Language-Independent Call Routing using the Large Margin Estimation Principle

Moataz El Ayadi\textsuperscript{1,2} and Mohamed Afify\textsuperscript{2}

\textsuperscript{1}Dept. of Eng. Math. & Physics, F. of Eng., Cairo University, Giza, Egypt
\textsuperscript{2}Orange Labs Cairo, Giza, Egypt
melayadi.ext@orange.com, mohamed.afify@orange.com

Abstract

There has been an increasing research interest in natural language call routing (NLCR) applications. One of the challenges often encountered in typical applications is the difficulty of performing language-dependent tasks such as morphological analysis of words and stop-word filtering. In this paper, we propose a novel NLCR system which does not depend on language-specific information and thus, it can be ported easily to many languages. The proposed system is based on a combination of character c-gram terms and discriminative training using large margin estimation principle. Compared to traditional vector-based NLCR methods, the proposed NLCR system does not need language-dependent processing and achieves around 1% increase in the classification accuracy.

Index Terms: natural language call routing, large margin estimation, discriminative training.

1. Introduction

Natural Language call routing (NLCR) consists of receiving a call from a customer and directing it to one among several departments without any human intervention. This finds numerous applications in call centers and customer services of banks and department stores \cite{1,2}. Compared to relying entirely on human operators, NLCR saves a lot of time and effort and can only pass the call to the human operator when automatic classification fails. The quantitative approach to NLCR typically works as follows. First, the transcriptions of all calls are obtained by an ASR module and are collected to form a document\textsuperscript{1}. Some text processing techniques are then applied to the sentences in the document in order to extract important keywords, called terms. The filtered transcriptions are then scanned in order to determine the list of relevant terms, called salient terms \cite{2}. In the training phase, a term-document matrix is constructed from the filtered text and the salient terms. The $ij$-entry of this matrix contains the frequency of occurrence of the $i$-th term in sentences belonging to the $j$-th destination. This term-document matrix is typically used in the classification.

The term-document matrix can be used directly in the classification with some properly defined distance measure in the so-called vector-based methods \cite{2} or alternatively the columns of the matrix can be used in conjunction with machine learning or statistical classification techniques to yield even better classification performance. In recent work \cite{3}, the idea of minimum classification error (MCE) learning was combined with vector-based call routing. This combination does not only improve the classification performance but it also makes call routing possible with less language-specific knowledge. The basic idea is to write a smooth (differentiable) expression of the classification error in terms of the so-called routing matrix. Probabilistic descent optimization methods are then used to minimize this expression with respect to the routing matrix parameters. On the same task as in \cite{2}, it was shown in \cite{3} that MCE leads to 10-30\% error rate reduction compared to the classical vector-based method. In \cite{4}, a case was made for using the support vector machine (SVM) for text categorization; experiments showed one of the best results on the Reuters task. We refer the reader to \cite{5} for an overview of machine learning techniques in text classification.

Large margin estimation (LME) has witnessed a lot of interest in the past several years as a promising discriminative training criterion. Inspired by the SVM, the basic idea of LME is to make the classification boundaries as far from all training examples as possible. This will increase the margin of each class and will improve the classifier generalization performance \cite{6}. For example, maximum margin training of continuous density hidden Markov models and Gaussian mixture models was used for automatic speech recognition (ASR) in \cite{7,8}. Both works show superior recognition performance compared to more traditional minimum classification error training. LME has not been used in the context of NLCR by the time of this writing to the best of our knowledge. Typically, NLCR (more generally text classification) is based on terms formed from words and word-grams (w-grams). While the systems generally use statistical or machine learning principles, there is considerable linguistic knowledge needed to define words, provide ignore-word and stop-word filtering and apply morphological analysis \cite{2}. Without these pre-processing steps, the performance of word-based systems can significantly degrade. Alternatively character and character-grams have been used for text classification in \cite{9}. It is generally agreed that the character approach is more flexible from a linguistic point of view but can lead to less accurate classification compared to the word approach. However, in \cite{3} it was shown that using discriminative training can reduce the need for using linguistic knowledge while maintaining the classification accuracy. The contributions of this paper are, thus, two-fold:

- Integrate LME-based training into NLCR schemes
- Experiment with discriminative training in conjunction with character-based terms to reduce language dependency. The motivation of the latter is to be able to quickly build NLCR systems for less-studied languages in Arabic or African languages.

This paper is organized as follows. In section 2, we present a short overview of vector-based NLCR since it is the building

\footnote{\textsuperscript{1}In this work we assume the transcription of all calls are given and hence no ASR is involved.}
block of our proposed system. In section 3, we describe how the LME is used to update the term-document matrix. Experimental results are demonstrated in section 4. Finally, conclusions are discussed in section 5.

2. An overview of vector-based natural language call routing

Vector-based methods originated in information retrieval. One of the earlier works that applied them to large scale call routing is [2]. These methods are based on defining a set of important and relevant words and word combinations, called salient terms. As their name indicates, vector-based methods transform each document or call transcription \( W \) into a vector \( x \) in which each dimension corresponds to one of the terms. The value of each component in \( x \) is equal to the frequency of occurrences of the corresponding term in the document \( W \). It was found that, in word-based systems, the extraction of salient terms is not trivial and highly depends on the language-specific information, e.g., grammar and structure rules. Therefore, it was proposed in [9,10] to replace the ordinary word-based terms by character-based terms, aka c-grams. In this modeling, terms are defined as frequent letter combinations rather than word combinations. Typically, the order of c-grams ranges from 2 to 10.

Since the number of possible c-grams is large, the resulting vector is usually in a very high dimensional space (e.g., several thousand dimensions) and dimensionality reduction techniques can be optionally applied. For more details about dimensionality reduction techniques, we refer the reader to [2].

In turn, each call class is represented by a vector \( r_i \). This vector is created from call transcripts in the training set labeled by its corresponding class. During classification, the class corresponding to an observed transcription \( W \) is determined as the one having the maximum similarity to its corresponding vector \( x \) as follows:

\[
k = \arg \max_{j=1,...,C} g_j(x) \tag{1}
\]

where \( g_j(x) \) is a similarity measure between \( x \) and the \( j \)th class. The overall training and testing algorithms of vector-based methods are described below:

2.1. Training phase

In the training phase, we have a set of \( n \) transcripts \( W_i, i = 1, \ldots, n \) classified into one of \( C \) classes and it is required to estimate the important parameters to be used later for classification. This is done as follows:

- For each document, extract all the \( w \)-grams (or c-grams) within the specified order range.
- Combine all the extracted \( w \)-grams (or c-grams) and count their frequencies. Eliminate infrequent \( w \)-grams (or c-grams); i.e., terms occurring less than a specified threshold.
- Form the term-document matrix \( R_{m \times C} \) where \( m \) is the number of terms. The element on the \( i \)th row and \( j \)th column of \( R \) is the number of occurrences of the \( i \)th term in documents belonging to the \( j \)th class. Elements of the matrix are usually weighted using inverse-document frequencies to deemphasize terms occurring in many documents.

2.2. Classification phase

The classification phase can be described by the following process:

- To classify a document \( W \), it is first converted into the corresponding vector as above. Let us call the resulting vector \( q_{m \times 1} \).
- Optionally reduce the dimensionality of \( q_{m \times 1} \) to \( r \) by pre-multiplying it by the matrix \( U_r^T \). Thus, we have the test vector \( x = U_r^T q \). As mentioned above this step is not used in the experiments.
- Determine the destination as the one having the maximum similarity to the obtained vector. Several similarity measures can be used here but the cosine measure \( g(x,y) = x^T y / (||x|| ||y||) \) is very popular in this context.

Based on the above presentation, it becomes clear that vector-based methods are very easy to train; just some count collection and matrix manipulations. Classification amounts to the evaluation of few distance measures and the obtained results are accurate in many situations. This makes vector-based methods a very popular choice for practical systems. However, in some cases, their classification performance is not satisfactory. Moreover, as will be shown in section 4, when c-grams are used instead of word grams, degradation in the classification accuracy occurs. Therefore, it has been thought of discriminative training techniques as a way of improving the classification performance by proper updating of the term-document matrix \( R \). Fortunately, it has been found that incorporating MCE and LME discriminative techniques to vector-based NLCR employing c-gram models compensates the degradation in performance resulted from using c-gram models. While the MCE method has already been covered in [3], we present our proposed LME update method in the next section.

3. Natural language call routing using large margin estimation

Basically, the main idea of discriminative training algorithms is to estimate the classifier parameters (the term-document matrix in this context) by optimizing a properly defined objective function that resembles the classification performance. Discriminative training techniques are usually based on the concept of a discriminant function which is a score function relating the input feature vector to each class. In this context, the discriminant function of the \( j \)th class, \( g_j(x) \), is the cosine similarity measure between the normalized input document terms vector and the normalized \( j \)-th column in the term document matrix; i.e.,

\[
g_j(x) = r_j^T x \quad j = 1, \ldots, C \tag{2}
\]

\(^1\)In the literature of NLCR, some researchers consider distance measures rather than similarity measures. In this case, a class is decided as the one having the minimum distance to the document vector. However, we prefer the notion of similarity measures because it is consistent with the following discussion of LME.

\(^2\)If \( R \) has the singular value decomposition \( R = USV^T \) where the diagonal entries of \( S \) are sorted in descending way, the matrix \( U_r \) is an \( m \times r \) containing the first \( r \) columns of \( U \), \( V_r \) is an \( n \times r \) matrix containing the first \( r \) columns of \( V \), and \( S_{r \times r} \) is an \( r \times r \) diagonal matrix containing to the first \( r \) rows and \( r \) columns of \( S \).

\(^3\)Although the resulting matrix size is large, it is typically sparse. Therefore, efficient techniques for performing SVD for large sparse matrices are usually employed. Moreover, in earlier versions of vector-based algorithms, SVD can be entirely eliminated.
where \( r_j \) is the \( j^{th} \)-column of the term document matrix \( \mathbf{R} \) and we assume that each of \( r_j \) and \( \mathbf{x} \) is normalized before the dot product operation.

The basic idea of LME is to place the decision boundaries as far as possible from the training examples. According to the theory of structural risk minimization, the Bayes error is bounded by the classification error in a finite training size plus a quantity related to the VC dimension of the classifier [11]. Therefore, maximizing the classifier should reduce the classifier generalization error [8]. Hence, for each training example, \( \mathbf{x}_i \), we seek to have

\[
\forall j \neq k_i \quad g_{k_i}(\mathbf{x}_i) - g_j(\mathbf{x}_i) \geq 1
\]

where \( k_i \) is the true class index of the training example \( \mathbf{x}_i \). Intuitively, this condition is equivalent to having the training example \( \mathbf{x}_i \) one unit apart from its closest decision boundary. Defining the margin of \( \mathbf{x}_i \) as \( g_{k_i}(\mathbf{x}_i) - g_j(\mathbf{x}_i) \) – 1, we can maximize the margin of the training examples by minimizing the following objective function

\[
\mathcal{L}(\mathbf{R}) = \sum_{i=1}^{n} \sum_{j=1}^{C} \left[ 1 + g_j(\mathbf{x}_i) - g_{k_i}(\mathbf{x}_i) \right]_+, \quad (4)
\]

where the function \([x]_+ = \max(0, x)\) is the hinge function. However, it can be noted that the term-document matrix \( \mathbf{R} \) can be scaled arbitrarily resulting in an unbounded objective function. Therefore, we have to add a regularization term constraining the \( \ell_2 \) norm of the columns of \( \mathbf{R} \). Moreover, in order to have a smooth differentiable margin function, we propose to approximate the hinge operator \([\cdot]_+\) by the following function

\[
h(y; \eta) = \frac{y}{1 + \exp(-\eta y)}
\]

where \( \eta \) is a smoothing parameter usually set manually. Thus, our final objective function to be minimized is

\[
\mathcal{L}(\mathbf{R}) = \sum_{i=1}^{n} \sum_{j=1}^{C} h(1 + g_j(\mathbf{x}_i) - g_{k_i}(\mathbf{x}_i); \eta) + \gamma \sum_{j=1}^{C} r_j^T r_j
\]

\[
= \sum_{i=1}^{n} \sum_{j=1}^{C} h(1 + (r_j - r_{k_i})^T \mathbf{x}_i; \eta) + \gamma \sum_{j=1}^{C} r_j^T r_j
\]

where \( \gamma \) is a regularization parameter usually set also by experiments. The objective function in (5) is optimized using the method of steepest descent. Initially, we set \( \mathbf{R} \) as the term-document matrix described in section 2. The matrix is iteratively updated according to the following rule

\[
\mathbf{R}^{(k+1)} = \mathbf{R}^{(k)} - \epsilon \frac{\partial \mathcal{L}(\mathbf{R})}{\partial \mathbf{R}} |_{\mathbf{R} = \mathbf{R}^{(k)}}, \quad (6)
\]

where \( \mathbf{R}^{(k)} \) is the term-document matrix at the \( k^{th} \) iteration and \( \epsilon \) is the step size usually determined by one-dimensional-search techniques so that \( \mathbf{R}^{(k+1)} \) attains the minimum of \( \mathcal{L}(\mathbf{R}) \) in the direction of the gradient at \( \mathbf{R}^{(k)} \). The above update rule should be reiterated until no significant decrease in the objective function is obtained or a maximum number of iterations is exceeded. The gradient \( \partial \mathcal{L}(\mathbf{R})/\partial \mathbf{R} \) is simply the horizontal concatenation of the derivatives of \( \mathcal{L}(\mathbf{R}) \) with respect to the columns of \( \mathbf{R} \), i.e.,

\[
\frac{\partial \mathcal{L}(\mathbf{R})}{\partial \mathbf{R}} = \left[ \frac{\partial \mathcal{L}(\mathbf{R})}{\partial r_1} \cdots \frac{\partial \mathcal{L}(\mathbf{R})}{\partial r_C} \right]
\]

4. Experimental results

In order to demonstrate the efficacy of our proposed NLCR algorithm, we employed a database of inquiries collected from subscribers calling the 1013 service provided by Orange in France. The used database contains 6000 sentences in French routed to 5 destinations, as shown in Table 1. Moreover, we translated the 1013 task to Arabic using a professional translation service. While the resulting Arabic task could be considered a bit artificial, it constitutes an interesting way to create voice services for Arabic or other languages in AMEA.

We applied the following NLCR methods:
- Vector-based method with cosine score (CS) [2]
- Vector-based method with logistic regression (LR) [2]
- Vector-based method with MCE [3]
- Vector-based method with LME training (our proposed method). We set \( \eta = 50 \) and \( \gamma = 1 \) after some experiments.

The terms used are:
- Word-based (w-gram) unigrams, bigrams, and trigrams.
- Character-based with matching word start and end (c-gram). The order of grams ranges from 2 to 10.

The experiments are run with 3-fold cross validation. In all the tested methods, terms occurring less than 3 times are ignored. It was found that the translated-to-Arabic corpus contains 856 w-grams and 5108 c-grams. For the French language, the number of c-grams (after removing infrequent terms) is 5294. We did not consider w-grams in French.

The summarized results for the translated-to-Arabic database are shown in Table 2 where all the above-mentioned classification techniques and term selection methods are considered. From the table, we get the following important observations:

<table>
<thead>
<tr>
<th>Class</th>
<th>Problem</th>
<th>No. of Sentences</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1</td>
<td>GARBAGE_</td>
<td>936</td>
</tr>
<tr>
<td>C2</td>
<td>Hearing difficulty</td>
<td>347</td>
</tr>
<tr>
<td>C3</td>
<td>Voice-box service</td>
<td>366</td>
</tr>
<tr>
<td>C4</td>
<td>Another request</td>
<td>639</td>
</tr>
<tr>
<td>C5</td>
<td>Line interruption</td>
<td>3692</td>
</tr>
</tbody>
</table>

Table 1: Description of the 1013 task.
The proposed classifiers, namely c-gram with MCE and LME perform generally better than the other two methods; they provide more than 90% classification accuracy. This demonstrates the usefulness of employing discriminative training techniques in NLCR.

Character c-gram with discriminative training performs as good as the word w-gram counterpart. Hence, it can be concluded that discriminative training methods compensate the degradation of performance due to the lack of linguistic information when c-gram models are used. Hence, in order to employ NLCR for rare languages, one may recognize the spoken phonemes by using an ASR system of another common languages such as English; the recognized phonemes with then serve as input for the c-grams.

The performances of MCE and LME are quite similar.

Generally, the values of individual fold accuracies for each classification method are quite close to each other. This indicates the consistency of performance of all the discussed methods. That is, the classification performance basically depends on the amount of training and testing data and not on the specific sentences used for training and testing.

The class precisions and recalls for each of the above-mentioned classification methods are presented in Tables 3 and 4, respectively. The precision of a certain class is defined as the number of sentences correctly classified to that class divided by the number of all sentences classified as belonging to that class. Meanwhile, the recall of a certain class is defined as the number of sentences correctly classified to that class divided by the number of all sentences which truly belong to that class. For those tables, we also consider the translated-to-Arabic 1013 database and c-gram models. Clearly, the MCE and LME-based algorithm provide higher precision and recalls than those provided by the two other methods. In particular, they significantly improved the classification of certain classes such as C1 and C2. Comparing MCE to LME, we see that, generally, MCE is better than LME in terms of class precisions while LME provides higher class recalls. Thus, it would be interesting to investigate the effect of aggregating the two methods in order to have an even better NLCR algorithm e.g. [12].

### Table 2: Comparison of the performance of different algorithms when applied to the translated-to-Arabic 1013 database.

<table>
<thead>
<tr>
<th>Classification method</th>
<th>Language model</th>
<th>Overall Accuracy</th>
<th>Fold accuracies</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS</td>
<td>w-gram 84.75%</td>
<td>84.70%, 85.20%,</td>
<td>84.35%</td>
</tr>
<tr>
<td></td>
<td>c-gram 81.30%</td>
<td>80.75%, 82.20%,</td>
<td>80.95%</td>
</tr>
<tr>
<td>LR</td>
<td>w-gram 86.20%</td>
<td>86.60%, 86.80%,</td>
<td>85.20%</td>
</tr>
<tr>
<td></td>
<td>c-gram 84.05%</td>
<td>83.60%, 84.25%,</td>
<td>84.30%</td>
</tr>
<tr>
<td>MCE</td>
<td>w-gram 89.20%</td>
<td>89.80%, 89.20%,</td>
<td>88.60%</td>
</tr>
<tr>
<td></td>
<td>c-gram 90.55%</td>
<td>90.75%, 90.95%,</td>
<td>89.95%</td>
</tr>
<tr>
<td>LME</td>
<td>w-gram 89.33%</td>
<td>89.90%, 89.45%,</td>
<td>88.65%</td>
</tr>
<tr>
<td></td>
<td>c-gram 90.47%</td>
<td>91.20%, 90.40%,</td>
<td>89.80%</td>
</tr>
</tbody>
</table>

### Table 3: Comparison of class precisions of different algorithms when applied to the translated-to-Arabic 1013 database. e-gram models are used.

<table>
<thead>
<tr>
<th>Classification method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS</td>
<td>C1</td>
</tr>
<tr>
<td>LR</td>
<td>C1</td>
</tr>
<tr>
<td>MCE</td>
<td>C1</td>
</tr>
<tr>
<td>LME</td>
<td>C1</td>
</tr>
</tbody>
</table>

### Table 4: Comparison of class recalls of different algorithms when applied to the translated-to-Arabic 1013 database. e-gram models are used.

<table>
<thead>
<tr>
<th>Classification method</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS</td>
<td>C1</td>
</tr>
<tr>
<td>LR</td>
<td>C1</td>
</tr>
<tr>
<td>MCE</td>
<td>C1</td>
</tr>
<tr>
<td>LME</td>
<td>C1</td>
</tr>
</tbody>
</table>

The second set of experiments is carried on the French database. The results shown in Table 5 are limited for e-gram terms. It is observed that both MCE and LME provide very good performance without any specific linguistic knowledge. In addition, when we compare the results in Table 2 to those in Table 5, it may be concluded that the translation may lead to a slight decrease (1%-2%) in the classification performance. This result is very encouraging if we want to port to a new language.

### Table 5: Comparison of the performance of different algorithms when applied to the French 1013 database. e-gram models are used.

<table>
<thead>
<tr>
<th>Classification method</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS</td>
<td>81.42% 80.90%, 81.90%, 81.45%</td>
</tr>
<tr>
<td>LR</td>
<td>83.33% 82.35%, 83.85%, 83.80%</td>
</tr>
<tr>
<td>MCE</td>
<td>91.68% 92.20%, 91.05%, 91.80%</td>
</tr>
<tr>
<td>LME</td>
<td>91.83% 92.85%, 90.55%, 92.10%</td>
</tr>
</tbody>
</table>

### 5. Conclusions

In this paper, we have proposed a novel NLCR procedure which is simple to implement and does not depend on language-specific structure and grammar rules. Thus, it avoids the difficulty often encountered in the morphological analysis of words and the fact that interpretation of each word heavily depends on its context. The proposed NLCR algorithm has been successfully applied to Arabic and French databases and shown to outperform other competing methods employing linguistic information.

### 6. Acknowledgment

We would like to thank Geraldine Damnati, Florence Duclaye, Roamin Laroche and Nicolas Voisine for providing the French database and for very useful and interesting discussions.

The proposed classifiers, namely c-gram with MCE and LME perform generally better than the other two methods; they provide more than 90% classification accuracy. This demonstrates the usefulness of employing discriminative training techniques in NLCR.

Character c-gram with discriminative training performs as good as the word w-gram counterpart. Hence, it can be concluded that discriminative training methods compensate the degradation of performance due to the lack of linguistic information when c-gram models are used. Hence, in order to employ NLCR for rare languages, one may recognize the spoken phonemes by using an ASR system of another common languages such as English; the recognized phonemes with then serve as input for the c-grams.

The performances of MCE and LME are quite similar.

Generally, the values of individual fold accuracies for each classification method are quite close to each other. This indicates the consistency of performance of all the discussed methods. That is, the classification performance basically depends on the amount of training and testing data and not on the specific sentences used for training and testing.

The class precisions and recalls for each of the above-mentioned classification methods are presented in Tables 3 and 4, respectively. The precision of a certain class is defined as the number of sentences correctly classified to that class divided by the number of all sentences classified as belonging to that class. Meanwhile, the recall of a certain class is defined as the number of sentences correctly classified to that class divided by the number of all sentences which truly belong to that class. For those tables, we also consider the translated-to-Arabic 1013 database and c-gram models. Clearly, the MCE and LME-based algorithm provide higher precision and recalls than those provided by the two other methods. In particular, they significantly improved the classification of certain classes such as C1 and C2. Comparing MCE to LME, we see that, generally, MCE is better than LME in terms of class precisions while LME provides higher class recalls. Thus, it would be interesting to investigate the effect of aggregating the two methods in order to have an even better NLCR algorithm e.g. [12].

The second set of experiments is carried on the French database. The results shown in Table 5 are limited for e-gram terms. It is observed that both MCE and LME provide very good performance without any specific linguistic knowledge. In addition, when we compare the results in Table 2 to those in Table 5, it may be concluded that the translation may lead to a slight decrease (1%-2%) in the classification performance. This result is very encouraging if we want to port to a new language.

### 5. Conclusions

In this paper, we have proposed a novel NLCR procedure which is simple to implement and does not depend on language-specific structure and grammar rules. Thus, it avoids the difficulty often encountered in the morphological analysis of words and the fact that interpretation of each word heavily depends on its context. The proposed NLCR algorithm has been successfully applied to Arabic and French databases and shown to outperform other competing methods employing linguistic information.

### 6. Acknowledgment

We would like to thank Geraldine Damnati, Florence Duclaye, Roamin Laroche and Nicolas Voisine for providing the French database and for very useful and interesting discussions.
7. References


