A Comparison of some Different Techniques for Vector Based Call-Routing

Stephen Cox † *and Ben Shahshahani* ‡

† School of Information Systems, University of East Anglia, Norwich NR4 7TJ, U.K.
‡ Nuance Communications, 1005 Hamilton Court, Menlo Park, CA 94025, U.S.A.
sjc@sys.uea.ac.uk, ben@nuance.com

Abstract

Two approaches to vector-based call-routing are described, one based on matching queries to routes and the other on matching queries directly to stored queries. We argue that there are some problems with the former approach, both when used directly and when latent semantic analysis (LSA) is used to reduce the dimensionality of the vectors. However, the second approach imposes a higher computational load than the first and we have experimented with reducing the number of reference vectors (using the multi-edit and condense algorithm) and the dimensionality of the vectors (using linear discriminant analysis (LDA)). Results are presented for the task of routing queries on banking and financial services to one of thirty-two destinations. Best results (5.1% routing error) were obtained by first using LSA to smooth the query vectors followed by LDA to increase discrimination and reduce vector dimensionality.

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.

The major challenge in call routing is that the prompt to the customer is deliberately very general (e.g. "Please state your query or request", or "Please say which service you would like"). Hence, in contrast to the typical "Please say yes or no" prompts encountered in current voice dialogue systems, the 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 some alternative techniques for the vector-based approach to call routing. In this approach, a spoken query is viewed as a "vector" of words and vector pattern processing techniques are used to route the query to the correct destination. This approach is somewhat different from the statistical approach, in which the likelihood of the set of query words being associated with a particular route is estimated and statistical techniques used to decide the significance of this likelihood [6]. Chu Carroll and Carpenter have shown that the vector based technique offers superior performance on a call-routing problem with 23 destinations [2].

The paper is organised as follows: in section 2 we discuss the essential ideas behind vector-based call-routing and describe two variants of the technique experimented with here. Section 2.2 outlines how latent semantic analysis (LSA) has previously been used for information retrieval and section 2.3 gives some arguments for why a different approach to the use of LSA may be appropriate for call-routing. Section 3 describes the routing scenario used and the experiments performed, and includes a description of the application of linear discriminant analysis (LDA), which produced the most accurate routing. Finally, section 4 is a discussion of the ideas and results presented.

2. Vector techniques for call routing

The vector approach to call-routing is based on forming a matrix W using the transcriptions of the queries available to train the system. We assume that each of these has been labelled by an expert with the correct route. The rows of W correspond to different words (or sequences of words) in the vocabulary, and the columns to either different routes or different queries. To route a new query, it is first represented as an additional column vector of W and then matched to the other column vectors in W. Note that this approach ignores word order in queries.

Two different approaches to routing have been tested in this paper. In the first, which we term *T-ROUTE*, all training-data transcriptions associated with the same route are effectively pooled before being processed. In the second, termed *T-TRANS*, any duplicate utterances are discarded, but there is no pooling of utterances associated with the same route.

2.1. Overview of the *T-ROUTE* and *T-TRANS* training and testing procedure

Figure 1 shows the sequence of processes that were applied to the training data transcriptions prior to application of any further transformation, such as LSA, in the *T-ROUTE* approach (upper route) and the *T-TRANS* approach (lower route). In both approaches, the first steps are to identify commonly occurring phrases (collocations) and to remove any words on the "stop list"—these steps are described in more detail in section 3.2. An $M \times N$ term/document matrix W is then formed in which the words and collocations (hereafter called collectively "terms") are the rows, and the columns (called "documents" by association with information retrieval work) correspond to either indi-

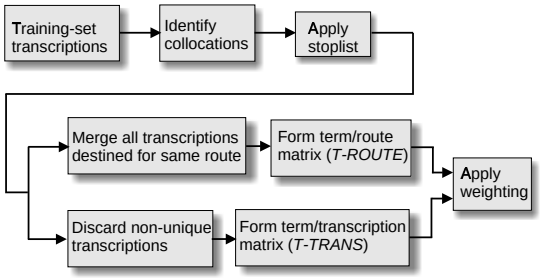

Figure 1: The T-ROUTE and T-TRANS approaches to call-routing

vidual routes (*T-ROUTE*) or indivdual documents (*T-TRANS*). Hence a row (term) vector in this space is of dimension N, and a column (document) vector of dimension M.

In the *T-ROUTE* approach, W is made by pooling the terms associated with queries destined for the same route, so that element W_{ij} of the matrix is the number of times that term t_i appeared in all the transcriptions associated with route r_j. The next step is to weight the terms in the matrix in a way that emphasizes words that are important for identifying a route or a transcription—this is described in section 3.3. At this stage, we have the option of applying further processing to the matrix in the form of LSA and/or LDA. These steps are described in sections 2.2 and 3.5. For classification, a query is pre-processed into a document vector by identifying the terms present in the query and effectively making an entry as column $N+1$ of the W matrix. Pre-processing is then applied to the query as described in section 3.3. If LSA or LDA have been used in the training process, the appropriate transformation is applied to the vector so that it can be matched in a subspace. The query is then classified by measuring the distance (in either the original or transformed space) of the query vector to each of the N document vectors, using an appropriate metric, and choosing the route associated with the "closest" vector.

In the *T-TRANS* approach, the W matrix is constructed by discarding duplicate transcriptions (i.e. utterances that consist of the same set of words, without regard to word order) and then assigning a column to each unique transcription, so that element W_{ij} is the number of times that term t_i appeared in (unique) transcription q_j. We also keep a record of which route each column is associated with. The terms are weighted (section 3.3) and LSA or LDA can then be applied to the W matrix if required. Classification of a query vector is by measuring the distance of the vector to each of the N document vectors, finding the "closest" vector and then looking up the route with which this it is associated.

A discussion of the issues in the *T-ROUTE* and *T-TRANS* representations of the data is given in section 2.3.

2.2. Latent semantic analysis (LSA)

Latent semantic analysis (LSA) has proved to be a successful technique for information retrieval [3]. LSA is based on using the technique of singular value decomposition (SVD) to find the lowest error representation of the matrix W in a compact subspace. This can be seen as "smoothing" the term or document vectors. It is not proposed to describe the theory of LSA in detail here—for an introduction, see [8]. However, the essential steps in the process are as follows:

1. SVD is applied to the term/document matrix W so that $W = USV^T$, where U and V are orthonormal matrices (dimensions $M \times N$ and $N \times N$ respectively) and S is an $N \times N$ diagonal matrix of eigenvalues.
2. S is inspected and the dimensions corresponding to only the top R eigenvalues are retained: the other dimensions are discarded. If $R \ll N$, this means that W is represented in a much reduced dimensionality.
3. A query is pre-processed into a vector $\mathbf{q}$ (as described in section 3.3) and is then projected into the reduced subspace to become the vector $\mathbf{q}'$, where $\mathbf{q}' = \mathbf{q}^T US^{-1}$.
4. $\mathbf{q}'$ can then be matched (using an appropriate criterion) in the subspace to the training-set vectors.

2.3. The use of LSA in call routing

The description of the operation of LSA given in section 2.2 is essentially the one given in the work of Landauer [8], which was concerned with learning synonyms using an encyclopaedia as training material. In this case, the number of words was over 60 000 and the number of documents over 30 000. The central tenet of Landauer's experiments is that there is some hidden connection between words and between documents (which is presumed to be governed by semantics), and that this can be discovered by transforming to a dimensionality much lower than the large dimensions of W. The authors found that performance peaked when $R \approx 300$, which might be thought of as the number of different "semantic units" in their encyclopaedia data.

However, this scenario is fundamentally different from the call routing scenario. In call routing, the number of "semantic units" is known *a priori* and, for our purposes, is simply equal to the number of routes. Also, the labelled training data tells us which words are associated with each route. Hence there is no requirement to use LSA for dimensionality reduction to discover groups of words that are associated with the same semantic unit, as in [8]. If LSA is used, it is for smoothing of the "noisy" query vectors . However, the maximum number of dimensions that can be used to represent a document vector after application of SVD to W is $\min(M, N)$. In call routing applications, N is generally very small in comparison to the number of terms, (e.g. $N = 32$ in our application, $N = 23$ in [2]), and so the document vectors are represented in a space that may be too low.

An alternative approach is not to pool the words associated with each route but to define the document vectors to be (unique) individual utterances, as described in the *T-TRANS* approach (section 2). Then N is equal to the number of unique documents (utterances) in the training corpus which will be substantially more than the number of routes. If it is desired to apply LSA to smooth the vectors, the reduced dimensionality will not be as low as that dictated by SVD when $N =$ the number of routes. Note that when this approach is used, it is necessary, for classification purposes, to keep a record of the route with which each unique utterance is associated.

Another justification for using the *T-TRANS* approach is apparent when one compares the form of the query vector and the form of the document vectors in the *T-ROUTE* approach. A typical query vector (before any pre-processing) consists of a column of mostly zeros with only a few integer entries, usually of value one—in our application, the average number of terms in a query was 2.89. By contrast, the "route" column vectors are the union of all vectors associated with a certain route and have many non-zero entries, some of which may be large integers—

for instance, the most frequently used route in our application had non-zero entries for 262 terms and a count of 463 for one term. So when the query vector is matched to these vectors, one is not matching like with like.

In addition, callers tend to use mainly use short stereotyped phrases to make queries: although the number of training-data queries available was 3300, after use of the stoplist, only 777 of these were found to be different document vectors. Although this is an order of magnitude more than the number of routes, it is not computationally unfeasible to match

3. Experimental Procedure and Results

3.1. Scenario

The system developed for these experiments was designed to route telephone queries relating to banking and financial services to one of thirty-two destinations. Training data consisted of about 3300 calls to a prototype system and testing data a further 2271 calls made at the same time and under the same conditions. These calls were transcribed and labelled by an expert with the appropriate route. Because we were concentrating on routing issues in these experiments, we used only the transcriptions of the calls rather than the output from the speech recogniser. Our own experiments have indicated that routing performance is degraded only slightly when the transcriptions are replaced with output from the speech recogniser.

3.2. Term extraction and stoplist definition

Certain phrases occur regularly in the transcriptions of the queries: examples from our application are "travel money", "change of address", "I would like to", "to speak to" etc. It would seem to be useful to include these collocations as terms as they may bear more information about the route than the same words in isolation. Any phrase in the training data that occurred fifteen or more times was added to the vocabulary as a term (there were approximately fifty such phrases). The mutual information (MI) between each term and the routes was then estimated, the terms were ranked by their MI, and the salient terms were identified as all terms whose MI was above a threshold T (T was determined experimentally). Terms whose MI was less than T formed the "stoplist" for experiments i.e. the set of words that were discarded from a transcription prior to processing it. Using collocations and a stoplist gave a small but consistent gain in performance in all cases.

3.3. Term weighting

The count W_{ij} of the number of times term t_i occurred when requesting route r_j (as in *T-ROUTE*) or in transcription j (as in *T-TRANS*) is not suitable for direct use in routing an input query. Various techniques for weighting the elements of W have been described. Most of these techniques replace W_{ij} by the product of two weightings: one that takes account of the large variation in the number of occurrences of each term by applying some form of compression or normalisation and another that accounts for the fact that terms that occur in only a few documents are more likely to be useful for routing purposes than terms that occur in many documents.

We experimented with the following weighting schemes:

1. Inverse document frequency (IDF) (as defined in [9])
2. The weighting described by Bellegarda in [1]
3. The weighting described by Sparck-Jones in [10])
4. The weighting described by Carpenter in [2])

Our conclusion over several experiments using different matching techniques in different vector spaces was that there was little to choose between the schemes but the Bellegarda scheme appeared to be the most consistent and so this scheme was adopted for the experiments reported here.

3.4. Use of the multi-edit and condense algorithm

When the *T-TRANS* approach is used, there are 777 unique document vectors and it is required to compare the query vector with each of them. Although this is reasonably fast on modern computers, it is still over twenty times slower than using the *T-ROUTE* approach. The *multi-edit and condense* algorithm [4] is a well-known way of reducing the size of the comparison set in k nearest-neighbour classification. This algorithm is applied to the training-set vectors in two distinct phases:

1. Multi-edit: The set is edited so that, after partitioning into subsets for training and testing, all vectors are correctly classified.
2. Condense: Vectors that are not useful for classification are discarded so that the size of the set of reference vectors is greatly reduced.

When used with a range of different vector dimensionalities, the application of multi-edit and condense reduced the reference set from 777 to 140–160 vectors. One nearest-neighbour classification was used throughout—no improvement was observed for $k > 1$.

3.5. Use of linear discriminant analysis (LDA)

A successful discriminative approach to call routing based on the minimum error classification criterion was reported in [7]. Linear Discriminant Analysis (LDA) [5] is a discriminative classification technique that is implemented by applying a linear transformation to the training and query vectors. LDA reduces the dimensionality of the vectors to $N-1$, where N is the number of classes. It has two attractive features when applied to call routing:

1. the number of classes equals the number of routes and this is usually small (< 40) in call routing; hence, after application of LDA, classification occurs in a low-dimensionality space;
2. if LSA is used in conjunction with LDA, the LDA transformation can be integrated with the LSA transform.

For a full description of LDA, see [5]. The required discriminative transformation matrix W that transforms a vector from the original space (dimension R) to dimension $N-1$,is the matrix of eigenvectors $\mathbf{w}_i$ that satisfy

$$S_b \mathbf{w}_i = \lambda S_w \mathbf{w}_i. \quad (1)$$

In equation 1, λ is the corresponding eigenvalue of $\mathbf{w}_i$, S_b is the "between class scatter matrix" for the classes (=routes) and S_w the "within class scatter matrix".
The original space could be the untransformed (pre-processed) word counts, or an LSA space. It was found that estimation of S_b and S_w was difficult in the untransformed space because of the sparsity of the entries. Therefore, the *T-TRANS* matrix was used, and the vectors were smoothed by applying LSA as detailed in section 3.6. Then S_b, S_w and hence w were calculated, and applied to the training-set document vectors. At recognition time, the LSA smoothing was applied to the query vector

$\mathbf{q}$, followed by the w transform to reduce the dimensionality of $\mathbf{q}'$ to $N-1$. The query was then classified using the Euclidean distance between $\mathbf{q}'$ and the document vectors.

3.6. Experiments

Two techniques for classification were implemented when W was configured as a *T-ROUTE* matrix:

1. Classification was done in the untransformed (but pre-processed) word count space (32 vectors of dimension 582) using a Euclidean distance metric (*R-UNTRANS*);
2. Classification was done in LSA space (32 vectors of dimension 32) using a cosine distance metric (*R-LSA*).

Four techniques for classification were implemented when W was configured as a *T-TRANS* matrix:

1. Classification was done in the untransformed (but pre-processed) word count space (777 vectors of dimension 582) using a Euclidean distance metric (*T-UNTRANS*);
2. Classification was done in LSA space (777 vectors of a variable number of dimensions) using a cosine distance metric (*T-LSA*);
3. Classification was done using a variable number of reference document vectors (approximately 140–160) selected by the *multi-edit and condense* algorithm (*T-MEDIT*, see section 3.4) in LSA space (variable number of dimensions), using a Euclidean distance metric;
4. Classification was done in LDA space (777 vectors of 31 dimensions) using a cosine distance metric, after application of LSA (variable number of dimensions) (*T-LDA*, see section 3.5).

3.7. Results

Figure 2 shows the results obtained from the six different schemes listed in section 3.6. The schemes that used the *T-ROUTE* approach (R-UNTRANS and R-LSA) were the worst performing, followed by the multi-edit and condense technique applied to the *T-TRANS* vectors (T-MEDIT). A problem with T-MEDIT is that in the *multiedit* stage of the process, in order to ensure that the reference set is classified completely correctly, vectors which may be useful for correct classification are discarded. It is interesting to note that using LSA on its own was worse in every case than using matching in the untransformed space. However, the best performance was 5.1% error using LSA followed by LDA (T-LDA), using 350 LSA dimensions to smooth the data and then reducing to a dimensionality of 31 using LDA.

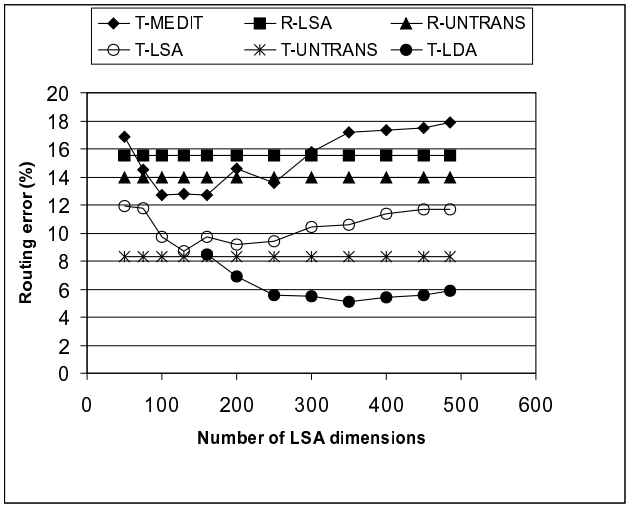

Figure 2: Error-rates for the six different schemes tested

4. Discussion

In this paper, we have described two techniques to vector-based call-routing. We have argued that there are some problems with the application of LSA to the "standard" call-routing scenario and our experimental results indicate that in this application, working in the untransformed term/document space is superior to using LSA alone. However, when LSA was combined with linear discriminant analysis (LDA), we obtained the best results, and we attribute these to the smoothing effect of LSA followed by the discriminative power of LDA. This technique also has the advantage that it reduces the data to a low dimensionality so that matching is relatively quick. We also experimented with four different term-weighting schemes and found little to choose between them. In the future, we plan to investigate ways of using recogniser transcriptions in the routing decision, and also how to couple the routing task more closely to the recognition.

Acknowledgment

This work was carried out whilst the first author was on study leave from the University of East Anglia at Nuance Communications. We are grateful to Benoit Dumoulin and other members of the dialogue research group at Nuance for their help.

5. 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. Deerwester et al. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41:391–407, 1990.

[4] P. Devijver and J. Kittler. *Pattern Recognition - a Statistical Approach*. Prentice-Hall International Inc., 1982.

[5] R.O. Duda and P.E. Hart. *Pattern Classification and Scene Analysis*. John Wiley and Sons, 1973.

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

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

[8] T.K. Landauer and S.T. Dumais. A solution to Plato's problem: representation of knowledge. *Psychological Review*, 104:211–240, 1997.

[9] C.D. Manning and H. Schutze. *Foundations of Statistical Natural Language Processing*. MIT Press, 1999.

[10] S.E. Robertson and K. Sparck Jones. Relevance weighting of search terms. *Journal of the American Society for Information Science*, 27:129–146, 1976.