DISCRIMINATIVE TECHNIQUES IN CALL ROUTING

Stephen Cox

School of Information Systems, University of East Anglia, Norwich NR4 7TJ, U.K.
sjc@sys.uea.ac.uk

ABSTRACT

Call-routing is a technology that attempts to route automatically a telephone query from a customer to one of a number of destinations. In vector-based call-routing, a query is represented in a high-dimensional vector space whose axes correspond to words, or sequences of words, that appear in the vocabulary used by callers. In this paper, we examine three different discriminative techniques applied to call-routing. Although some of these techniques give very substantial reductions in error-rate on the training-set, performance on the test-set is disappointing, the most likely reason being a lack of generalisation. Using examples of mis-classified calls, we speculate on why this might occur and propose an improved approach.

1. INTRODUCTION

"Call routing" refers to the technique of automatically relaying a customer's telephone enquiry to the appropriate destination, using computational speech and language processing techniques. The potential benefits of such a technology are obvious to anyone who has used the slow and frustrating systems which are currently universally provided when one telephones a company, institution, government department etc. The user responds to prompts from these systems using touch-tones, but the menus are rigid and it may require navigation through several levels of menu to reach the destination appropriate to the query.

In a call routing system, the prompt to the customer is deliberately general (e.g. "Please state your query or request", or "Please say which service you would like"). In contrast to the typical "Please say yes or no" prompts encountered in current voice dialogue systems, this prompt elicits a wide range of responses. These responses can be very different in length, ranging from single words (e.g. "Mortgages") to long responses that may be syntactically and semantically complex or ambiguous, and that may incorporate a large vocabulary (e.g. "There's a transaction on my account that isn't my charge so I need to talk to somebody about getting this removed"). However, the task is made feasible by the fact that the number of possible "destinations" for a call is usually quite low (< 40) and most calls can be unambiguously routed to a single destination.

In this paper, we consider the vector-based approach to call routing. The vector-based approach represents a spoken query as a vector within a high-dimensional vector space whose axes correspond to words, or sequences of words, that appear in the vocabulary used by callers. Standard pattern processing techniques may then be used to classify the query and hence route it to its correct destination. This approach is an alternative to the probabilistic approach, in which the likelihood of the set of query words being associated with a particular destination is estimated, and statistical techniques used to decide the significance of this likelihood [7]. Chu Carroll and Carpenter have shown that the vector based technique offers superior performance on a call-routing problem with 23 destinations [2].

This paper is organised as follows: in section 2, we describe the problem studied here and the data-set used. We also describe the approach to classification taken in the paper and the details of the construction of a baseline classifier. Section 3 describes in detail the discriminative algorithms that were investigated in this work, and section 4 reports results on both the training and development sets. Section 5 concludes with a brief discussion and some pointers to how progress might be made in this technology.

2. DATA AND BASELINE CLASSIFIER

2.1. Data

The application studied here was the enquiry-point for the store-card for a large retail store. Customers were invited to call up the system and to make the kind of enquiry they would normally make when talking to an operator. Their calls were routed to 61 different destinations, but some destinations were used very infrequently. 95% of the calls were routed to the top 35 routes, and these were the calls used in this study. Each call was transcribed and labelled by an expert with the appropriate destination (e.g. "I need my account balance, please" would be routed to *Balance*, "My card was stolen" to *LostCard* etc.). The transcriptions were divided into a training-set of 6674 queries and a development set of 4713 queries. In fact only 4642 of the training-set queries consisted of a unique set of words, and it was this reduced set that was used for training the algorithms. Note that the experiments reported here used only these transcriptions; no speech recogniser output was used at this stage.

2.2. Classifier

To construct the classifier, the training-set transcriptions and their associated labels were used to construct a matrix W. The rows of W are called the "terms", and correspond to different single words or sequences of words that appear in the transcriptions. The columns of W correspond to the different destinations. Entry W_{ij} is the number of times that term t_i appears in destination r_j in the training-data. W is then transformed into a new matrix X which is in turn used to classify a query whose destination is unknown. X always has the same number of columns as W (= the number of different destinations), but may have fewer rows (see section 3).

To classify a query whose destination is unknown, the query's transcription is first represented as an additional column vector of W. This extended matrix W' is appropriately transformed to an extended matrix X', and the additional column vector of X' is then classified by comparing it with the other column vectors of X'. Two different techniques were used to classify the query vector q:

1. Dot-product: Classify q as destination r^*, where r^* maximises $x_k \cdot q$, $k = 1, 2, \ldots, Nroutes$. ($x_k$ is the k'th column

vector of X and $\cdot$ means "scalar product").

2. Euclidean difference: Classify q as destination r^*, where r^* minimises $(x_k - q)(x_k - q)^T$, $k = 1, 2, \ldots, Nroutes$.

This paper concentrates on exploring different discriminative transformations of $W \rightarrow X$ with the aim of achieving the best classification accuracy of unseen queries. The matrix of counts, W, is a poor discriminator of the destinations, because the magnitude of the count of a term is only weakly related to the importance of the term for identifying the correct destination. The highest counts in any of the columns of W are mostly due to terms representing function words such as "a", "the", "of", "and" etc., words which convey no information about the destination. This problem has been long appreciated in the field of information retrieval, and various weighting schemes for W (which we can regard as transformations) have been proposed to improve discrimination.

For the baseline classifier, the transformation $W \rightarrow X$ consisted of a weighting of the elements of W. This weighting is due to Bellegarda [1] and has been found previously to perform well [4]—details are given in either of these papers. The baseline classifier used this weighting together with the dot-product classifier.

2.3. Term generation

One approach to generating the set of terms to be used in W is to make a "stop-list" of words that do not contribute to identification of any destination and to delete these words from the transcriptions before constructing W. A simple way of generating such a stop-list (used in [5]) is to measure the mutual information (MI) between each word and the destinations and to place on the stop-list any word with a value of MI lower than some threshold. The stop-list has the effect of reducing a phrase such as "what is the balance on my account" to something like "balance account". A disadvantage of using a stop-list is that it is then impossible to make use of collocations, word sequences that can be regarded as operating as single words. Collocations are commonly effective compound nouns such as "account balance", "available credit", or phrases such as "i'd like to", "my card was stolen". Collocations contain information useful for routing purposes which is lost when words are removed using a stop-list. For instance, the query "did you get my check" might be reduced to just "check" by using a stop-list, since the four preceding words are very common. However, although the individual words "did", "you" and "get" have low information content, the collocation "did_you_get" is highly correlated with the destination *lastpayment* and so it is useful to retain it.

In this work, no stop-list was used, and terms consisted of all the different single words in the vocabulary plus selected collocations. Collocations were selected experimentally by measuring the routing accuracy when different terms were used. The identity of the terms was varied by varying the maximum number of words in a sequence of words comprising a term from two words to six words. Hence if this maximum were set to three, the query "i want a new credit card" would yield the terms "i", "i_want", "i_want_a", "want", "want_a", "want_a_new" etc. etc. We also varied the number of times that a sequence had to be seen in the training data before it was used as a term. The result was that the best performance was obtained when the number of words in a sequence comprising a term was a *maximum* (six) and the number of times above which a sequence is regarded as a term is a *minimum* (twice). However, applying these two criteria leads to an explosion in the number of terms in the training-data. There are 1494 different words in the training data, but if the above criteria are applied, the result is 17875 different terms, which is too high to use practically as a vector dimensionality. As a comprise, our terms included collocations of up to three words which occurred three times or more in the training data, which results in 6089 terms and only a small increase in error. However, in some experiments, collocations were not used, so that the number of terms was 1494 (the minimum number if no stop-list is used).

3. DISCRIMINATIVE TECHNIQUES

The entropy weighting described in [1] can be viewed as a simple discriminative weighting. Many different weighting schemes have been proposed and tested to optimise information retrieval from both documents or speech recognition output. Initially, we proposed and tested several new *ad hoc* weightings that we reasoned should be an improvement over the weighting of [1]. However, performance was generally similar to or worse than baseline, and it was clear that an automatic and principled way of improving discrimination performance was desirable. In this section, we describe three techniques that were developed and evaluated for this task:

1. Generalised Probabilistic Descent (GPD)
2. Corrective Training (CT)
3. Linear Discriminant Analysis (LDA)
4. LDA followed by CT

We now describe these techniques in detail.

3.1. Generalised Probabilistic Descent (GPD)

In [8], Kuo and Lee described a discriminative technique for vector-based call-routing which is based on probabilistic descent. As in this work, destination classification in their work is based on taking the dot-product between an input test vector and the column vectors of a matrix X, each column vector corresponding to a different destination, and finding the maximum. The algorithm seeks to adapt the elements of X in such a way as to minimise the term $d_k = G_k - g_k$, where g_k is the dot-product of the test vector x to the *correct* class and G_k is a function of the sum of the dot-products of x with column vectors of the *incorrect* classes. Minimisation is accomplished by forming a smooth function of d_k, L_k, using a sigmoid, differentiating L_k with respect to the elements of X, and then updating these elements using a gradient descent algorithm. Full details of the algorithm are given in [8].

Kuo and Lee's algorithm was implemented as described in their paper. The only difference was in the pre-processing, where we used the weighting described in [1] rather than the inverse document frequency weighting used in [8]. The reduced (unduplicated) training-set was used to form the baseline classifier and also provided the test vector utterances for adapting the weights of the matrix X. Because of the computational intensity of this algorithm, the terms consisted of the 1494 different words with no collocations used. The algorithm parameters used were similar to the ones used in Kuo and Lee's experiments. Iteration on the training-set was carried out until no further improvement in the error-rate was available (see Figure 1).

3.2. Corrective Training(CT)

Nilsson described a simple procedure for adaptation of a classifier for a multi-class problem [9]. The algorithm is as follows:

```
Iterate over training-set vectors:
    If test vector v_i has destination r_j but
       is mis-classified as r_k:
```

$x_j \rightarrow x_j + \varepsilon(v_i - x_j)$
$x_k \rightarrow x_k + \varepsilon(v_i + x_k)$

Else `if classification correct:`
`Do nothing.`

where x_j is the column vector representing the true class and ε a positive constant. In other words, when a mis-classification occurs, the algorithm moves the vector representing the true class closer to the input vector, and the vector representing the recognised class further away. Clearly, the choice of ε is important; intuitively, if the test vector is much closer to the recognised class than the true class, the amount of movement should be more than if the distances are more equal. We used an empirical value of ε given by $\varepsilon = \alpha[(v_i \cdot x_k)/(v_i \cdot x_j)]^{\eta}$. where α is a small positive constant. This gives a value of ε that has a minimum value which is greater than α, and that increases as the mis-classification gets "worse" in the sense that the ratio $(v_i \cdot x_k)/(v_i \cdot x_j)$ increases. Experimentation with different values of η showed that values greater than one tended to make the adaptation process unstable. Adaptation of the columns of X was done in "batch" mode: during an iteration in which the training-set vectors were classified, the values of x_j, x_k, v_i and ε were stored for each mis-classification and then used to adapt X at the end of the iteration. Prior to iterating the algorithm, both the input vectors and the column vectors of X were normalised to unit vectors, and after each iteration, the column vectors were re-normalised. α was decreased on each iteration by 5%. Iteration was performed until the error-rate on the training-set appeared to be fluctuating about a low value: typically, one hundred iterations were used (see Figure 1).

3.3. Linear Discriminant Analysis (LDA)

Linear Discriminant Analysis (LDA) [10] is a discriminative classification technique that is implemented by applying a linear transformation to the training and query vectors prior to classification. LDA has the attractive features of making no assumptions about the distribution of the vectors and of reducing the dimensionality of the space to $N-1$, where N is the number of classes. In this problem, it therefore reduces the dimensionality from 6089 (or 1494 if no collocations are used) to 34.

The transformation matrix $\mathbf{A}$ is the matrix of eigenvectors $\mathbf{a}_i$ that satisfy the equation

$$\mathbf{S}_W^{-1}\mathbf{S}_B\mathbf{a}_i = \lambda\mathbf{a}_i, \quad (1)$$

where $\mathbf{S}_W$ is the within-class scatter matrix, $\mathbf{S}_B$ the between-class scatter matrix and λ an eigenvalue. However, $\mathbf{S}_W$ is singular in our application and so we cannot compute $\mathbf{S}_W^{-1}$ directly. Instead, we compute $\mathbf{A}$ as

$$\mathbf{A} = \mathbf{U}_r\mathbf{\Lambda}_r^{-1/2}\mathbf{V}_r, \quad (2)$$

where $\mathbf{U}_r = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_r]$ is the matrix of eigenvectors of $\mathbf{S}_W$ with non-zero eigenvalues, $\mathbf{\Lambda}_r$ is a diagonal matrix of non-zero eigenvalues of $\mathbf{S}_W$ and $\mathbf{V}_r$ is the matrix of eigenvectors of $\mathbf{S}'_B = \mathbf{\Lambda}_r^{-1/2}\mathbf{U}_r^T\mathbf{S}_B\mathbf{U}_r\mathbf{\Lambda}_r^{-1/2}$. Note that $\mathbf{S}_W$ is constructed directly from the 6674 vectors of counts of terms and no weighting is applied to these vectors prior to LDA. Note also that since LDA is based on a maximisation of a Euclidean distance between transformed vectors, we use the Euclidean distance between the (transformed) query vector and the refence column-vector rather than the dot-product for classification.

We also experimented with applying CT to the transformed vectors generated by LDA.

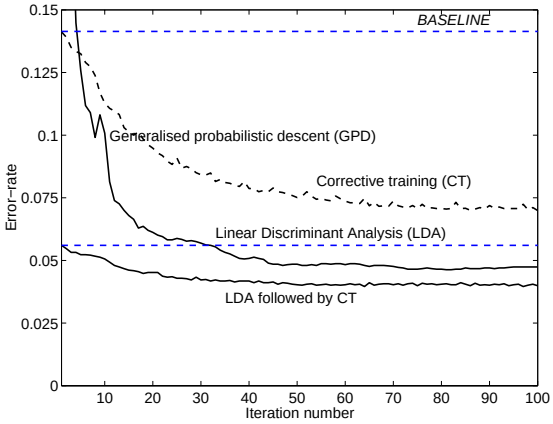

Fig. 1. Error-rates on the training-set

4. RESULTS

4.1. Training-set

Three of the algorithms investigated (GPD, CT and LDA+CT) require iteration to produce the classifier, and the error-rate on the training-set during iteration was measured. Results are shown in Figure 1. Figure 1 also shows the baseline error on the training-set (14.12%) and the error after application of LDA (5.30%). Note that the GPD algorithm starts from a higher baseline error than CT because it was run with single words as terms rather than three-word collocations. The algorithms were iterated 100 times. After this, no further consistent improvement in classification performance was observed.

4.2. Development-set

Results on the development-set are presented in Table 1 and compared with results from the training-set when iteration was complete. We also tested whether the result obtained by each technique was significantly better than the baseline result using the statistical test described in [6] and the result of this test is given in the final column of Table 1. Table 1 shows a striking difference in performance on the training-set and test-sets. All the algorithms give very large reductions (72% for LDA+CT) in the error-rate on the training-set, but the best (relative) reduction in error-rate on the development-set was 14.4%, obtained by GPD, although three of the algorithms gave a performance gain that is statistically significant when compared with the baseline. This is an effect that might occur with small data-sets that are not-well matched in their content, but these two sets are large, randomly chosen from a single set, and the baseline classifier error-rate is approximately equal on both sets. The difference in performance is most likely to be due to a failure of the algorithms to generalise from training- to test-set.

Technique	% error on		Result on dev-set significant?
	Training-set (4642 calls)	Development-set (4713 calls)	
Baseline	*14.12*	*14.60*	
GPD	4.67	12.08	Yes ($p > 0.0001$)
CT	7.02	12.63	Yes ($p > 0.0001$)
LDA	5.30	14.15	No
LDA+CT	4.00	13.52	Yes ($p > 0.02$)

Table 1. Error rates on the training and development sets

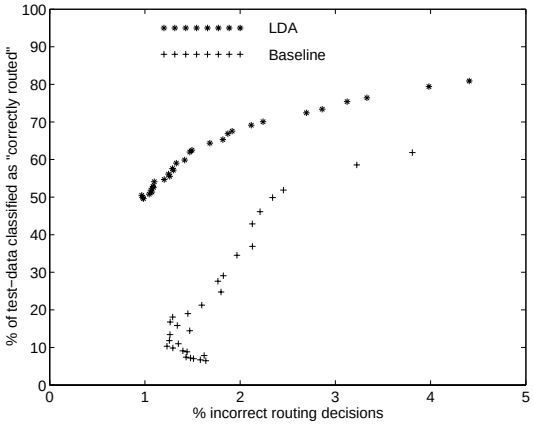

Fig. 2. % of data classified as "correctly routed" vs. routing error, using baseline and LDA confidence measures

4.3. Confidence Measures

Confidence measures [3] have found extensive use in speech recognition where they are useful for e.g. improving the efficiency of a speech dialogue/understanding system by requesting confirmation or re-input of uncertain words, for detection of out-of vocabulary words, to aid unsupervised speaker adaptation etc. They are also useful in call-routing systems, where they can be used to detect whether a call is likely to have been routed correctly or whether human intervention may be required to correct a possible error.

A simple confidence measure (CM) designed to predict whether the "top-choice" destination for the i'th call v_i is likely to be correct, is $C_i = S_i^1/S_i^2$, where S_i^1 is the recognition "score" of the top choice class and S_i^2 the score of the second choice class. When the dot-product is used, $C_i > 1$ and when the Euclidean distance is used, $C_i < 1$. In Figure 2, we compare the performance of a CM derived from the baseline classifier with a CM derived from the LDA classifier. A call is classified as "correctly routed" when $C_i < T$, where T is a variable threshold. The y-axis is the percentage of calls in the development set that were classed as "correctly-routed" using the CM and the x-axis is the percentage of these calls that were actually in error. The superiority of the LDA-derived CM over the baseline CM is clear: using the LDA CM, the routing classification error is only about 1% when 50% of the data is classified as "correctly routed". Put another way, when a call produces a value of $C_i < T$, which happens on about 50% of occasions, we can be 99% certain that it has been correctly routed.

5. DISCUSSION

A vector approach to call-routing has been shown to be an effective method by many researchers, and we have argued that a discriminative approach is attractive for this application. Three discriminative techniques have been investigated here. All the techniques produced very large reductions in the error-rate on the training-set but did not generalise well to the test-set—the best result (using GPD) produced an absolute error-rate reduction of 2.5% in a baseline error of 14.6%. However, a simple confidence measure designed to predict whether the call would be correctly routed was greatly improved by using a discriminative technique.

A considerable number of mis-classifications are due to inherent ambiguity in the query, which means it cannot be routed by an expert to a single destination. There are also some destinations that are so similar to each other that there is some doubt as to the consistency of the expert routing provided (e.g.*LostCard* and *ReplacementCard*). However, an examination of some of the errors reveals that there are linguistical reasons for many of them that are unlikely to be solved by improved generalisation of the current technique. Consider, for example the query "i'd like to purchase a refrigerator for six hundred dollars and i needed to know if i can put that on my card". The caller clearly requires to know his credit limit, but this is not explicitly stated and must be inferred. Or consider the query "i would like my account balance due on a different date". This is a request to change a date, and so should be routed to *Misc*, but was routed to *Balance* because of the presence of the words "account balance". Another frequent error occurs when the system does not appreciate that the utterance consists of multiple different queries and/or requests and simply assigns it to the single most likely destination. These examples suggest that a linguistic analysis of the utterance is essential. A preliminary step may be to classify queries into "straightforward" and "difficult" so that appropriate analysis may be given to more difficult ones. This would require a confidence measure that could detect ambiguity of the sort displayed in the queries above.

Acknowledgments

We thank Nuance Communications for providing the data for these experiments and Hong-Kwang Heff Kuo for advice on the GPD algorithm.

6. REFERENCES

[1] J.R. Bellegarda. A multispan language modeling framework for large vocabulary speech recognition. *IEEE Transactions on Speech and Audio Processing*, 6(5):456–467, September 1998.

[2] J Chu-Carroll and R Carpenter. Vector-based natural language call-routing. *Computational Linguistics*, 25(3):361–388, 1999.

[3] S.J. Cox and S Dasmahapatra. High-level approaches to confidence estimation in speech recognition. *IEEE Transactions on Speech and Audio Processing*, November 2002. (to appear).

[4] S.J. Cox and B. Shahshahani. A comparison of some different techniques for vector based call-routing. In *Proc. 7th European Conf. on Speech Communication and Technology*, September 2001.

[5] S.J. Cox and B. Shahshahani. Improved techniques for automatic call-routing. In *Institute of Acoustics Workshop on Innovation in Speech Processing (WISP-2001)*, April 2001.

[6] L Gillick and S.J. Cox. Some statistical issues in the comparison of speech recognition algorithms. In *Proc. IEEE Conf. on Acoustics, Speech and Signal-processing*, pages 532–535, April 1989.

[7] A.L. Gorin, G Riccardi, and J.H. Wright. How may I help you? *Speech Communication*, 23:113–127, 1997.

[8] H.K.J. Kuo and C Lee. Discriminative training in natural language call-routing. In *Proc. Int. Conf. on Spoken Language Processing*, October 2000.

[9] N.J. Nilsson. *Learning Machines: Foundations of Trainable Pattern-Classifying Systems*. McGraw Hill, 1965.

[10] A.R. Webb. *Statistical Pattern Recognition*. John Wiley & Sons Ltd., second edition, 2002.