Discriminative Training of Natural Language Call Routers

Hong-Kwang Jeff Kuo, *Member, IEEE*, and Chin-Hui Lee, *Fellow, IEEE*

***Abstract*—This paper shows how discriminative training can significantly improve classifiers used in natural language processing, using as an example the task of natural language call routing, where callers are transferred to desired departments based on natural spoken responses to an open-ended "How may I direct your call?" prompt. With vector-based natural language call routing, callers are transferred using a routing matrix trained on statistics of occurrence of words and word sequences in a training corpus. By re-training the routing matrix parameters using a minimum classification error criterion, a relative error rate reduction of 10–30% was achieved on a banking task. Increased robustness was demonstrated in that with 10% rejection, the error rate was reduced by 40%.**

Discriminative training also improves portability; we were able to train call routers with the highest known performance using as input only text transcription of routed calls, without any human intervention or knowledge about what terms are important or irrelevant for the routing task. This strategy was validated with both the banking task and a more difficult task involving calls to operators in the UK.

The proposed formulation is applicable to algorithms addressing a broad range of speech understanding, information retrieval, and topic identification problems.

***Index Terms*—Discriminative training, information retrieval, natural language call routing, speech understanding, text categorization, topic identification.**

I. Introduction

TOUCH-TONE menus are prevalent in call centers for steering callers to different departments, but callers are often frustrated because the list of options may be long and what they want to do may not even appear to be related to the given choices. In *natural language call routing* (or *steering*), the caller is given a chance to say what he/she wants and is automatically transferred to the correct department or directed to a human operator if the machine is unable to determine the caller's intent with certainty. The goal is to achieve some level of automation and reduce the work load of the human operator without sacrificing customer satisfaction.

Portability is one of the most important issues in the design of natural language call routers. The successful deployment of such systems may be limited by the need for costly human intervention to customize systems and optimize performance for each new call center. It is therefore important to find call classification algorithms that are completely portable and require little human intervention and yet are robust to the demands of natural language dialogue.

Natural language dialogue between human and machine is still a challenging research problem despite the apparent success of some commercial telephony applications using speech recognition. These systems almost always constrain the caller to say a specific set of system-defined keywords that must fit within the finite-state grammar of the task; thus some amount of user training is necessary.

Such systems are appropriate for repeat callers who have successfully adapted to the system but are not appropriate for applications such as customer/operator service, where, although there is a high volume of calls, each call is typically from a unique person who may use the service only once and therefore would not appreciate taking time to learn to interact with the machine. Experienced callers may learn "machine talk" by saying key phrases to which the machine has previously shown understanding, but first-time callers may not even know they are talking to a machine. The caller cannot be prompted with a long list of options, otherwise the service will be as cumbersome as long or nested touch-tone menus.

Callers may not know the specific name of the department they need (e.g. "*home equity loan department*"), but they do know what they want to do (e.g. "*good morning um i wanna speak with someone about um a second mortgage loan do you all do that*"). In fact, a previous study [1] has shown that callers prefer to say what they want to accomplish instead of the name of the department, by a factor of more than three to one.

In natural language call routing, callers are routed to desired departments based on natural spoken responses to an open-ended "How may I direct your call?" or "How may I help you?" prompt [1]–[3]. In designing a voice response system to handle these calls, it is not sufficient to include in the speech recognizer just the names of the departments in the vocabulary or to use a hand-designed finite-state grammar, because what the callers may say cannot be fully anticipated. Instead, requests from real callers have to be collected to train probabilistic language models for the system. Data-driven techniques are therefore essential in the design of such systems. Because of disfluencies and misrecognition by the automatic speech recognizer (ASR), a robust natural language call router/classifier is also necessary in order to process the recognized speech and determine where the call should be steered or whether the system should initiate a disambiguation dialogue [4].

Manuscript received November 28, 2001; revised September 16, 2002. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Hermann Ney.

H.-K. J. Kuo is currently with the IBM T. J. Watson Research Center, Yorktown Heights, NY 10598 USA (e-mail: jeff_kuo@alum.mit.edu).

C.-H. Lee is currently with Georgia Institute of Technology, School of Electrical and Computer Engineering, Atlanta, GA 30332 USA (e-mail: chin.lee@ece.gatech.edu).

Digital Object Identifier 10.1109/TSA.2002.807352

Several papers have been published in recent years on natural language call classification. Among the approaches are those using a vector-based information retrieval technique [1], [4]–[6], using a probabilistic model with salient phrases [2], [7], and using a boosting-based system [8].

In the vector-based information retrieval approach for call routing [1], a routing matrix was trained based on statistics of occurrence of words and word sequences in a training corpus after morphological and stop-word filtering. New user requests were represented as feature vectors and were routed based on the cosine similarity score with the model destination vectors encoded in the routing matrix. Therefore the performance of such a call routing system often depends on the quality of the routing matrix.

In this paper we propose the use of discriminative training on the routing matrix to improve routing accuracy and robustness. Instead of simple counting in conventional maximum likelihood training [1], we use the minimum classification error criterion in discriminative training of the routing matrix parameters. Classification accuracy and robustness are improved by adjusting the models to increase the separation of the correct class from competitors. By retraining the routing matrix, a relative error rate reduction of 10–30% was achieved. Increased robustness was demonstrated in that with 10% rejection, there was a relative error rate reduction of 40%.

To our knowledge, this is the first study of discriminative training using the minimum classification error criterion in an information retrieval problem, and we believe the proposed formulation is equally applicable to algorithms addressing a broad range of speech understanding, information retrieval, and topic identification problems. In particular, although we used the vector space model, which is a simple, fast, and popular model for information retrieval [9], [10], as the baseline framework in this paper, the discriminative training paradigm we are proposing is applicable to any parametrized classifier, including concept Hidden Markov Models (HMMs), probabilistic recursive transition networks, semantic classification trees, and other more complex parametrized models that can better account for structured concepts in natural language.

In Section II we review the formulation of vector-based natural language call routing. Section III describes the discriminative training algorithm used to achieve the minimum classification error criterion for optimizing the call classifier. Section IV describes an initial study on a banking task where we show the effectiveness of discriminative training in reducing the classification error rate [11]. In Section V, we propose several techniques to simplify the training of the routing matrix in order to eliminate human expert knowledge in training the router for new domains. We simplify the features in order to reduce the number of parameters in the model and compensate for the resulting loss in performance by discriminative training [12]. In Section VI, we examine portability issues when designing a new call classifier for an operator task, and demonstrate the portability of our design [13]. In Section VII, we examine in greater detail the discriminatively trained weights and explain intuitively why this algorithm can give much better results through suppressive learning. Finally, in Section VIII, we end with some conclusions.

II. Vector-Based Natural Language Call Routing

In vector-based natural language call routing, call routing is treated as an instance of document routing, where a collection of labeled documents is used for training and the task is to judge the relevance of a set of test documents. Each destination in the call center is treated as a collection of documents (transcriptions of calls routed to that destination), and a new caller request is evaluated in terms of relevance to each destination [1], [4], [6].

The training process involves constructing a routing matrix R. Each document (customer utterances within a call session) is first passed through morphological processing where the root forms of words are extracted. **Ignore** words are eliminated and **stop** words are replaced with place holders. The ignore list consists of noise words common in spontaneous speech, such as *um* and *uh*, while the stop list consists of words that are common but do not contribute to discriminating between destinations, such as *the*, *be*, *for*, and *morning* [4]. We used the same stop list that was used in previously published work [4], which was based on modifying the standard stop list distributed with the SMART information retrieval system [14] to include domain-specific terms and proper names that occur in the training corpus. As an example, the caller utterance "*I am calling uh to apply for a new um car loan*" becomes "$<sw>$ $<sw>$ *call* $<sw>$ *apply* $<sw>$ $<sw>$ *new car loan*" after morphological filtering and ignore and stop word filtering.

After ignore and stop word filtering, n-grams are extracted, specifically unigrams, bigrams and trigrams. (Note that this terminology should not be confused with that for n-gram probabilistic language models, where the term *n-gram* refers to a model of the probability of a word given its history. Strictly speaking, we should be using the terms *single words*, *word pairs*, and *word triplets*, but to be consistent with previous authors [4], we will still use *unigrams*, *bigrams*, and *trigrams*.) Only unigrams that occur at least twice and bigrams and trigrams that occur at least three times in the corpus are included [4]. This leads to a list of F terms (features).

The $F \times K$ term-document matrix is then created. The rows represent the F terms and the columns the K destinations. The routing matrix R is constructed from the transpose of the term-document matrix, where r_{vw} is the frequency with which term w occurs in calls to destination v. Each term is weighted according to term frequency inverse document frequency (*tf-idf*) and are also normalized to unit length. Singular value decomposition and sigmoid transformation were used in the original study [1] but for simplicity can be left out of the discriminative training procedure, without any loss of generality.

New user requests are represented as feature vectors and are routed based on the cosine similarity score with the K model destination vectors $\vec{r}_i$ in the routing matrix R. Let $\vec{x}$ be the F-dimensional observation vector representing the weighted terms that have been extracted from the user's utterance. One possible routing decision is to route to the destination $\hat{j}$ with the highest cosine similarity score

$$\text{destination } \hat{j} = \underset{j}{\text{argmax}} \cos\phi_j = \underset{j}{\text{argmax}} \frac{\vec{r}_j \cdot \vec{x}}{||\vec{r}_j|| ||\vec{x}||}. \qquad (1)$$

Disambiguating questions can be asked by the system if the caller request is ambiguous [4]. The answer provided by the caller would then be processed in a similar manner into a vector that is added to the original request. Based on the augmented request, a new decision is made about which destination is most likely. In this paper, our focus is not on the disambiguating mechanism but rather improving the accuracy and robustness of the routing decision. An improved router accuracy and robustness would improve the dialogue strategy in the sense that fewer disambiguating questions need to be asked, and fewer calls need to be rejected (transferred to the human operator).

A classification error occurs when the score of the correct class is less than the maximum score among the competing hypotheses. Notice that according to the way the routing matrix is constructed, there is no guarantee that the classification error rate will be minimized. The routing matrix can be improved by adjusting the models to achieve a minimum (at least locally, and in the probabilistic sense) in classification error rate, which is the subject of the next section.

III. Minimum Classification Error Criterion

To solve the nonlinear optimization problem of achieving minimum classification error, we adopt the generalized probabilistic descent (GPD) algorithm [15]–[17]. The $K \times F$ elements of the routing matrix are regarded as the classifier parameters to be adjusted to achieve minimum classification error by improving the separation of the correct class from competing classes. The dot product of normalized query and destination vectors is used as the discriminant function. In the training algorithm, the model destination vectors are normalized after each adjustment step in order to maintain the equivalence between the measures of dot product and cosine score used in computing the classification error rate. Intuitively, this algorithm looks at each training example and adjusts the model parameters of the correct and competing classes in order to improve the scores of the correct class relative to the other classes.

Specifically, let $\vec{x}$ be the observation vector and $\vec{r}_j$ be the model document vector for destination j. We define the *discriminant function* for class j and observation vector $\vec{x}$ to be the dot product of the model vector and the observation vector

$$g_j(\vec{x}, R) = \vec{r}_j \cdot \vec{x} = \sum_{i=1}^{F} r_{ji} x_i. \tag{2}$$

Note that this function is identical to the cosine score shown in Equation (1) if the two vectors are pre-normalized to unit length, which we do for both the training and test data.

Given that the correct target destination for $\vec{x}$ is k, we define the *misclassification function* as

$$d_k(\vec{x}, R) = -\, g_k(\vec{x}, R) + G_k(\vec{x}, R) \tag{3}$$

where

$$G_k(\vec{x}, R) = \left[\frac{1}{K-1} \sum_{j \neq k, 1 \leq j \leq K} g_j(\vec{x}, R)^{\eta} \right]^{\frac{1}{\eta}} \tag{4}$$

is the *anti-discriminant function* of the input $\vec{x}$ in class k and $K-1$ is the number of competing classes. Note that in the limit as the positive parameter $\eta \rightarrow \infty$, the anti-discriminant function is dominated by the biggest competing discriminant function: $G_k(\vec{x}, R) \rightarrow \max_{j \neq k} g_j(\vec{x}, R)$. Notice also that $d_k(\vec{x}, R) > 0$ implies misclassification, i.e. the discriminant function for the correct class is less than the anti-discriminant function.

A smooth differentiable 0–1 function such as the sigmoid function is chosen to be the *class loss function*

$$l_k(\vec{x}, R) = l(d_k) = \frac{1}{1 + \exp^{-\gamma d_k + \theta}} \tag{5}$$

where γ and θ are constants which control the slope and the shift of the sigmoid function, respectively. The overall empirical loss for the entire training set consisting of N training vectors is then given by

$$L(R) = \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} l_k(\vec{x}_i, R) \mathbf{1}(\vec{x}_i \in C_k) \tag{6}$$

where the indicator function is defined as

$$\mathbf{1}(\vec{x}_i \in C_k) = \begin{cases} 1, & \text{if } \vec{x}_i \in C_k \\ 0, & \text{otherwise} \end{cases} \tag{7}$$

where C_k represents class k. Note that the empirical loss is essentially a smoothed function approximating the classification error rate that is differentiable so that it can be used in gradient descent optimization.

The parameter set (routing matrix) R is adjusted iteratively according to

$$R_{t+1} = R_t + \delta R_t \tag{8}$$

where R_t is the parameter set at the t-th iteration. The correction term δR_t is solved using the training sample $\vec{x}_t$ given for that iteration, whose true destination is k

$$\delta R_t = -\epsilon_t \nabla l_k(\vec{x}_t, R_t) \tag{9}$$

where ϵ_t is the learning step size.

Once this essential framework for discriminative training has been laid out, what is left is to work out the algebra for this particular problem. Let r_{vw} be elements of the routing matrix R. Then, at iteration step t

$$\nabla l_k(\vec{x}_t, R_t) = \frac{\partial l_k(\vec{x}_t, R_t)}{\partial R_t} = \frac{\partial l_k}{\partial d_k} \frac{\partial d_k(\vec{x}_t, R_t)}{\partial r_{vw}}. \tag{10}$$

Note that for the l_k we have chosen

$$\frac{\partial l_k}{\partial d_k} = \gamma l_k(d_k)(1 - l_k(d_k)). \tag{11}$$

From equations (2), (3), and (4), the following can be shown:

$$\frac{\partial d_k(\vec{x}_t, R_t)}{\partial r_{vw}} = \begin{cases} -x_w, & \text{if } v = k \\ \frac{x_w G_k(\vec{x}_t, R)(\vec{r}_v \cdot \vec{x}_t)^{\eta-1}}{\sum_{j \neq k} (\vec{r}_j \cdot \vec{x}_t)^{\eta}}, & \text{if } v \neq k. \end{cases} \tag{12}$$

Therefore, given the observation vector $\vec{x}_t$ at each iteration, each element of the routing matrix is adjusted according to

$$r_{vw}(t+1) = \begin{cases} r_{vw}(t) + \epsilon_t \frac{\partial l_k}{\partial d_k} x_w, & \text{if } v = k \\ r_{vw}(t) - \frac{\epsilon_t \frac{\partial l_k}{\partial d_k} x_w G_k(\vec{x}_t, R)(\vec{r}_v \cdot \vec{x}_t)^{\eta-1}}{\sum_{j \neq k} (\vec{r}_j \cdot \vec{x}_t)^\eta}, & \text{if } v \neq k. \end{cases} \tag{13}$$

Equation (13) shows that the model vector for the correct class is adjusted differently from those of the competing classes; notice in particular the difference in sign of the adjustment. Intuitively, the score of the correct class is improved relative to the scores of the competitors by the incremental adjustments. The adjustment to the wth component of each model vector is proportional to the learning step size ϵ_t, the size of the wth component in the observation vector $\vec{x}_t$, and the slope of the sigmoid function $(\partial l_k)/(\partial d_k)$. This slope is close to zero for very large or small values of d_k and positive in a certain region: the decision boundary which depends on θ and γ. Only the training data whose d_k values fall within the decision boundary will affect the model parameters significantly.

It is important to process the training data in a random order when making the adjustment steps. After each adjustment step, the affected models $\vec{r}_i$ are normalized to unit length in order that the discriminant function be identical to the cosine similarity score used in classification. The training vectors are normalized once before the discriminative training procedure.

IV. Initial Study

A. Experimental Setup—USAA Banking Task

Experiments were performed using the training and test sets collected for the USAA call routing task as reported in [1], consisting of a total of about 4000 calls. The baseline classifiers reported in this paper are trained on all the training data. For the discriminatively trained classifiers, 80% of the training set used in [1] was used for training, while 20% was reserved as a tuning set to determine the appropriate step size and stopping criterion. The same set of training data was used both to construct the initial routing matrix and for performing discriminative training. Each training vector is constructed from everything said by the customer before the call was routed by the operator. Each call has been manually routed to a destination, representing the ground truth of the correct class. There are a total of $K = 23$ destinations and $F = 756$ terms. In the discriminative training, multiple passes are made through the training set. Within each pass, the order in which each training vector is processed is randomized. For testing, only the 307 unambiguous initial utterances are used from the unseen test set. The baseline system as reported in [1] has a classification rate of 92.2% or error rate of 7.8% for this same set of 307 test utterances. Examples of caller utterances for this banking task include: "*i want the balance on my car loan*" (Loan-Services), "*i just want to know what the new car financing rate is*" (Consumer-Lending), "*just got disconnected but i was calling to ask for an increase on the credit line for my mastercard*" (Credit-Card-Services), "*um i just wanna know how late the drive through windows are open*" (Bank-Hours), "*well um um i need to make a change in electronic fund transfer*" (ACH), and "*oh i eh want to get some deposit slips sent to me for my savings account i called once and i got the envelopes but no deposit slips that's what i'm trying to find out what department i need*" (Deposit-Services).

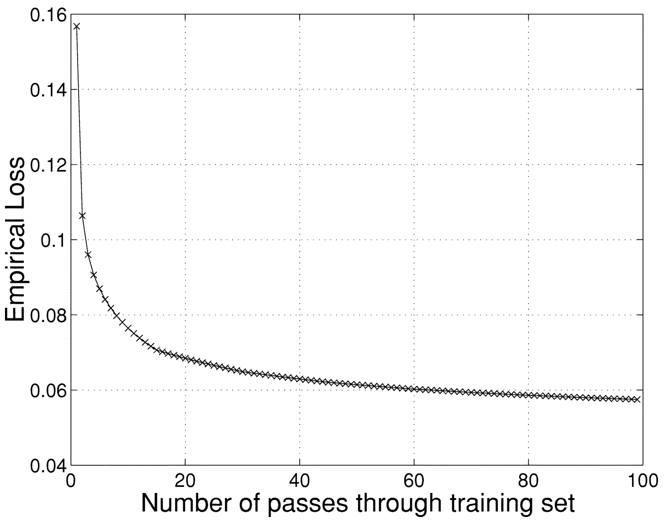

Fig. 1. Empirical loss of the training data versus number of descriminative training passes through the training set.

B. Parameter Selection

We see from the above equations that a number of parameters for GPD training have to be chosen. η controls the relative importance among the competitors—a larger value emphasizes the strongest competitors only. γ and θ controls the decision boundary through modifying the shape and location of the sigmoid function. ϵ_t controls the step size of the gradient descent. It is reduced gradually in order to achieve stable convergence; specifically, the step size is chosen to be a function like $1/t$, but chosen so that it changes only once every 15 passes. Note that $K-1$ is the total number of competitors to the correct class. In practice, the discriminative training can be focused on just the top M competitors instead of all $K-1$ competitors.

Several experiments were run to find appropriate values of parameters to use in the discriminative training. The results do not seem to be too sensitive to most of the parameters. Because the computational cost is low, we were able to run about a hundred passes through the training set. In the following results, we chose the following parameter values: $\eta = 20, \gamma = 32, \theta = 0.0$, $\epsilon_t = 1 \times 10^{-4}$ (initial), and $M = 6$. As we shall see later, the initial step size ϵ_t is obtained by using the tuning set.

C. Results

Fig. 1 shows the empirical loss [as defined in (5)] of the training set as a function of the number of discriminative training passes through the training set. The empirical loss function is a smooth, differentiable approximation of the classification error rate and is the objective that is minimized by the GPD algorithm. The algorithm appears to be doing the correct thing in progressively decreasing the empirical loss. It can be seen from the figure that the ϵ_t step size is reduced over time, specifically once every 15 passes, as seen in the decrease in spacing between the data points and in the level-off of the slope of the empirical loss curve.

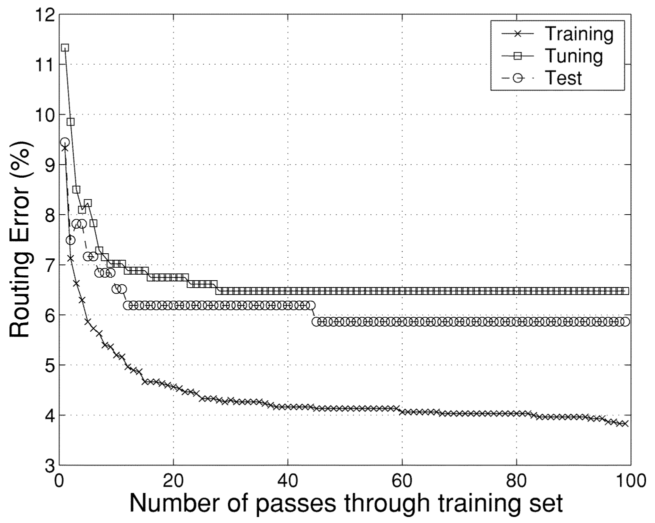

Fig. 2. Classification error rate of the training, tuning, and test data, as a function of the number of discriminative training passes.

Fig. 2 shows the classification error rate (CER) for the training, tuning, and test sets as a function of the number of discriminative training passes. The CER of the training data mirrors the empirical loss function, following a generally downward trend. After about 100 passes, the error rate has been decreased from about 9.3% to 3.8%, a relative reduction of about 59%. The CER of the tuning set is reduced by 43%, from 11.3% to 6.5%. The test set CER is reduced by 38%, from 9.4% to 5.9%.

The GPD algorithm is an iterative algorithm that improves the empirical loss in the training data. One issue is how to define a stopping criterion. In our experiments, we initially chose a threshold in the change of the empirical loss, i.e. we stop if the change in empirical loss is less than a certain threshold. However, the empirical loss of the training data may not generalize to unseen test data; notice, for example, that the empirical loss is still decreasing even after the CER for the tuning and test sets have reached a plateau. Therefore, the tuning set (20% of the original training data) is used to find an appropriate value of the step size so as to have a roughly monotonic decrease in error rate. As we can see from Fig. 2, the tuning set CER may be used to determine the stopping criterion. For example, one can stop when the tuning set CER has been flat for over 30 passes. In any case, Fig. 2 shows that there is not much change in the CER after the first 20 passes, so it is not very important to determine exactly when to stop.

Table I shows a summary of the effects of discriminative training (DT) on the test set. For the test set, we have both human transcriptions of the customer utterances and the recognized strings returned by the automatic speech recognizer (ASR), which has a word error rate of around 20% for this task. For the human transcribed data, the CER is reduced by 24%, from 7.8% to 5.9%. (Note that this baseline result (7.8%) is different from that in Fig. 2 (9.4%) because this baseline classifier is trained on all the original training data, which include the tuning set.) When the ASR recognized strings are used for routing instead, the relative error reduction is around 10%, from 10.1% to 9.1%.

TABLE I
CLASSIFICATION ERROR RATE BEFORE AND AFTER DISCRIMINATIVE TRAINING (DT) USING STOP WORD FILTERING AND 756 FEATURES, CONSISTING OF 62 TRIGRAMS, 274 BIGRAMS, AND 420 UNIGRAMS

	baseline	after DT	%change
human transcription	7.8%	5.9%	24%
ASR recognized strings	10.1%	9.1%	10%

TABLE II
CLASSIFICATION ERROR RATE BEFORE AND AFTER DISCRIMINATIVE TRAINING (DT) USING STOP WORD FILTERING AND 420 UNIGRAM FEATURES

	baseline	after DT	%change
human transcription	12.4%	5.2%	58%
ASR recognized strings	16.3%	8.5%	48%

In the original routing matrix, all the elements are positive because they are derived from the counts of the occurrence of the terms. The discriminative training procedure, as we have formulated it, does not guarantee that the parameters remain positive. In fact, checking the routing matrix after the training reveals that many of the elements have now become negative. This makes sense intuitively since the presence of some terms can provide *negative* evidence against a particular destination, particularly when they are helpful in distinguishing a class from its closest competitors. This interesting issue will be discussed further in Section VII.

V. IMPROVING PORTABILITY

In this section, to improve the portability of the call router, we propose several techniques to simplify the training of the routing matrix in order to eliminate human expert knowledge in training the router for new domains. We simplify the features in order to reduce the number of parameters in the model and compensate for the resulting loss in performance by discriminative training. In particular, we show that we are able to get performance similar to the original optimized system of over 90% accuracy.

A. Results

We first begin by reviewing the baseline results of the natural language call router using 756 features, consisting of 62 trigrams, 274 bigrams, 420 unigrams. As shown in Table I, the classification error rate of transcribed utterances was 7.8% and with discriminative training (DT) could be reduced to 5.9%, representing a relative improvement of 24%. With recognized strings from the speech recognizer (which has a word error rate of about 30%), the improvement is less, only about 10%.

What happens if we use less detailed features in the classifier? As an example, the trigram and bigram features are excluded, leaving only 420 unigram features in the routing matrix. As can be expected, Table II shows that the baseline classification results are worse, the error rate increasing from 7.8% to 12.4%. Using discriminative training (DT) on the unigram features, however, the error rate can be reduced from 12.4% to 5.2%, representing a relative improvement of about 58%, and

TABLE III
CLASSIFICATION ERROR RATE BEFORE AND AFTER DISCRIMINATIVE TRAINING (DT) WITHOUT STOP WORD FILTERING (FULLY AUTOMATIC) AND 1236 UNIGRAM FEATURES

	baseline	after DT	%change
human transcription	9.4%	5.2%	45%
ASR recognized strings	12.1%	8.1%	33%

close to the best error rate with trigram, bigram, and unigram features. With ASR recognized strings, a relative improvement of 48% is achieved, and the resulting error rate is around 8.5%. Thus with this simpler set of unigram features, less computation is required and there is a greater relative improvement using discriminative training.

Recall that the original routing matrix was constructed from n-gram features after preprocessing, where ignore words were eliminated and stop words replaced with place holders. The lists of stop and ignore words are typically manually constructed, because the words that are considered semantically unimportant are different for different contexts and applications. Because such lists require some degree of linguistic knowledge and hand-tuning, supporting natural language call routers for many different domains can be very difficult if not impossible. It is therefore desirable to fully automate the call router design so that such human knowledge and hand-tuning are not necessary. Ideally, the router can be trained using only transcribed utterances and routing destination information.

However, the stop and ignore word lists are necessary to filter out words in order to reduce the total number of features and parameters in the routing matrix. As an example, if no stop and ignore words were filtered, there would have been 7434 features (2756 trigrams, 3442 bigrams, and 1236 unigrams) instead of 756 features. The routing matrix would then have about ten times more parameters (7434×23 in total). Considering that the training set contained less than 4000 calls in total, such a model will not be well trained due to data sparseness.

Since we saw earlier that unigram features together with discriminative training achieved similar results as with all features, we now choose to use only the unigram features (1236 in total) from the list of features which had not been processed with stop word filtering. By not using stop word filtering, we can fully automate the training since no human knowledge is now required to construct the stop word list.

Table III shows the results of using these 1236 unigram features without stop word filtering. As expected, the baseline result in Table III (9.4%) is worse than that using more complex features with stop word filtering (7.8%, in Table I), but is better than that using a smaller set of unigram features that had undergone stop word filtering (12.4%, in Table II). Nevertheless, we see that after discriminative training, the error rate has again been reduced significantly (by 45%) to 5.2%, a result similar to the best results using the other two feature sets. We see therefore that discriminative training is able to bring all three feature sets to similar classification results. For ASR recognized strings, again the final result after discriminative training is similar to those of the other two feature sets.

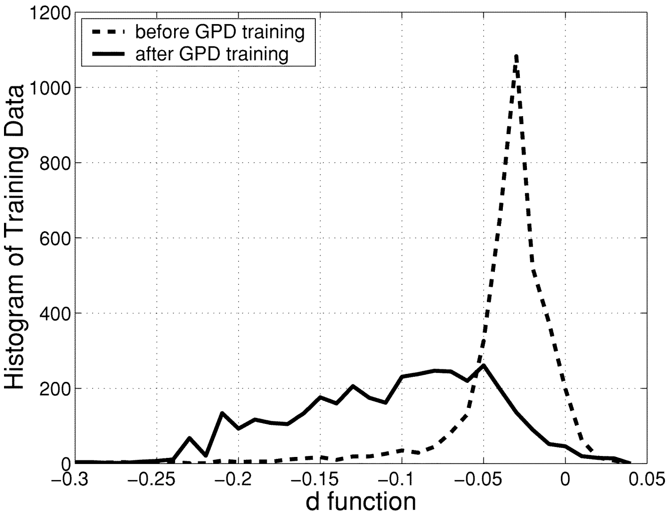

Fig. 3. Distribution of the misclassification function is shifted left after discriminative training, yielding a more robust classifier.

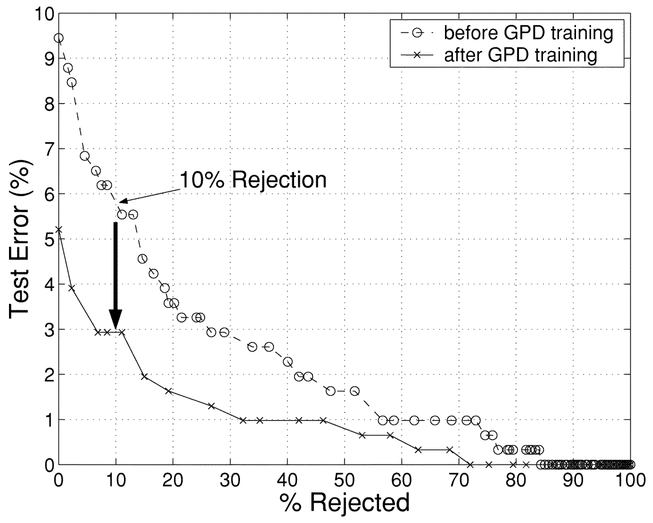

Fig. 4. Error rate versus rejection rate.

We are thus able to achieve similar classification results as the human optimized system using only unigram features that are automatically extracted from the training data without any human expert intervention or specification of stop word lists. Note that bigram and trigram features can still be useful for certain tasks that have finer distinctions than the call departments in the USAA banking task, e.g. the difference between "*transfer money from checking to savings*" and "*transfer money from savings to checking.*" However, models including such bigram or trigram features would require more training data for the parameters to be well trained.

Discriminative training not only reduces the classification error rate, but also improves the robustness of the classifier. Fig. 3 shows the distribution of the misclassification function [as described in (3)] of the training data before and after discriminative training. The distribution has been shifted left, indicating a more robust classifier. (Recall that a positive value implies misclassification.)

Fig. 4 shows the classification error rate of the test data when a subset is rejected using a confidence measure based on the difference in scores of the top two candidates. Note that in this figure the rejected calls are considered correctly routed because it is assumed that a human operator will perform the routing accurately; however, these calls represent missed opportunities for automation. At 0% rejection, as seen earlier, the error rate is reduced from about 9% to about 5% after discriminative training. At 10% rejection for the test data, there is a relative error reduction of about 45% (from about 5.5% to 3%). At almost all levels of rejection, discriminative training consistently does better than the baseline because the separation of the correct class from the competing classes has been increased. Therefore, discriminative training is useful for not only reducing the classification error rate, but also improving the robustness of the classifier. This increased robustness can lead to dramatic cost savings for call centers. For example, suppose we would like to guarantee a routing error probability at or below 2%. From Fig. 4, we see that the baseline system would have to reject over 40% of the calls, whereas the discriminatively trained one would have to reject only about 15% of the calls.

VI. Portability Validation

Since we have demonstrated that we can eliminate stop word selection, an expensive procedure requiring human expert tuning for each task, we wanted to demonstrate the portability of our natural language call router and further examine some of the issues of portability. We had the opportunity to study an operator task in collaboration with BTexaCT, the research division of BT.

Recent papers [18], [19] have described results of joint collaborative research between BTexaCT and Lucent Bell Labs on natural language call steering in the UK, including real field trials. The OASIS task involves calls to operator assistance, accessed through the well known "100" code in the UK [20], called by people for many purposes, including problems with the phone line, setting up alarm calls, directory enquiries, money lost in the pay-phone, etc. This task is very challenging; in human-human recordings, the caller can be very conversational, saying over 300 words in a single utterance.

A. Experimental Setup—BT Operator Task

Experiments were performed using all of the training and test sets from the OASIS corpus described in [19]. For this paper a version of this database was used which had 15 classes, including an "other" set. The "other" class represents rejected utterances that have to be handled by a human operator; thus, in contrast with the USAA banking task, the BT task has an explicit class designed to perform rejection. The results reported in [19] used a later version with an additional ten low-frequency classes added. Classification results have however been found to be broadly comparable between these two different classification schemes. Examples of what the caller may say include "*hello um i've been trying this number in london but it's been constantly engaged*" (Line-Test) and "*i need a number in london it's for this hotel on …*" (Domestic-Directory-Enquiry).

The first engaged utterance [19] for each call has been transcribed and manually assigned a class label, representing the ground truth of the correct class. Each utterance is considered a token that can be used for training or testing. There are a total of about 7400 tokens for training and 1000 for testing.

TABLE IV
Classification Error Rates of Transcribed Utterances Using Classifiers Having Different Sets of Features. The First set Consists of 34 Trigram, 276 Bigram, and 1155 Unigram Features Derived After Stop Word Filtering. The Second set Was Derived Without Stop Word Filtering but Consists of Only Unigram Features

	cosine scores	sigmoid scores
Trigram + task indep. stop words	42.5%	40.4%
Unigram, no stop words	40.6%	29.5%
%difference	4.5%	28%

The same set of training data was used both to construct the initial routing matrix and for performing discriminative training, except for a small set of tuning data used to determine the appropriate value for the step size parameter, as discussed previously. For simplicity, this paper reports only the results on the held out test set consisting of 1000 calls (human-human dialogue), and not data from an actual trial (human-machine dialogue).

B. Results

The main issue we wanted to investigate here was the portability of the natural language call classifier which was originally designed to route calls in the banking application. To this end, the stop word list previously used for the banking task was considered task-independent [4] and was not modified. A new classifier was automatically built without human intervention.

After appropriate filtering as described previously, the number of features chosen was 1465 (34 trigrams, 276 bigrams, 1155 unigrams). The resulting classification error rates on human transcriptions of the spoken utterances are shown in the first line of Table IV. (Note that the terms "trigram" or "trigram model" are meant to refer to a classifier model that uses a feature set containing trigram, bigram, and unigram features.) Using the cosine scores, the classification error was 42.5%, whereas using sigmoid confidence mapping [4], the error rate was 40.4%. The baseline error rates for this operator task are considerably higher than the ones for the USAA banking task because the call classes are inherently more confusable with each other. Note that the classification error rate in this case is defined to be the errors when classifying to one of the 15 classes. The rejection ("other") class is treated just like any of the other 14 classes. We will also report results later in ROC plots that show classification results with rejection.

Also shown in Table IV are the results when all unigram features were used without any stop word filtering. To reduce the number of features, instances of a certain class such as country names were mapped to the country class. A total of 1526 unigram features were used by the classifier. Surprisingly, the classification accuracy was just as good or even better (by up to 28%) than the classifier using a mixture of trigram, bigram and unigram features. One important reason is that the stop word list was not truly task-independent, leading to important terms being excluded from the feature set. For example, all the words

TABLE V
CLASSIFICATION ERROR RATE FOR THE CLASSIFIER USING 7000 TRIGRAM, 5749 BIGRAM, AND 1526 UNIGRAM FEATURES, AS COMPARED WITH THE BASELINE UNIGRAM MODEL (1526 UNIGRAM FEATURES)

	cosine scores	sigmoid scores
human transcription		
Unigram, no stop words	40.6%	29.5%
Trigram, no stop words	23.1%	22.7%
%difference	43%	23%
ASR recognized strings		
Unigram, no stop words	48.7%	41.9%
Trigram, no stop words	37.2%	34.7%
%difference	24%	17%

TABLE VI
CLASSIFICATION ERROR RATE BEFORE AND AFTER DISCRIMINATIVE TRAINING USING 1526 UNIGRAM FEATURES

	cosine scores	sigmoid scores
human transcription		
Unigram, no stop words	40.6%	29.5%
After DT	21.1%	21.9%
%change	48%	26%
ASR recognized strings		
Unigram, no stop words	48.7%	41.9%
After DT	33.9%	34.5%
%change	30%	18%

in the sentence “hi can i have the time please” were considered stop words in the banking task. Therefore stop word lists are frequently task dependent and constructing the list can be costly and require expert knowledge and hand-tuning.

Without stop word filtering, the number of features are likely to increase significantly when trigram features are used. For example, without stop word filtering, but using class mapping and frequency pruning, the number of features derived was 14275, consisting of 7000 trigrams, 5749 bigrams, and 1526 unigrams. This represents almost 10 times the number of features and parameters in the routing matrix (200 K versus 20 K). However, the number of parameters in the routing matrix can be substantially reduced by singular value decomposition (SVD), so it is still feasible to work with this large number of features. Table V shows the classification error rates using these 14 275 features (Trigram, no stop words), for both human transcriptions and ASR recognized strings and compares these results to the baseline unigram results. In general, the trigram model consistently outperforms the unigram model, by as much as 23–43% for transcribed text and as much as 17–24% for ASR recognized strings. (In this paper, we used the real-time recognition results as described in [19] which had a word correct rate of 58.6% and word error rate of 48.1%.)

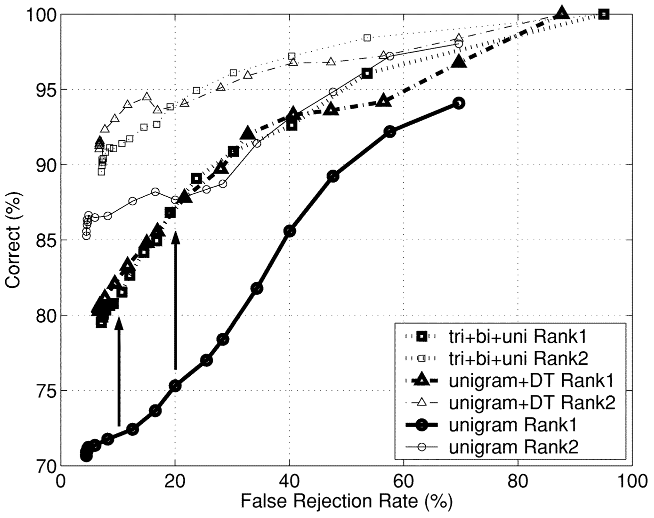

Fig. 5. Semantic classification accuracy for human transcribed text using three different classifiers: the trigram model and the unigram model with and without discriminative training.

Earlier in Section V, we showed that the gap in performance between using the trigram and unigram models in the USAA call routing task can be reduced or even eliminated by using discriminative training, and we wanted to see if this conclusion is also valid for the BT operator task. Table VI shows the results before and after applying discriminative training on the unigram model. The classification error was indeed brought down to the current best level achieved by the trigram model, using about ten times fewer features.

The classification error rates we have mentioned so far treat the “other” class like any of the other 14 classes. However, in order to compensate for the inaccuracy in classification, the call steering system may decide to engage in a disambiguation dialogue or reject the utterance and transfer the call to a human operator [19]. Fig. 5 shows the classification accuracy of the test data as a function of the false rejection rate. Note that this figure differs from Fig. 4 because there is an explicit rejection class (“other”) in the BT task but not the USAA banking task; in addition, it follows the same format reported by other authors [2]. Tokens are considered rejected if they are either classified into the “other” class or if the confidence score for the classification is low. If a rejected token is not a member of the “other” class, this counts as a false rejection; the false rejection rate is defined to be the ratio between the number falsely rejected and the total number of tokens that should have been accepted (do not belong to the “other” class). Out of the tokens that are actually accepted, a percentage of them are classified correctly; this is the classification accuracy shown in this plot.

The figure shows the results for three classifiers: the baseline unigram model (represented by circles), the discriminatively trained unigram model (triangles), and the model utilizing trigram, bigram, and unigram features (squares). Also shown are the Rank1 (heavy lines) and Rank2 (lighter lines) curves. Rank1 refers to the classification accuracy where an utterance is considered correctly classified only if the correct class has the highest score. For Rank2, if the correct class has either the best or second best score, the token is considered correctly classified. The curve for Rank2 gives us an idea of how well we may do if

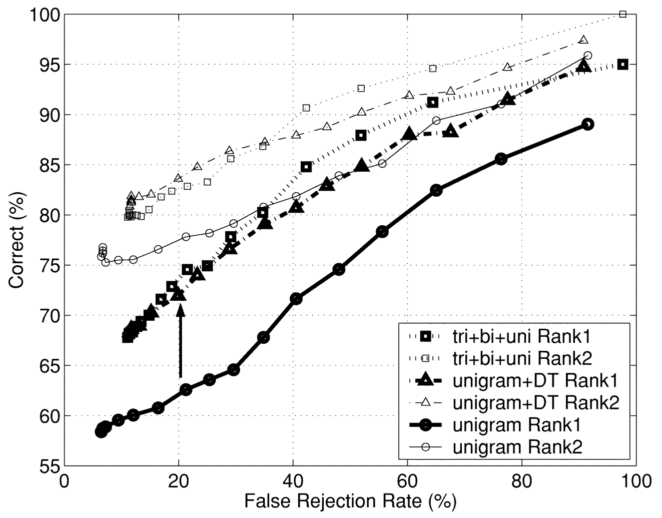

Fig. 6. Semantic classification accuracy for ASR recognized strings using three different classifiers: the trigram model and the unigram model with and without discriminative training.

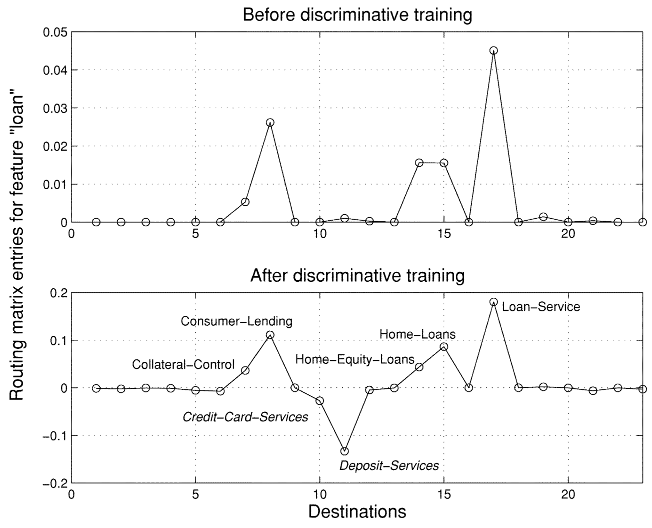

Fig. 7. Routing matrix entries for the feature "*loan*" for the 23 destinations in the USAA banking task *before* and *after* discriminative training.

the system engages the caller in a disambiguation dialogue by asking them to choose between two destinations. As the figure shows, the Rank2 results are much better than Rank1, especially at low false rejection rates (FRR). At 20% FRR, Rank1 had a classification accuracy of about 75% for the baseline unigram model, but Rank2 was about 87% accurate.

After discriminative training, at 10% false rejection rate, the error rate was reduced from about 28% to 18%, a relative reduction of 36%. At 20% false rejection rate, the error rate was reduced from about 25% to 13%, a relative reduction of 48%. At all levels of rejection, the discriminative training consistently does better than the baseline because the separation of the correct class from the competing classes has been increased. In fact the Rank1 curve for the discriminatively trained model is very close to the Rank2 curve for the baseline model for FRR over 20%. Therefore, the utility of discriminative training is not only in reducing the classification error rate, but also in improving the robustness of the classifier.

A similar effect is seen in Fig. 6 where the input is ASR recognized strings. At 20% FRR, the baseline classification accuracy was about 63%. For the discriminatively trained model, the accuracy was about 72%, representing an error reduction of 24%. Although the classification error rate remains quite high even after discriminative training, the call classifier can still be used effectively to automate the routing of a percentage of calls while maintaining a low rate of incorrectly routed calls, through an intelligent application of rejection and dialogue strategy. Details may be found in earlier papers [4], [19]. We can infer from Fig. 6 that for any level of routing accuracy, fewer calls need to be rejected using the discriminatively trained classifier, resulting in increased automation and cost savings to the call center.

C. Other Approaches to Stop Word Filtering

In the literature, there are other approaches to the problem of finding the set of important features for a task, and filtering out unimportant features. One approach is to acquire salient features automatically without human intervention using a salience measure that is motivated by information theory [2]. This measure has been shown to be a nonnegative measure of how much information an event in one channel provides for the random variable of the second channel.

Another approach is the BoosTexter algorithm for text categorization [8] which is based on the boosting algorithm. In this case a set of weak features such as n-grams is considered by the boosting algorithm iteratively to determine the best set of predictors for the task.

Both of these approaches are appealing in their own ways, but we believe that using the discriminative training approach allows the data to speak for itself. An optimal set of weights is determined for the features so that important features are accentuated to achieve minimum classification error, while the weights for unimportant features are automatically reduced through normalization. One possible area of future work is to determine whether discriminative training can be used for pruning the feature set size.

VII. Negative Weights

In the common vector space model for information retrieval [14], many heuristic re-weightings [21] such as *tf-idf* (term frequency inverse document frequency) have been proposed. Discriminative training is a better and more principled way to readjust the weights of the routing matrix because it is based on the minimum classification error criterion. In contrast to heuristic re-weighting schemes, DT can create negative entries in the routing matrix, corresponding to anti-features that give negative evidence for particular destinations. Such suppressive learning is not possible with conventional maximum likelihood learning.

The top figure in Fig. 7 shows the original weights in the routing matrix corresponding to the feature "*loan*" for the 23 destinations in the USAA banking task. The weights are greater than 0.005 for five destinations and mildly positive but less than 0.005 for at least two destinations: Deposit-Services and Operator. The bottom figure in Fig. 7 shows the

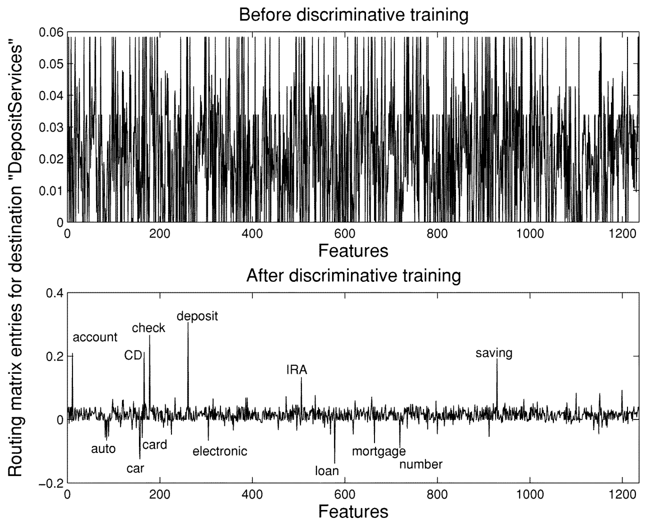

Fig. 8. Routing matrix entries for the destination “Deposit-Services” in the USAA banking task *before* and *after* discriminative training.

weights after discriminative training. The positive weights have been increased for the destinations Collateral-Control, Consumer-Lending, Home-Equity-Loans, Home-Loans, and Loan-Service, all of which have something to do with loans. However, the weights have become significantly negative for two destinations: Credit-Card-Services and Deposit-Services. The presence of the *anti*-feature “*loan*” can be used as evidence to *rule out* these two hypotheses. For the other destinations, the presence or absence of the word “*loan*” does not affect the likelihood judgement for that destination, i.e. the feature is irrelevant.

Fig. 8 shows the values of feature weights for the destination Deposit-Services. After discriminative training, certain distinct features are strong evidence for Deposit-Services (*account*, *CD*, *check*, *deposit*, *IRA*, *saving*) while others are strong evidence against that destination (*auto*, *car*, *card*, *electronic*, *loan*, *mortgage*, *number*). In contrast, no distinct features can be observed before discriminative training. The negative weights after discriminative training are due to suppressive learning, which is not possible with conventional maximum likelihood models.

VIII. Discussion and Conclusion

In this paper we proposed a discriminative training algorithm for natural language call routers that is explicitly formulated to minimize the classification error rate. Other methods have also previously been proposed to try to derive decision rules through explicit error minimization, such as *linear discriminant analysis*, *logistic regression*, and *neural networks* [22], [23]. However, all these algorithms are not based on the minimum classification error (MCE) criterion and thus do not truly achieve minimum error.

To our knowledge, this is the first application of MCE-based discriminative training to discrete features, in contrast to acoustic modeling. For problems with discrete features, the relative amount of training data may be less, and fewer events are observed. This makes discriminative training even more appropriate for improving classifier performance and robustness because it ameliorates data sparseness by training each class model using not only data tokens belonging to the true class but also all those belonging to competing classes. The computation cost is low for the present problem, allowing us to perform about 100 passes over the entire training set, a luxury not available in acoustic modeling. In many cases, however, 15–20 passes were found to be sufficient.

Using the USAA banking task, we first showed that given any classifier, which may already have been optimized for a particular task through manual tuning, discriminative training is still able to provide substantial improvements. Even though the training was done only on transcribed text of the caller utterances, the improvements generalized to the task of classifying text returned by the speech recognizer.

We also studied portability issues for natural language call routers. In porting the call router to a task involving calls to the operator in the UK, we found that one main bottleneck in fully automating the training of the call router is the requirement of a list of stop words, words which are judged irrelevant for a task by a human expert. This list was found to be task dependent and thus inappropriate for the new domain. We successfully demonstrated the feasibility of fully automating the training of the routing matrix by eliminating the need for specifying a stop word list. The only human effort required is to supply training material in the form of transcriptions of callers’ utterances and the destinations to which they should be routed.

We achieved good performance for the admittedly difficult BT task without any tuning. Because the classes in this task are much more confusable than in the USAA call routing task, the absolute classification accuracy is not comparable. Eliminating stop word filtering greatly increases the number of features, and this increase can be compensated by using only single word terms and eliminating the word pairs and triplets. We showed that the resulting performance degradation can be compensated entirely by MCE training. Discriminative training also increased the robustness of the classifier. The discriminatively trained unigram model was shown to have similar performance to the model with trigram, bigram, and unigram features, and had 18–48% fewer errors than the baseline unigram model. Therefore, we believe that the call steering algorithms we have proposed are portable, robust, and require no tuning by human experts.

Finally, we want to reiterate a few main points about discriminative training. The GPD discriminative training algorithm we have proposed is based on the minimum classification error criterion and will work with any probabilistic or parametrized classifier, not just the vector space model. In our experience, the GPD algorithm can reduce classification error by 10–60%, depending on the quality of the initial classifier. Given any classifier that we start with, obtained through any other method, GPD training can always create a better (at least no worse) classifier. In addition to error reduction, robustness is another important benefit of GPD training because it increases the score separation between the correct and incorrect classes (Fig. 3). As a consequence, better rejection based on confidence measures can be achieved, as shown in Fig. 4, 5, and 6. Even if there were no training errors, GPD training can still improve the robustness of the classifier because it considers all the training tokens close to the decision boundary, even those already correctly classified. Moreover, GPD training is insensitive to outliers that are far away from the decision boundary.

One important result of our study is that the training of the classifier can be made task and language independent using discriminative training, eliminating the need to manually tune various parameters or select a stop word list. As an example, we are able to train a natural language call router with the highest known performance using as input only the raw text transcription of routed calls, without any human intervention or knowledge about what terms are important or irrelevant for the routing task.

In summary, this is the first study of its kind to apply MCE-based discriminative training in natural language information retrieval problems. Discriminative training is a powerful technique to improve classifiers to outperform maximum likelihood (or other counting based) classifiers by reducing the classification error rate and by increasing robustness. The specific formulation outlined in this paper can be directly applied to any existing algorithms using a routing matrix, for example in information retrieval [9] and search engines [24]. We believe that discriminative training, which has been successful with acoustic model training and joint feature HMM design [25], will play an increasingly important role in areas of speech understanding, natural language processing, and information retrieval.

Acknowledgment

The authors are grateful to their colleagues at Bell Labs for their support and contributions. They thank J. Chu-Carroll for the baseline natural language call router and F. Soong for insightful discussions. They are indebted to M. Farrell at BT for his help with the BT task, particularly the code to compute ROC plots. They thank the other BT team members: D. Attwater, P. Durston, J. Allen, F. Scahill, and others. Thanks to BT Retail who gave permission for the results to be published. They also thank the three anonymous reviewers for their helpful comments.

References

[1] B. Carpenter and J. Chu-Carroll, "Natural language call routing: a robust, self-organizing approach," in *Proc. ICSLP-98*, Sydney, Australia, Dec. 1998, pp. 2059–2062.
[2] A. L. Gorin, G. Riccardi, and J. H. Wright, "How may I help you?," *Speech Commun.*, vol. 23, pp. 113–127, 1997.
[3] D. L. Thomson and J. J. Wisowaty, "User confusion in natural language services," in *Proc. ESCA Workshop on Interactive Dialogue in Multi-Modal Systems*, Kloster Irsee, Germany, June 1999, pp. 189–196.
[4] J. Chu-Carroll and B. Carpenter, "Vector-based natural language call routing," *Comput. Linguist.*, vol. 25, no. 3, pp. 361–388, 1999.
[5] ——, "Dialogue management in vector-based call routing," in *Proc. 36th Annu. Meeting Assoc. Computational Linguistics (COLING-ACL98)*, 1998, pp. 256–262.
[6] C.-H. Lee, B. Carpenter, W. Chou, J. Chu-Carroll, W. Reichl, A. Saad, and Q. Zhou, "On natural language call routing," *Speech Commun.*, vol. 31, no. 4, pp. 309–320, Aug. 2000.
[7] J. H. Wright, A. L. Gorin, and G. Riccardi, "Automatic acquisition of salient grammar fragments for call-type classification," in *Proc. Eurospeech-97*, Rhodes, Greece, Sept. 1997, pp. 1419–1422.
[8] R. E. Schapire and Y. Singer, "BoosTexter: a boosting-based system for text categorization," *Mach. Learn.*, vol. 39, no. 2/3, pp. 135–168, 2000.
[9] R. Baeza-Yates and B. Ribeiro-Neto, *Modern Information Retrieval*. New York: ACM Press/Addison-Wesley, 1999.
[10] J. R. Bellegarda, "Exploiting latent semantic information in statistical language modeling," *Proc. IEEE*, vol. 88, pp. 1279–1296, Aug. 2000.
[11] H.-K. J. Kuo and C.-H. Lee, "Discriminative training in natural language call routing," in *Proc. ICSLP-2000*, Beijing, China, Oct. 2000.
[12] ——, "Simplifying design specification for automatic training of robust natural language call router," in *Proc. ICASSP-2001*, Salt Lake City, Utah, May 2001.
[13] ——, "A portability study on natural language call steering," in *Proc. Eurospeech-2001*, Aalborg, Denmark, Sept. 2001.
[14] G. Salton, *The SMART Retrieval System*. Englewood Cliffs, NJ: Prentice-Hall, 1971.
[15] S. Katagiri, C.-H. Lee, and B.-H. Juang, "New discriminative algorithm based on the generalized probabilistic descent method," in *Proc. IEEE Workshop on Neural Network for Signal Processing*, Princeton, NJ, Sept. 1991, pp. 299–309.
[16] S. Katagiri, B.-H. Juang, and C.-H. Lee, "Pattern recognition using a family of design algorithms based upon the generalized probabilistic descent method," *Proc. IEEE*, vol. 86, pp. 2345–2373, Nov. 1998.
[17] C.-S. Liu, C.-H. Lee, W. Chou, B.-H. Juang, and A. E. Rosenberg, "A study on minimum error discriminative training for speaker recognition," *J. Acoust. Soc. Amer.*, vol. 97, no. 1, pp. 637–648, Jan. 1995.
[18] W. Chou, D. Attwater, Q. Zhou, P. Durston, H.-K. J. Kuo, M. Farrell, A. Saad, and F. Scahill, "Natural language call steering for service applications," in *Proc. ICSLP-2000*, Beijing, China, Oct. 2000.
[19] P. Durston, H.-K. J. Kuo, M. Farrell, M. Afify, D. Attwater, E. Fosler-Lussier, J. Allen, and C.-H. Lee, "OASIS natural language call steering trial," in *Proc. Eurospeech-2001*, Aalborg, Denmark, Sept. 2001.
[20] M. Edgington, D. J. Attwater, and P. J. Durston, "OASIS—a framework for spoken language call steering," in *Proc. Eurospeech-99*, Budapest, Hungary, Sept. 1999.
[21] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Inform. Process. Manage.*, vol. 24, no. 5, pp. 513–523, 1988.
[22] H. Schütze, D. A. Hull, and J. O. Pedersen, "A comparison of classifiers and document representations for the routing problem," in *Proc. 18th Annu. Int. ACM SIGIR Conf. Research and Development in Information Retrieval*, E. A. Fox, P. Ingwersen, and R. Fidel, Eds., July 1995, pp. 229–237.
[23] D. Hull, J. Pedersen, and H. Schütze, "Document routing as statistical classification," in *AAAI Spring Symp. Machine Learning in Information Access*, Palo Alto, CA, Mar. 1996.
[24] M. W. Berry and M. Browne, *Understanding Search Engines: Mathematical Modeling and Text Retrieval*. Philadelphia, PA: Society for Industrial and Applied Mathematics, 1999.
[25] M. G. Rahim and C.-H. Lee, "Simultaneous ANN feature and HMM recognizer design using string-based minimum classification error (MCE) training," in *Proc. ICSLP-96*, Oct. 1996.

Hong-Kwang Jeff Kuo (M'99) received the S.B. degree in computer science and the S.M. degree in electrical engineering and computer science in 1992, and the Ph.D. degree in electrical and medical engineering in 1998, all from the Massachusetts Institute of Technology, Cambridge.

In 1998, he joined Bell Laboratories, Murray Hill, NJ, as a Member of Technical Staff, where he worked on research in speech recognition and spoken dialogue systems. In December 2002, he joined the IBM T. J. Watson Research Center, Yorktown Heights, NY, as a Research Staff Member. He has published more than 30 papers in journals and international conferences and workshops on topics in language and pronunciation modeling, natural spoken language parsing and understanding, spoken dialogue systems, and models of normal and pathological speech production.

Chin-Hui Lee (S'79–M'81–SM'90–F'97) received the B.S. degree in electrical engineering from National Taiwan University, Taipei, Taiwan, R.O.C., in 1973, the M.S. degree in engineering and applied science from Yale University, New Haven, CT, in 1977, and the Ph.D. degree in electrical engineering with a minor in statistics from the University of Washington, Seattle, in 1981.

He joined Verbex Corporation, Bedford, MA, and was involved in research on connected word recognition. In 1984, he became affiliated with Digital Sound Corporation, Santa Barbara, CA, where he engaged in research and product development in speech coding, speech synthesis, speech recognition, and signal processing for the development of the DSC-2000 Voice Server. Between 1986 and 2001, he was with Bell Laboratories, Murray Hill, where he became a Distinguished Member of Technical Staff and Director of the Dialogue Systems Research Department. His research interests include multimedia communication, multimedia signal and information processing, speech and speaker recognition, speech and language modeling, spoken dialogue processing, adaptive and discriminative learning, biometric authentication, information retrieval, and bioinformatics. His research scope is reflected in a bestselling book entitled *Automatic Speech and Speaker Recognition: Advanced Topics* (Norwell, MA: Kluwer, 1996). From August 2001 to August 2002, he was a Visiting Professor at the School of Computing, The National University of Singapore. In September 2002, he joined the Faculty of the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta. He has published more than 250 papers and 25 patents on the subject of automatic speech and speaker recognition.

Dr. Lee has participated actively in professional societies. He is a member of the IEEE Signal Processing Society, Communication Society, and the European Speech Communication Association. He is also a lifetime member of the Computational Linguistic Society in Taiwan. In 1991–1995, he was an associate editor for the IEEE TRANSACTIONS ON SIGNAL PROCESSING and TRANSACTIONS ON SPEECH AND AUDIO PROCESSING. During the same period, he served as a member of the ARPA Spoken Language Coordination Committee. In 1995–1998, he was a member of the Speech Processing Technical Committee of the IEEE Signal Processing Society (SPS) and chaired the Speech TC from 1997 to 1998. In 1996, he helped promote the SPS Multimedia Signal Processing (MMSP) Technical Committee in which he is a founding member. He received the SPS Senior Award in 1994 and the SPS Best Paper Award in 1997 and 1999, respectively. In 1997, he was also awarded the prestigious Bell Labs President's Gold Award for his contributions to the Lucent Speech Processing Solutions product. More recently he was named one of the six Distinguished Lecturers for the year 2000 by the IEEE Signal Processing Society.