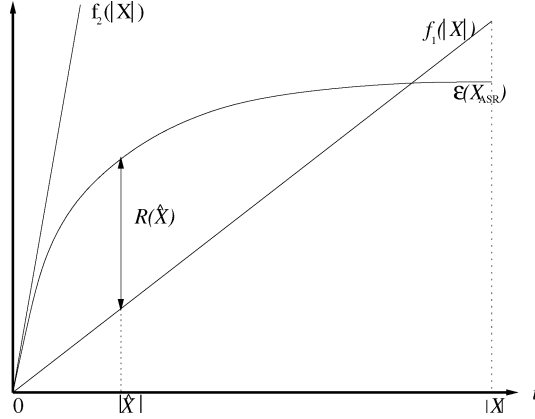


Fig. 5. An illustration of the estimated error and cost functions.

$\mathcal{E}(\bar{X})$  we use the estimator  $\epsilon(x_i)$  of the error rate for each sample  $x_i \in \bar{X}$ :

$$\mathcal{E}(\bar{X}) \approx \sum_{x_i \in \bar{X}} \epsilon(x_i)$$

where  $0 \leq \epsilon(x_i) \leq 1$ . We assume that  $f$  is positive ( $f(|\bar{X}|) \geq 0, \forall \bar{X}$ ) and monotonically increasing with the number of examples to be labeled. For practical purposes, we can assume  $f(|X|)$  to be linear in the number of examples:

$$f(|X|) = g(X) \times |X|$$

where  $g(X) > 0$  reflects the cost-benefit ratio of data labeling. If  $0 < g(X) < \epsilon(x_i)$  the benefit of labeling sample  $x_i$  is larger than its cost, if  $g(X) = \epsilon(x_i)$  cost and benefit are equal and for  $g(X) > \epsilon(x_i)$  the benefit of labeling  $x_i$  is smaller than its cost. In general, the factor  $g(X)$  depends on  $X$ . Figure 5 illustrates two cost functions,  $f_1(|X|)$  and  $f_2(|X|)$ , and  $\mathcal{E}(X)$ , where the set of examples in  $X$  are sorted according to decreasing estimated error. We assume two different constants as  $g(X)$  in  $f_1(|X|)$  and  $f_2(|X|)$ .

Note that the statistics that are drawn are biased and the prior distributions are not preserved. In the case of maximum likelihood estimation,

$$\hat{\theta} = \operatorname{argmax}_{\theta} P(\hat{X}|\theta)$$

the parameters  $\hat{\theta}$  will be biased by the active learning selection in order to minimize the error rate.

### 3.3 Active Learning for Labeling Error Correction

Labeling is an error-prone process due to various reasons, such as human factors or imperfect description of classes. Thus, usually more than one pass of labeling is required in order to check and fix the labeling errors and inconsistencies.

As another application of active learning, we aim to minimize the number of examples to be checked again by automatically selecting the ones that are likely to be erroneous or inconsistent with the previously labeled examples.

- (1) Given some amount of training data,  $S_{all\_labeled}$ .
- (2) Train a model using  $S_{all\_labeled}$  and initiate the incoming data stream
- (3) While (labeler/data available) do
  - (a) Buffer data from the stream forming  $S_{all}$ .
  - (b) Manually label a portion of  $S_{all}$ , call it  $S_{labeled}$ .
  - (c) Compute the disagreements  $KL(M_i||s_j)$  with the classifier,  $M_i$ , and the example  $s_j \in S_{labeled}$ .
  - (d) Manually re-label the set  $S_{selected} = \{s_i : KL(M_i||s_i) > th\}$  and update  $S_{labeled}$ .
  - (e)  $S_{all\_labeled} = S_{labeled} \cup S_{all\_labeled}$ .
  - (f) Train a new model using  $S_{all\_labeled}$ .

Fig. 6. The algorithm of the active learning for labeling error correction.

Figure 6 presents our approach. Inspired by certainty-based active learning, we leave out the ones that the classifier agrees with the labeler’s decision with high confidence but select the examples that the classifier is confident about but disagree with the first labeler’s decision. The disagreement between the classifier output and manual label for one example can be computed using multiple ways. One is using the Kullback–Leibler (KL) divergence (or relative entropy) for classification tasks:

$$KL(P \parallel Q) = \sum_{i \in L} p_i \times \log \left( \frac{p_i}{q_i} \right),$$

where  $L$  is the set of all classes,  $q_i$  is the posterior probability of the  $i$ th class obtained from the classifier. Since the first pass labels do not provide a probability distribution, we handled it as follows: we set  $p_i$  to 1 if that class is labeled and 0 otherwise. This method requires that the classifier returns a confidence between 0 and 1 for each of the labels,  $i \in L$ , where  $L$  is the set of all calltypes, for a given utterance,  $U$ . Indeed this is the case for most statistical classifiers.

Although this method may be used for postprocessing the already labeled data, we consider it as part of the labeling process. In our previous work we have also proposed an alternative method for checking the labeling errors, which does not require a given model [Tur et al. 2003b]. In that method, the motivation is that the examples in the training data which are not classified correctly with a model trained with the very same data are more probably the labeling errors.

### 3.4 Semi-Supervised Learning

The examples that are not selected by active learning can be exploited using semi-supervised learning methods. This combination has various distinct advantages:

- It is possible to get a better performance by exploiting unlabeled data as described in Section 3.1.
- Active learning alone is not sufficient to track the new distributions in the incoming data stream. For the case of classification, active learning may ignore the easily classified examples although they might indicate a very different class distribution.
- Semi-supervised learning may help restoring the distributions perturbed by active learning. For the case of classification, one can assume that the easily

classified examples, which are not selected by active learning, belong to more frequent classes.

For ASR, after applying active learning to intelligently select and then transcribe a small fraction of the data that is most informative, we use the rest of the data that is not yet transcribed in semi-supervised learning. More formally, we compute the initial language model parameters,  $\theta$ , using selectively sampled and transcribed data,  $W_t$ . Then using  $\theta$ , we recognize the unselected, hence untranscribed data to get  $\hat{W}_u$ :

$$\hat{W}_u = \underset{W}{\operatorname{argmax}} P_\theta(W) \times P(X_u|W)$$

where  $P(X_u|W)$  is the acoustic observation probability computed using a given acoustic model. Then we compute the updated language model parameters,  $\hat{\theta}$ , by combining  $W_t$  and  $\hat{W}_u$ .

The  $n$ -gram counts  $C_{AL-UL}(w_1^n)$  from human transcribed (via Active Learning) and ASR transcribed speech utterances are computed in the following way, for each word sequence  $w_1^n$ :

$$C_{AL-UL}(w_1^n) = C_{AL}(w_1^n) + C_{UL}(w_1^n) \quad (1)$$

where  $C_{AL}(w_1^n)$  is the  $n$ -gram count computed from the manually transcribed data,  $W_t$ , and  $C_{UL}(w_1^n)$  is the  $n$ -gram count computed from the ASR output  $\hat{W}_u$ . To compute  $C_{UL}(w_1^n)$ , for each word  $w_i$  in  $\hat{W}_u$  we estimate the probability of being correctly decoded  $c_i (= 1 - e_i$ , where  $e_i$  is the error probability), that is its confidence score. The bidimensional channel is then represented as a sequence of  $n$ -tuples of symbol pairs  $(w_1^n, c_1^n) = (w_1, c_1)(w_2, c_2), \dots, (w_n, c_n)$ . The  $n$ -gram counts in presence of noise can be computed by marginalizing the joint channel counts:

$$C_{UL}(w_1^n) = \sum_{x \in \hat{W}_u} c_x \delta_{w_1^n}(x), \quad (2)$$

where  $c_x$  is the confidence score for the  $n$ -tuple  $x$  in the noisy spoken utterance transcriptions  $\hat{W}_u$  and  $\delta_{w_1^n}(x)$  is the indicator function for the  $n$ -tuple  $w_1^n$ . The confidence score of the  $n$ -tuple  $w_1^n$  can be computed by geometric or arithmetic means or max and min over the  $n$ -tuple of word confidence scores  $c_1^n$ . In this article, we take the simplest approach and compute  $c_{w_1^n} = 1$ . Other ways of computing  $c_1^n$  are tested in Riccardi and Hakkani-Tür [2003] and Gretter and Riccardi [2001]. The acoustic models can be trained by using all the data in a similar fashion [Zavaliagkos and Colthurst 1998].

In our previous work, we have presented semi-supervised learning approaches for exploiting unlabeled data for call classification [Tur and Hakkani-Tür 2003]. We proposed a novel approach for Boosting. Boosting aims to combine “weak” base classifiers to come up with a “strong” classifier. This is an iterative algorithm, and in each iteration, a weak classifier is learned so as to minimize the training error. More formally, the algorithm (for the simplified binary (+1 and -1) classification case) is as follows:

—Given the training data from the instance space  $X: (x_1, y_1), \dots, (x_m, y_m)$  where  $x_i \in X$  and  $y_i \in -1, +1$

- Initialize the distribution  $D_1(i) = 1/m$
- For each iteration  $t = 1, \dots, T$  do
  - Train a base learner,  $h_t$ , using distribution  $D_t$ .
  - Update  $D_{t+1}(i) = D_t(i) \exp(-\alpha_t y_i h_t(x_i)) / Z_t$  where  $Z_t$  is a normalization factor and  $\alpha_t$  is the weight of the base learner.
- Then the output of the final classifier is defined as:  
 $H(x) = \text{sign}(f(x))$  where  $f(x) = \sum_{t=1}^T \alpha_t h_t(x)$

This algorithm can be seen as a procedure for finding a linear combination of base classifiers which attempts to minimize a loss function, which in this case is:

$$\sum_i \exp(-y_i f(x_i))$$

A more detailed explanation and analysis of this algorithm can be found in Schapire [2001].

For semi-supervised learning, the idea is to train a model that fits both the human-labeled and machine-labeled data. For this purpose, we first train an initial model using some human-labeled data. Then, the boosting algorithm tries to fit both the machine-labeled data and the prior model using the following loss function:

$$\sum_i (\ln(1 + \exp(-y_i f(x_i))) + \eta \text{KL}(P(\cdot|x_i) \parallel \rho(f(x_i))))$$

where  $\text{KL}$  is the Kullback–Leibler divergence between two probability distributions  $p$  and  $q$ . In our case,  $p$  corresponds to the distribution from the prior model,  $P(\cdot|x_i)$ , and  $q$  corresponds to the distribution from the constructed model,  $\rho(f(x_i))$ , where  $\rho(x)$  is the logistic function  $1/(1 + \exp(-x))$ . This term is basically the distance from the initial model built by human-labeled data and the new model built with machine-labeled data. The weight  $\eta$  is used to control the relative importance of these two terms. This weight may be determined empirically on a held-out set. In order to reduce the noise added because of the classifier errors, we only exploit those utterances that are classified with a confidence higher than some threshold.

### 3.5 Active Evaluation

System performances vary over time for data which exhibits stationary and non-stationary statistics. Typically, these changes are manually monitored, and if it becomes significant, a new model is trained with the newly labeled data. Active evaluation automates this process by controlling the proposed active approach. More specifically, it has three main goals:

- keeping track of performance of the system with respect to a dynamic test set,
- deciding on when to label more data, and
- deciding on when to update the models.

When there is enough labeling capacity to employ active learning all the time, or there is no active or semi-supervised learning capability in place, active evaluation can still be used.

We propose using three ways of evaluating the performance throughout the life-time of an application:

- (1) *Fixed test set* is used to track and compare the performances of the newly trained models in time.
- (2) *Variable test set* is used to handle the nonstationary data. The fixed test set is not enough to see the performance drop due to the changes in language used, class distribution, etc. Therefore, a variable test set is used to compare the performance of a model in time.
- (3) *Sliding window* is the middle way between fixed and variable test sets. The idea is to update the test set with the most recent data by removing the oldest data and keeping most of the test set the same.

Using the performances of these test sets, active evaluation decides on whether to continue manual labeling or not. If there is a significant deterioration in the performance, this means there is a change in incoming data. Then active learning is put in place and selectively sampled data is labeled (and re-labeled). A new model is created and performance is checked using the test sets. If there is a significant improvement, active learning decides on switching to this new model.

Active evaluation does not guarantee to avoid performance drops, but it helps make the system to be more robust to changes in data. Furthermore, it enables a proper and routine evaluation of the system, which is a very valuable performance tracking tool just by itself.

When there is no labeled data available for active evaluation, instead of the actual error, we propose to use the error estimations that are also used for active learning. We call this method *unsupervised active evaluation*. These estimations can be obtained using the candidate test sets described above. In our experiments, we have found out that it is possible to get very reliable error estimations for speech recognition and call classification applications.

#### 4. APPLICATION TO AUTOMATIC SPEECH RECOGNITION

We have applied the proposed active approach to AT&T VoiceTone<sup>®3</sup> Spoken Dialog System (SDS). AT&T SDS aims to identify intents of users, expressed in natural language, and take actions accordingly, to satisfy their requests [Gorin et al. 2002]. Two critical components of this system are automatic speech recognizer (ASR) and spoken language understanding (SLU). In AT&T SDS, first the speaker's utterance is recognized using an ASR, then the intent of the speaker is identified from the recognized sequence, using an SLU component.

---

<sup>3</sup>VoiceTone<sup>®</sup> is a service provided by AT&T, whereby AT&T develops, deploys and hosts spoken dialog applications for enterprise customers.

Table I. The Training and Test Data is Collected from a Spoken Dialog System for Pharmaceutical Domain Customer Care

	Training Set	Test Set
Number of utterances	29,561	5,534
Number of words	299,752	47,684
Vocabulary Size	5,353 words	2,053
Average utterance length	10.14 words	8.62 words
Trigram Perplexity	19.54	26.99

#### 4.1 Error Estimation

In order to estimate the error for ASR, we rely on word confidence scores which are extracted from the lattice output of ASR. To obtain word confidence scores, we combine the posterior probabilities of the transitions in the lattice corresponding to the same word, and which occur at around the same time interval. A detailed explanation of our algorithm and the comparison of its performance with other approaches is presented in Hakkani-Tür and Riccardi [2003].

We used various approaches to obtain utterance level confidence scores  $c_{w_1^n}$  from word confidence scores  $c_{w_i}$  [Hakkani-Tür et al. 2002], and two of them resulted in a better performance. One approach is to compute the confidence score of an utterance as the arithmetic mean of the confidence scores of the words that it contains:

$$c_{w_1^n} = \frac{1}{n} \sum_{i=1}^n c_{w_i} \quad (3)$$

The second approach is using a voting scheme with a threshold. The threshold can be determined using a development set. One can use the threshold which results in the minimum error when determining the correctly recognized words (sum of the false acceptance rate of misrecognized words and the false rejection rate of the correctly recognized words). The words with a score less than the threshold do not have any contribution to the utterance score. The words with a confidence score greater than the threshold contribute to the utterance score by 1. The final utterance score is normalized by the utterance length.

$$c_{w_1^n} = \frac{1}{n} \sum_{i=1}^n tr(c_{w_i}), \quad (4)$$

where

$$tr(c_{w_i}) = \left\{ \begin{array}{l} 1 \text{ if } c_{w_i} > \text{threshold} \\ 0 \text{ otherwise} \end{array} \right\}. \quad (5)$$

#### 4.2 Experiments and Results

We made active learning, semi-supervised learning and active evaluation experiments using AT&T VoiceTone<sup>®</sup> Spoken Dialog System data, collected from a pharmaceutical domain, with some features given in Table I. The trigram perplexity is computed using a trigram language model built from the training set. For ASR, we expect the labelers' transcription errors to be insignificant, hence we have not performed any experiments to check active learning for labeling error detection.

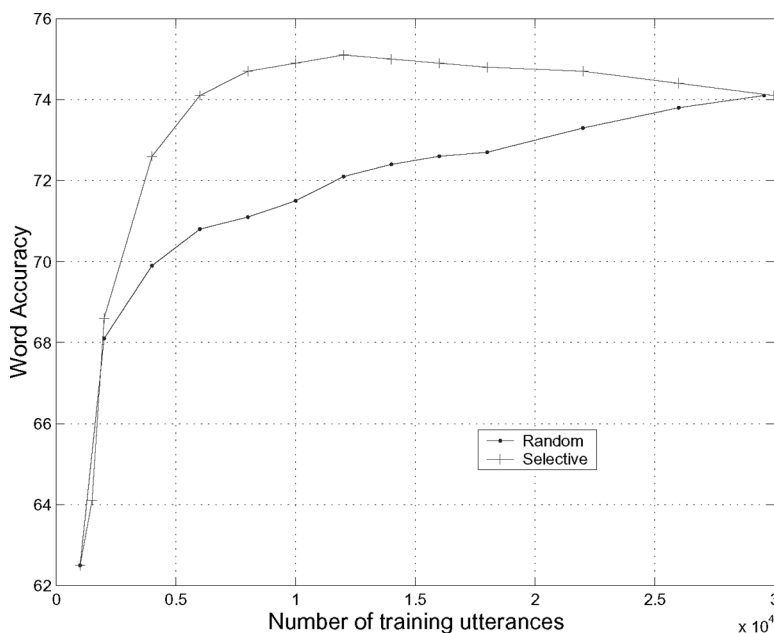


Fig. 7. Word accuracy learning curves with randomly and selectively sampled data.

**4.2.1 Active Learning for Transcription.** In order to show comparisons of  $n$ -gram probability estimation for language modeling and the resulting test set WER for random and selective sampling, we made some experiments using AT&T Spoken Dialog System data. In all the experiments, only the language model is re-trained, and the same acoustic model trained using previously collected telephone speech is used.

**4.2.1.1 Random and Selective Sampling Learning Curves.** Figure 7 shows the word accuracy learning curves for random and selective sampling, obtained using an initial set of 1,000 utterances, randomly selected from the training set. The initial set is used to order the rest of the data for selective sampling. We assume that  $f(|X|)$  (as described in Section 3.2) is linear in the number of examples, still we have shown the learning curves for all training data for completeness. To compute the utterance level confidence scores, we have used the first approach described in Section 4.1. With selective sampling, the lowest scored utterances are added to the transcribed training set first. We obtain the peak word accuracy at around 12,000 training utterances, with selective sampling, which is around 1% better than the word accuracy obtained using all the available training data. These learning curves are obtained by running the recognizer at the same beam-width. Therefore, the run-time performances may differ, due to the differences between the characteristics of language models (such as size, entropy, etc.), as will also be shown in the following sections.

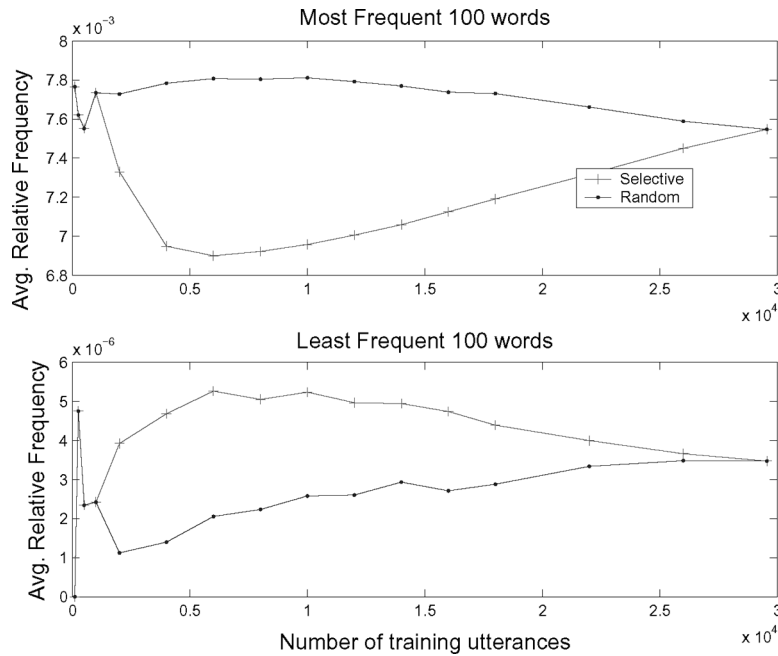


Fig. 8. Average relative frequency learning curves for very frequent (e.g., *Yes* and *Prescription*) and infrequent (e.g., *Above* and *Prepare*) words in our data.

4.2.1.2 *Relative Frequencies*. Figure 8 shows the average relative frequency of 100 most frequent and 100 most infrequent words, as the amount of training data changes. The most frequent and infrequent words are found using the transcriptions of all the data, but the most frequent ones are almost the same as the most frequent words in the initial set. Note that the range and scale of the average relative frequency in the two subplots for frequent and infrequent words are very different.

Figure 9 has the relative frequency plots as the amount of training utterances changes for two very frequent (“Yes” and “Prescription”) and very infrequent (“Above” and “Prepare”) words for this application.

As can be seen from the two sets of plots, the relative frequencies for the first 1,000 utterances are the same with random and selective sampling, as this is the randomly selected initial set for active learning. With random sampling, the average relative frequency of very frequent words converge very quickly, however, that is not the case for infrequent words. This is more clearly seen in the relative frequency learning curves of individual words. Another point to note is that, active learning underestimates the relative frequency of frequent words, and overestimates the relative frequency of infrequent ones. This is a result of our selection mechanism, unseen events are not recognized correctly, and so they get low confidence scores, and therefore are selected by active learning before they can be detected by random sampling. Moreover, with active learning for labeling, the unigram (therefore, also  $n$ -gram) relative frequencies seem to be naturally smoother than the  $n$ -gram probability estimates from random sampling.

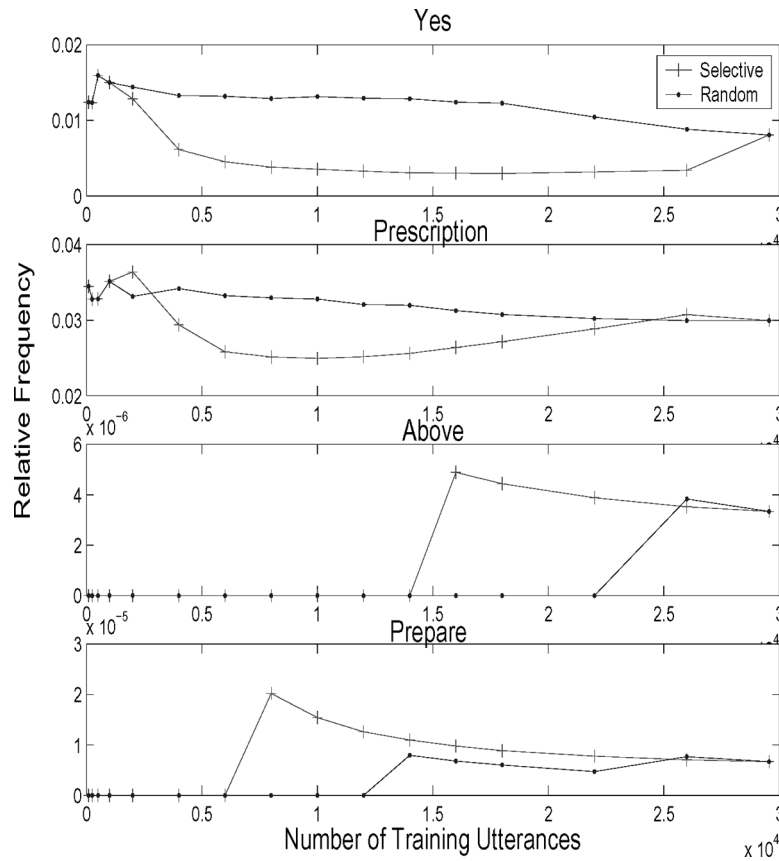


Fig. 9. Relative frequency learning curves for very frequent and infrequent words.

4.2.1.3 *Real Time Speech Recognition.* In the experiment in Figure 7, the beam-width of the recognizer is kept the same for all points. However, the run-time in each experiment changes as a function of the features of the language model, while the acoustic model is the same in all of them. Figure 10 shows the word accuracy versus run-time curves obtained by changing the ASR beam-width, for random and selective sampling with training sets of 8000, 12000, 16,000 and 22,000 utterances. Every subplot also has the run-time curve (the dashed curve) for the language model using all the available data (29,561 utterances). With active learning, when using more than 12,000 utterances are used, the curve does not change significantly<sup>4</sup> as we add the rest of the data, except at very high run-times. With random sampling, the curves improve as we use more data. According to this figure, for example, if we can run ASR at

<sup>4</sup>For example, at around 0.15 times real-time run time, the ASR word accuracy for selective sampling with 12,000, 16,000 and 22,000 utterances (bottom 3 plots) is the same as the word accuracy obtained using all the data. For the run-time in real time values less than 0.3, the difference between *all data* and *selective* curves with more than 12,000 utterances is not significant according to *Z*-test at 0.95 confidence interval.

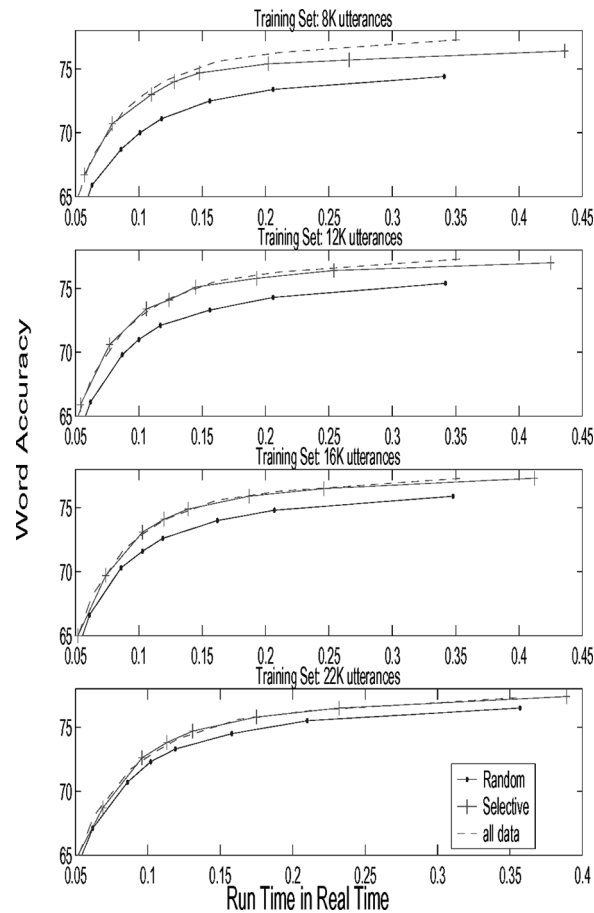


Fig. 10. Word accuracy versus run-times with random and selective sampling. The curve labeled *all data* is the same for all plots and is obtained using all of the available 29,561 utterances.

only 0.1 times real-time, which may be the case for a spoken dialog system, there is not much point in transcribing more than 8,000 selectively sampled utterances, as the performance with 8,000 utterances at that point is the same as the performance obtained using all the data.

**4.2.2 Semi-Supervised Learning.** Figure 11 presents our results using semi-supervised learning. As the initial data, we used some data from Switchboard human-human spoken dialog corpus, and also some data from the web pages related to the application. We again used an off-the-shelf acoustic model.

To compare selective sampling and random sampling, we again plotted learning curves as before, the top one (labeled “Selective”) is plotted using the selectively sampled data. In addition to these, we also used the remaining data, which is untranscribed, to combine active and unsupervised learning.

When we combined randomly sampled transcribed data with the untranscribed data, we got the lower solid curve (labeled “Random+UL”), instead of

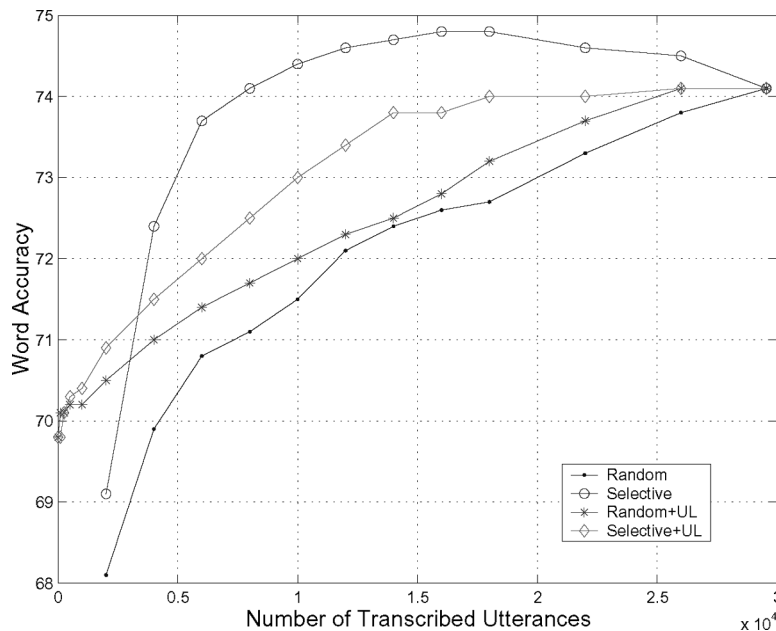


Fig. 11. Word accuracy learning curves when adding untranscribed data to randomly and selectively sampled data. The beam-width is kept constant in all these experiments, and the run-time is  $0.15 \pm 0.015$  fold real time.

the lower dashed curve (labeled “Random”) of random sampling. So, with all data set sizes, we achieved better performance using untranscribed data.

We also added the untranscribed data to the selectively sampled transcribed data, and obtained the upper solid curve (labeled “Selective+UL”), instead of upper dashed curve, which was obtained using only transcribed data. When we did not have much transcribed data, using the ASR output of untranscribed data helped us significantly, but as we started gathering more transcribed data, the untranscribed data started hurting the ASR performance. As we go along the curve, the untranscribed data, added to the selectively sampled data is of very high quality, as we selectively sample those utterances which have a lot of errors at the beginning, and leave the ones that are correctly recognized until the very end.

Therefore, our proposal is combining the two techniques at the beginning of application development, and then switching to transcribed data, as more data is collected. In this way, a reasonable performance can be guaranteed at every phase, even when we do not have any human supervision.

We have also examined the effects of data selection for unsupervised learning. For that purpose, instead of adding the ASR output of all the untranscribed data to our training set, we only added the utterances which are recognized with lower scores, as higher scored utterances seemed to hurt accuracy in the active learning experiments. Figure 12 shows the resulting curve (labeled “Selective + UL with Selective Sampling”) from this experiment as well as the previous active learning curve (labeled “Selective”) and the combination of active and

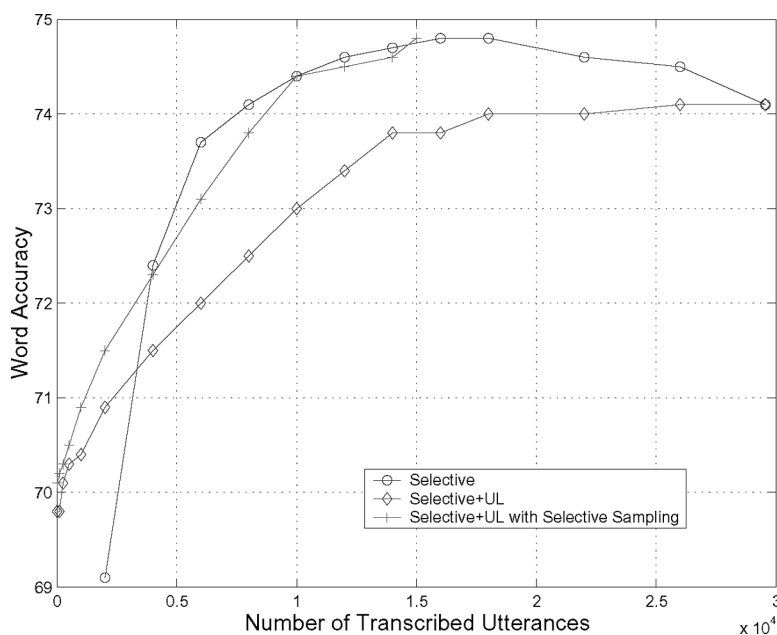


Fig. 12. Word accuracy learning curves with data selection for unsupervised learning.

unsupervised learning curves (labeled “Selective+UL”). As can be seen from the figure, filtering the high confidence utterances from the ASR output resulted in a better performance than adding all the untranscribed utterances. Our aim here is only to show that selective sampling may also be useful for unsupervised learning. Determination of the threshold for utterance confidence score during selection remains as the future work.

**4.2.3 Active Evaluation.** We have evaluated the active evaluation method for ASR using the same data sets. We combined training and tests sets into a pool and arranged them in a time order. We separated the first 1K utterances as the initial training set and trained a model (Fixed Model), and the next 1K utterances as the initial test set (Fixed Test Set). We used the rest to simulate the incoming traffic: we assumed that we get 2K utterances every day, the initial 1K of these are used as the pool to select the utterances to label from, and the next 1K is used as the test data (Variable Test Set). Out of the 1K pool, we assumed that we are able to transcribe 500 selectively sampled utterances, which is added to the previously transcribed model and used to train a new model every day using active learning (AL model). Figure 13 presents word accuracy results using these sets and models, where  $x$ -axis represents the time. The word accuracy of the Fixed Test Set with the Fixed Model is constant in time, and this horizontal line represents the performance with the passive framework. When we use the Fixed Model to recognize the Variable Test Set, which changes every day, the word accuracy drops as the properties of the incoming data may change in time. When we retrain the language model using the selectively sampled data by active learning, and use it to recognize the Fixed

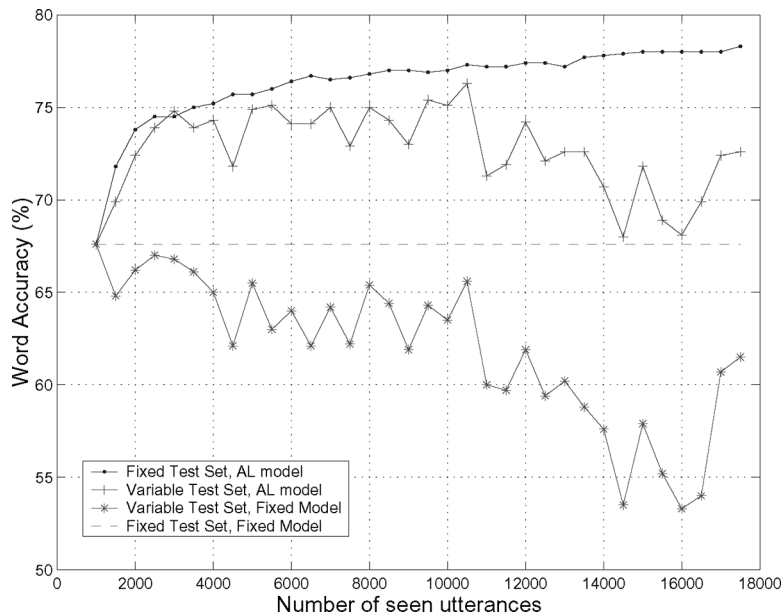


Fig. 13. Effect of active evaluation for ASR.

Test Set, the word accuracy improves as expected. When we use the AL Model to recognize the Variable Test Set, we see that the performance may degrade as the incoming data changes, but since the AL data is also collected from the field, the changes in the performance are not as drastic as the changes with the fixed model.

Transcription of a new test set every day is a costly process. For tracking the traffic changes, we also tried using average utterance confidence scores instead of computing the word accuracy. Figure 14 shows the controlled experiments, where we propose to use the average scores instead of the word accuracy. The top curve is the word accuracy of the Variable Test Set using the initial Fixed Model, as in the previous figure. The bottom curve is the average utterance confidence score for the Variable Test Set using the Fixed Model. Note that to obtain the utterance confidence scores, we do not need to transcribe the test sets. As can be seen in the figure, average confidence score can be used to approximately track the changes in the performance of the language model and signal when to start transcribing new data and update the model.

## 5. APPLICATION TO SPOKEN LANGUAGE UNDERSTANDING

In goal-oriented call routing systems like AT&T Spoken Dialog System, intent determination or SLU is framed as a multi-class multi-label classification problem [Gorin et al. 2002; Tur et al. 2002; Natarajan et al. 2002; Kuo et al. 2002]. As a call classification example, consider the utterance *I would like to know my account balance*, in a customer care application. Assuming that the utterance is recognized correctly, the corresponding intent or the call-type would be *Request(Account.Balance)* and the action would be learning the account

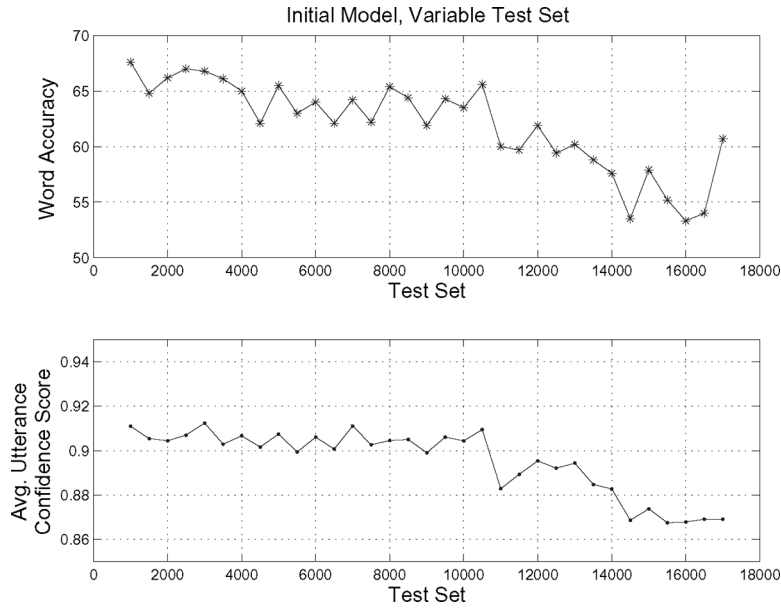


Fig. 14. Effect of unsupervised active evaluation for ASR.

number and prompting the balance to the user or routing this call to the Billing Department.

### 5.1 Error Estimation

For call classification, we employ state-of-the-art data-driven classifiers, namely Boosting. Thus, we use the scores associated to call-types for error estimation for SLU. Actually, many statistical classifiers output some sort of confidence to each of the classes. So our error estimation is independent of the classifier used. For example, a simple naive Bayes classifier tries to estimate the posterior probabilities using the well-known Bayes formula:

$$P(C_i|W) = \frac{P(W|C_i) \times P(C_i)}{P(W)},$$

where  $C_i$  is a call-type and  $W$  is the utterance (or ASR 1-best). Then, the confidence (normalized score to [0–1]) of each call-type can be computed as:

$$P(C_i|W) = \frac{P(W|C_i) \times P(C_i)}{\sum_j P(W|C_j) \times P(C_j)}.$$

In our experiments, we employ the Boostexter [Schapire and Singer 2000] tool, an implementation of the Boosting family of classifiers [Schapire and Singer 1999], and used word  $n$ -grams (phrases) as features. Boosting combines “weak” base classifiers to come up with a “strong” classifier [Schapire and Singer 1999]. This is an iterative algorithm, and in each iteration, a weak classifier is learned so as to minimize the training error. Then, one can define the output of the Boosting classifier,  $f(x, C_i)$ , for each class (call-type),  $C_i$ , for a given

Table II. Characteristics of Data Collected from a Spoken Dialog System for Telecommunications Domain Customer Care System

Number of utterances	40,551
Average utterance length	8.97 words
Number of call-types	65
Unigram Perplexity	15.95

object,  $x$ , as:

$$f(x, C_i) = \sum_t \alpha_t \times h_t(x, C_i)$$

where  $h_t(x, C_i)$  is score of the weak learner for iteration  $t$  and  $\alpha_t$  is the weight of it. The confidence of a call-type,  $C_i$ , is then given by the logistic formula:

$$P(C_i = +1|x) = \frac{1}{1 + \exp(-2 \times f(x))}$$

Then it is possible to use these confidence scores for error estimation in the classification of an utterance in multiple ways. The first approach would be using the confidence score of just the top scoring call-type:

$$Confidence(x) = \max_i P(C_i = +1|x)$$

Another approach would be using the Kullback-Leibler distance of the classifier output distribution,  $\mathcal{C}$  from the prior distribution,  $\mathcal{P}$ :

$$Confidence(x) = KL(\mathcal{C}||\mathcal{P})$$

We used the first approach in our experiments since it is simpler and resulted in similar performances.

## 5.2 Experiments and Results

We made the SLU experiments using AT&T Spoken Dialog System data, from telecommunications domain, with the characteristics presented in Table II. In this application there are 65 call-types with a unigram perplexity<sup>5</sup> of 15.95. We use top class error rate (TCER) in order to compute the performance of the classifier. TCER is the fraction of utterances in which the call-type with maximum probability was not one of the true call-types.

**5.2.1 Active Learning for Labeling.** Figure 15 presents our results comparing random and selectively sampled data. The  $x$  axis is the amount of manually labeled data, and the  $y$  axis is the TCER on the fixed test set. We set aside first 2000 utterances for testing and give the performances on that fixed test set. Once all the data is sorted chronologically, active learning selects 500 utterances among 2000 unlabeled ones to manually label, the remaining 1,500 are ignored. This corresponds to the algorithm presented in Figure 4. Random curve is obtained using the data as is, without any selection. We have got significant

<sup>5</sup>Computed from the prior distribution.

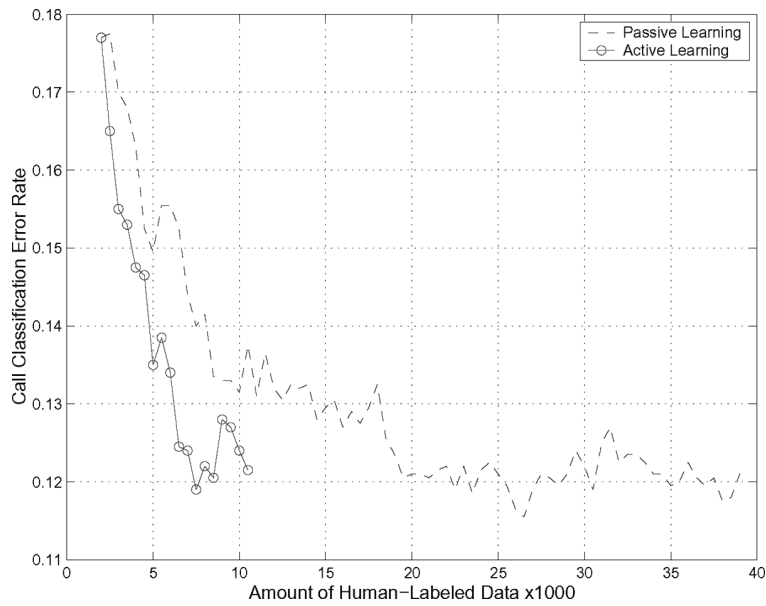


Fig. 15. Effect of selective vs. random sampling for call classification using manual transcriptions.

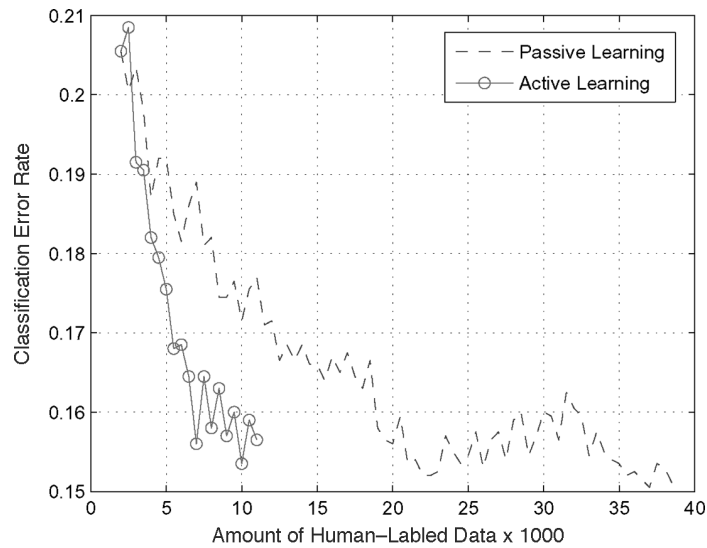


Fig. 16. Effect of selective vs. random sampling for call classification using ASR output.

reduction in the amount of labeled data needed to achieve a given performance. For instance, achieving a test error of 12% requires around 20,000 examples if randomly chosen, but only 7,500 actively selected examples, a savings of 62.5%. In this particular experiment, we use human transcriptions of the utterances. In order to see the effect of noise introduced by the ASR, we repeat this experiment using the ASR output of the same data. Figure 16 presents our results. As seen, using both manual and automatic transcriptions, active learning is

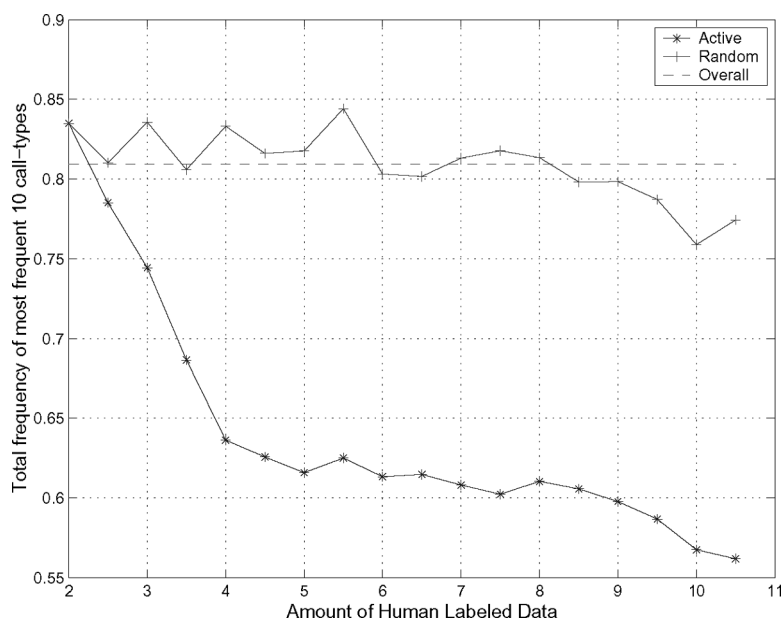


Fig. 17. Effect of active learning to call-type distribution.

as effective. When compared to the previous figure, this time, we do not see the sharp performance improvement at the beginning, probably because of the outliers introduced due to ASR mistakes.

Similar to ASR experiments, we have computed the average relative frequencies of 10 most frequent call-types, as the amount of training data changes throughout the learning curves. Figure 17 presents these results. As seen, there is a continuous decrease in the relative frequency of the most frequent call-types, meaning that active learning selects the examples from less represented call-types more than the ones which are the same as or similar to already known examples.

*5.2.2 Active Learning for Labeling Error Correction.* We have evaluated active learning for labeling error correction method using an application, where we have kept the first and second pass call-types. The test set includes 1,116 utterances, and we have tried using two training sets, one with the same amount of (i.e., 1,116) utterances and the other with 10,044 utterances in order to see the effect of the base model. Figure 18 shows the results of the experiments using the active learning for labeling error correction method for call classification. It shows the ratio of labeling errors found with respect to the ratio of utterances checked, where the ratio of labeling errors found is defined as follows:

$$\frac{\text{The number of erroneously labeled utterances that are found}}{\text{The number of erroneously labeled utterances}}$$

The diagonal dashed line is the baseline where both ratios are equal. This is the performance you may expect without active learning for labeling error correction. We have drawn these curves by putting a threshold on the

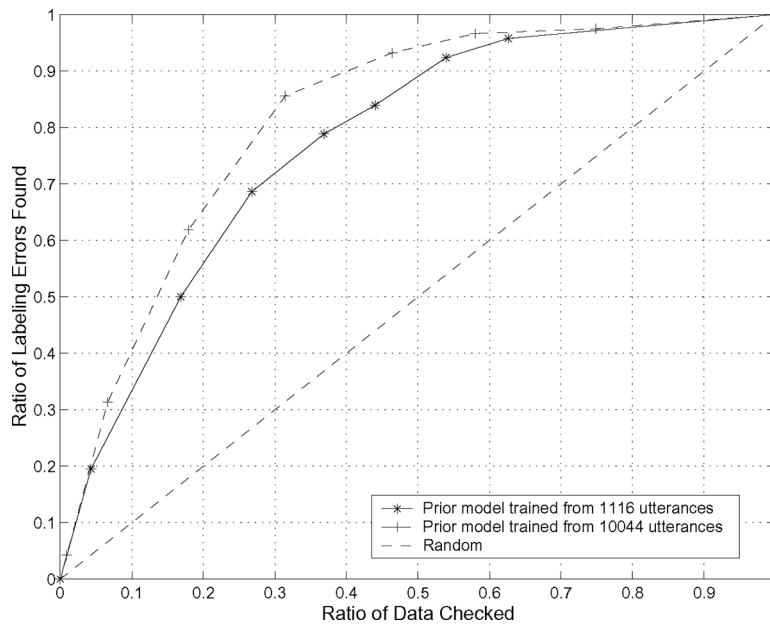


Fig. 18. The ratio of labeling errors found with respect to the ratio of utterances checked using active learning for labeling error correction. The baseline performance, that is, with no active labeling, is the diagonal, where both ratios should be equal.

KL divergence. The solid one is obtained using a prior classification model trained using 1,116 utterances and the dashed curve using all 10,044 utterances. For both curves, this method outperforms the baseline of random selection, even using just 1,116 utterances and finds about 90% of the errors by selecting only half of the data for checking, or finds 75% of the errors by selecting one third of the utterances for checking. Furthermore, active learning for labeling error correction performance increases as the prior model gets better with more data. The percentage of labeling errors found increases from 72% to 83% by using a better prior model when 30% of the utterances are checked.

**5.2.3 Semi-Supervised Learning.** Figure 19 depicts the performance employing semi-supervised learning on top of active learning using the method described in Section 3.4. This figure is the same as Figure 15, where the bottom-most curve indicates the performance for exploiting (instead of ignoring) the 1,500 examples which are not selected by active learning at each iteration. We have observed 0.5–1% reduction in the top class error rate consistently at each iteration. This means greater amount of savings in human labeling and unsupervised adaptation to new events and distributions. For example, achieving an error rate of 11.5% requires around 27,000 examples if randomly chosen, but only 7,500 actively selected and 22,500 unlabeled examples result in the same performance, a saving of 72.2%.

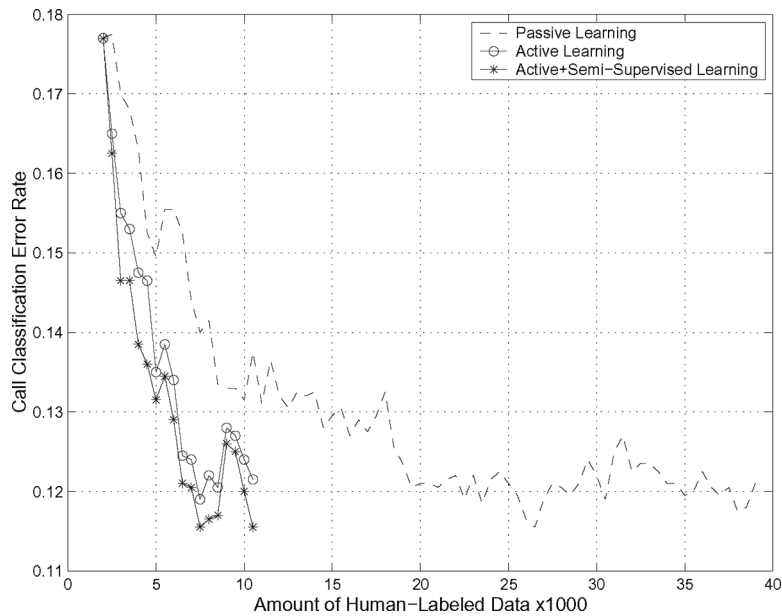


Fig. 19. Effect of active learning combined with semi-supervised learning for call classification.

**5.2.4 Active Evaluation.** In order to evaluate the concept of active evaluation we have made experiments similar to ones done for ASR. Figure 20 shows our results for active evaluation. First, we compute the performance of the call classifier trained with a small number of manually labeled data using variable test sets chronologically sorted covering 3 months. The top class error rate oscillates around 20%–22% for a while, then deteriorates throughout the time range resulting in an error rate of more than 24% (top curve). When we use a fixed test set, the performance is constant throughout the figure (the dashed curve). The bottom curve is the lower bound of employing active learning at every step and computing the performance on the fixed test set. This is the same active learning curve in Figure 15. The curve in between is obtained employing active learning and using a variable test set. Active learning does not guarantee eliminating the deterioration in performance during time, but gives more robustness to changes occurring due to nonstationarity of data as expected. A more realistic scenario would be labeling data when there is a significant decrease in performance. This curve gives a lower bound for that case, since we selectively sample and label data at every step.

Figure 21 justifies that it is feasible to perform active evaluation in an unsupervised fashion when there is no labeled data available. The top curve is the same as the top curve in the previous figure, that is, the performance of the base model with variable test sets. We have drawn the curve below using the average confidences of the top scoring call-types for each test set. As seen, there is a dramatic one to one correspondence with the correct evaluations.

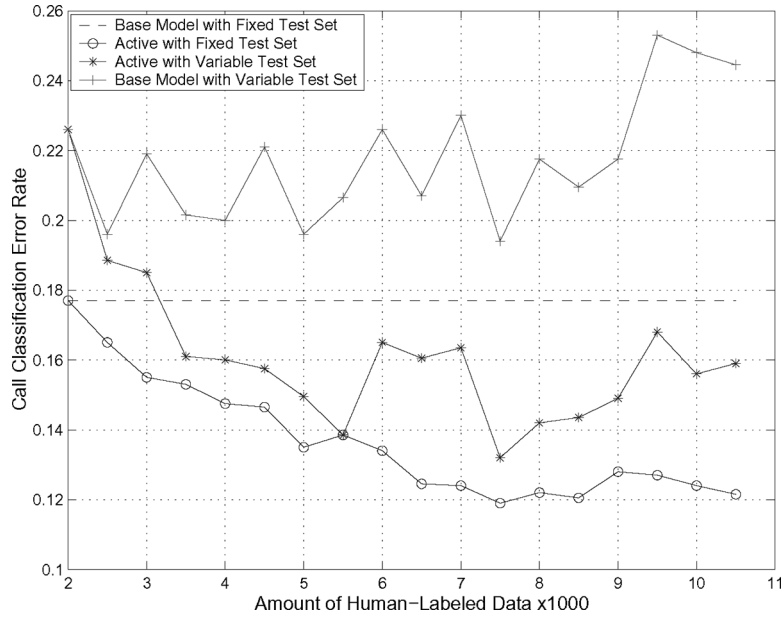


Fig. 20. Effect of active evaluation for call classification.

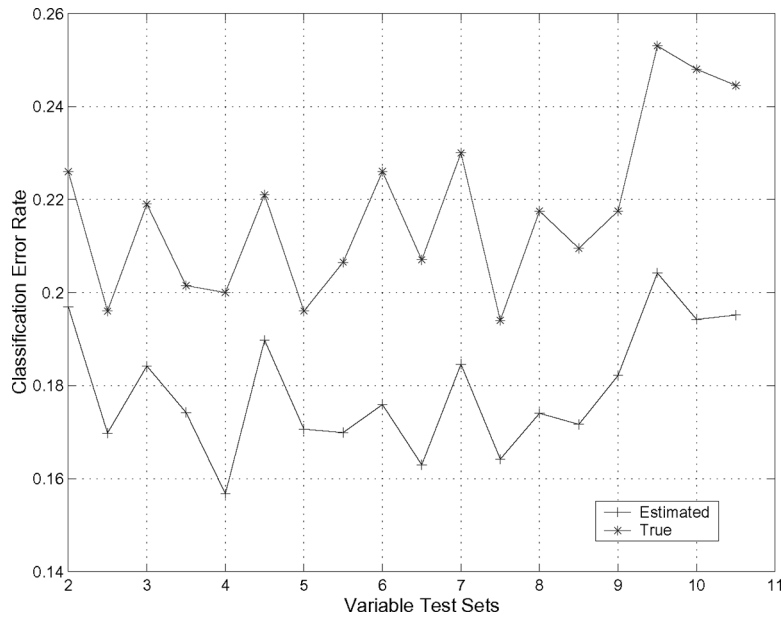


Fig. 21. Effect of unsupervised active evaluation for call classification.

## 6. CONCLUSIONS

We have shown an *active approach* where the system itself selects its own training data, re-trains, evaluates, and deploys itself when necessary. Active approach does not assume that the data is given or stationary, instead it can work with an incoming data stream. It can also track the learning rate to ensure the deployment of best possible model. We have shown experimental results employing active approach for the ASR and SLU components of the AT&T Spoken Dialog System. It is possible to tailor the proposed active approach for specific needs. This approach is general enough to handle any task that includes model training and provides incoming data stream. In that sense, ASR and SLU are just two example tasks for which an active approach can be used.

## REFERENCES

- ABNEY, S., SCHAPIRE, R., AND SINGER, Y. 1999. Boosting applied to tagging and PP attachment. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing (EMNLP) and Very Large Corpora* (College Park, MD). ACM, New York.
- ANGLUIN, D. 1988. Queries and concept learning. *Mach. Learn.* 2, 319–342.
- ARGAMON-ENGELSON, S. AND DAGAN, I. 1999. Committee-based sample selection for probabilistic classifiers. *J. Artif. Intel. Res.* 11, 335–360.
- BACCHIANI, M. AND ROARK, B. 2003. Unsupervised language model adaptation. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (Hong Kong).
- BALDRIDGE, J. AND OSBORNE, M. 2003. Active learning for HPSG parse selection. In *Proceedings of the Conference on Computational Natural Language Learning (CoNLL)* (Edmonton, BC, Canada).
- BLUM, A. AND MITCHELL, T. 1998. Combining labeled and unlabeled data with co-training. In *Proceedings of the Workshop on Computational Learning Theory (COLT)*. Madison, WI.
- COHN, D., ATLAS, L., AND LADNER, R. 1994. Improving generalization with active learning. *Mach. Learn.* 15, 201–221.
- DIGALAKIS, V., RTISCHEV, D., AND NEUMEYER, L. G. 1995. Speaker adaptation using constrained estimation of gaussian mixtures. *IEEE Trans. Speech Audio Process.* 3, 5, 357–366.
- ESKIN, E. 2000. Detecting errors within a corpus using anomaly detection. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)* (Seattle, WA).
- FEDERICO, M. 1996. Bayesian estimation methods for n-gram language model adaptation. In *Proceedings of the International Conference on Spoken Language Processing (ICSLP)* (Philadelphia, PA).
- FREUND, Y., SEUNG, H. S., SHAMIR, E., AND TISHBY, N. 1997. Selective sampling using the query by committee algorithm. *Mach. Learn.* 28, 133–168.
- GHANI, R. 2002. Combining labeled and unlabeled data for multiclass text categorization. In *Proceedings of the International Conference on Machine Learning (ICML)* (Sydney, Australia).
- GODFREY, J. J., HOLLIMAN, E. C., AND MCDANIEL, J. 1990. Switchboard: Telephone speech corpus for research and development. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (Albuquerque, NM).
- GORIN, A. L., ABELLA, A., ALONSO, T., RICCARDI, G., AND WRIGHT, J. H. 2002. Automated natural spoken dialog. *IEEE Comput. Mag.* 35, 4 (Apr.), 51–56.
- GRETTER, R. AND RICCARDI, G. 2001. On-line learning of language models with word error probability distributions. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (Salt Lake City, UT).
- HAKKANI-TÜR, D. AND RICCARDI, G. 2003. A general algorithm for word graph matrix decomposition. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (Hong Kong).

- HAKKANI-TÜR, D., RICCARDI, G., AND GORIN, A. 2002. Active learning for automatic speech recognition. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (Orlando, FL).
- HAKKANI-TÜR, D., TUR, G., RAHIM, M., AND RICCARDI, G. 2004. Unsupervised and active learning in automatic speech recognition for call classification. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (Montreal, Ont., Canada).
- HENDRICKX, I., VAN DER BOSCH, A., HOSTE, V., AND DAELEMANS, W. 2002. Dutch word sense disambiguation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL) Workshop on Word Sense Disambiguation* (Philadelphia, PA).
- HUANG, X., ACERO, A., AND HON, H.-W. 2001. *Spoken Language Processing*. Prentice-Hall, Upper Saddle River, NJ.
- IYER, R., GISH, H., AND MCCARTHY, D. 2002. Unsupervised training techniques for natural language call routing. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (Orlando, FL).
- JELINEK, F. 1997. *Statistical Methods for Speech Recognition*. MIT Press, Cambridge, MA.
- KEARNS, M. 1993. Efficient noise-tolerant learning from statistical queries. In *Proceedings of the ACM Symposium on Theory of Computing* (San Diego, CA). ACM, New York.
- KUO, J., LEE, C., ZITOUNI, I., FOSLER-LUSSER, E., AND AMMICH, E. 2002. Discriminative training for call classification and routing. In *Proceedings of the International Conference on Spoken Language Processing (ICSLP)* (Denver, CO).
- LEGGETTER, C. AND WOODLAND, P. 1995a. Maximum likelihood linear regression for speaker adaptation of continuous density hidden markov models. *Comput. Speech Lang.* 9, 2, 171–185.
- LEGGETTER, M. AND WOODLAND, P. 1995b. Flexible speaker adaptation using maximum likelihood linear regression. In *Proceedings of ARPA Spoken Language Technology Workshop* (Austin, TX).
- LEWIS, D. D. AND CATLETT, J. 1994. Heterogeneous uncertainty sampling for supervised learning. In *Proceedings of the International Conference on Machine Learning (ICML)* (New Brunswick, NJ).
- LIÈRE, R. AND TADEPALLI, P. 1997. Active learning with committees for text categorization. In *Proceedings of the Conference of the American Association for Artificial Intelligence (AAAI)* (Providence, RI).
- MARCUS, M. P., SANTORINI, B., AND MARCINKIEWICZ, M. A. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computat. Ling.* 19, 313–330.
- MCCALLUM, A. K. AND NIGAM, K. 1998. Employing EM and pool-based active learning for text classification. In *Proceedings of the International Conference on Machine Learning (ICML)* (Madison, WI).
- MURATA, M., UTIYAMA, M., UCHIMOTO, K., MA, Q., AND ISAHARA, H. 2002. Correction of errors in a modality corpus used for machine translation using machine-learning. In *Proceedings of the TMI* (Japan).
- MUSLEA, I., MINTON, S., AND KNOBLOCK, C. A. 2002. Active + semi-supervised learning = robust multi-view learning. In *Proceedings of the International Conference on Machine Learning (ICML)* (Sydney, Australia).
- NATARAJAN, P., PRASAD, R., SUHM, B., AND MCCARTHY, D. 2002. Speech enabled natural language call routing: BBN call director. In *Proceedings of the International Conference on Spoken Language Processing (ICSLP)* (Denver, CO).
- NIGAM, K. AND GHANI, R. 2000. Analyzing the effectiveness and applicability of co-training. In *Proceedings of the International Conference on Information and Knowledge Management (CIKM)* (McLean, VA).
- NIGAM, K., MCCALLUM, A., THRUN, S., AND MITCHELL, T. 2000. Text classification from labeled and unlabeled documents using EM. *Mach. Lear.* 39, 2/3, 103–134.
- PRICE, P. J. 1990. Evaluation of spoken language systems: The ATIS domain. In *Proceedings of the DARPA Workshop on Speech and Natural Language* (Hidden Valley, PA).
- REUTERS. 2004. <http://www.daviddlewis.com/resources/testcollections/reuters21578>.
- RICCARDI, G. AND GORIN, A. L. 2000. Stochastic language adaptation over time and state in a natural spoken dialog system. *IEEE Trans. Speech and Audio Processing* 8, 1, 3–9.

- RICCARDI, G. AND HAKKANI-TÜR, D. 2003. Active and unsupervised learning for automatic speech recognition. In *Proceedings of the European Conference on Speech Communication and Technology (EUROSPEECH)* (Geneva, Switzerland).
- RICCARDI, G. AND HAKKANI-TÜR, D. 2005. Active learning: Theory and applications to automatic speech recognition. *IEEE Trans. Speech Audio Process.* 13, 4, 504–511.
- SASSANO, M. 2002. An empirical study of active learning with support vector machines for Japanese word segmentation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*. Philadelphia, PA.
- SCHAPIRE, R. E. 2001. The boosting approach to machine learning: An overview. In *Proceedings of the MSRI Workshop on Nonlinear Estimation and Classification* (Berkeley, CA).
- SCHAPIRE, R. E. AND SINGER, Y. 1999. Improved boosting algorithms using confidence-rated predictions. *Mach. Learn.* 37, 3, 297–336.
- SCHAPIRE, R. E. AND SINGER, Y. 2000. Boostexter: A boosting-based system for text categorization. *Mach. Learn.* 39, 2/3, 135–168.
- SCHOHN, G. AND COHN, D. 2000. Less is more: Active learning with support vector machines. In *Proceedings of the International Conference on Machine Learning (ICML)* (Palo Alto, CA).
- SEUNG, H. S., OPPER, M., AND SOMPOLINSKY, H. 1992. Query by committee. In *Proceedings of the Workshop on Computational Learning Theory (COLT)* (Pittsburgh, PA).
- TANG, M., LUO, X., AND ROUKOS, S. 2002. Active learning for statistical natural language parsing. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)* (Philadelphia, PA).
- THOMPSON, C., CALIFF, M. E., AND MOONEY, R. J. 1999. Active learning for natural language parsing and information extraction. In *Proceedings of the International Conference on Machine Learning (ICML)* (Bled, Slovenia).
- TONG, S. AND KOLLER, D. 2001. Support vector machine active learning with applications to text classification. *J. Mach. Learn. Res.* 2, 45–66.
- TUR, G. AND HAKKANI-TÜR, D. 2003. Exploiting unlabeled utterances for spoken language understanding. In *Proceedings of the European Conference on Speech Communication and Technology (EUROSPEECH)* (Geneva, Switzerland).
- TUR, G., HAKKANI-TÜR, D., AND SCHAPIRE, R. E. 2005. Combining active and semi-supervised learning for spoken language understanding. *Speech Commun.* 45, 2, 171–186.
- TUR, G., RAHIM, M., AND HAKKANI-TÜR, D. 2003a. Active labeling for spoken language understanding. In *Proceedings of the European Conference on Speech Communication and Technology (EUROSPEECH)* (Geneva, Switzerland).
- TUR, G., SCHAPIRE, R. E., AND HAKKANI-TÜR, D. 2003b. Active learning for spoken language understanding. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (Hong Kong).
- TUR, G., WRIGHT, J., GORIN, A., RICCARDI, G., AND HAKKANI-TÜR, D. 2002. Improving spoken language understanding using word confusion networks. In *Proceedings of the International Conference on Spoken Language Processing (ICSLP)* (Denver, CO).
- VAN HALTEREN, H. 2000. The detection of inconsistency in manually tagged text. In *Proceedings of the Workshop on Linguistically Interpreted Corpora* (Luxembourg).
- ZAVALIAGKOS, G. AND COLTHURST, T. 1998. Utilizing untranscribed training data to improve performance. In *Proceedings of the Broadcast News Transcription and Understanding Workshop*.

Received September 2004; revised December 2005; accepted March 2006 by Kishore Papineni