



Phoneme Classification and Lattice Rescoring Based on a k -NN Approach

Ladan Golipour and Douglas O’Shaughnessy

INRS-EMT, Montreal, Canada

golipour@emt.inrs.ca, dougo@emt.inrs.ca

Abstract

In this paper we propose a k -NN/SASH phoneme classification algorithm that competes favourably with state-of-the-art methods. We apply a similarity search algorithm (SASH) that has been used successfully for classification of high dimensional texts and images. Unlike other search algorithms, the computational time of SASH is not affected by the dimensionality of the data. Therefore, we generate fixed-length but high-dimensional feature vectors for phonemes using their underlying frames and those of their boundaries. The k -NN/SASH phoneme classifier is fast, efficient, and could achieve a classification rate of 79.2% for the TIMIT test database. Finally, we apply this algorithm to rescore phoneme lattices, generated by the GMM-HMM monophone recognizer for both context-independent and context-dependent tasks. In both cases, the k -NN/SASH classifier leads to improvements in the recognition rate.

Index Terms: phoneme classification, nonparametric density estimation, lattice rescoring

1. Introduction

Today’s Continuous Speech Recognition (CSR) systems are usually based on HMMs. However, limitations such as “first-order” Markov models in use or the assumption of independent and identically distributed frames, which is far from reality, lead to their serious dependency on contextual information and language models to gain satisfactory results. Since the problem of phoneme recognition is closely related to phoneme classification, different techniques have been proposed to generate models of speech units based on discriminative measures of performance in order to obtain a better phoneme classification rate. Researchers attempt to substitute GMMs, which are based on a maximum likelihood criterion with probability estimations that have local discriminative ability, such as Neural Networks. They also employ “discriminative training” procedures such as Maximum Mutual Information (MMI)-based methods. However, the improvements achieved with these methods come with costs. These methods were expensive and computationally complex to implement [9]; for example, the training of a nonlinear SVM for n datapoints has a computational time of $O(n^2)$. Other examples of nonparametric classification techniques that have been applied for phoneme classification are methods based on Hidden Conditional Random Fields

(HCRF) [3], Support Vector Machines (SVM) [1], and Kernel Logistic Regression (KLR) [7]. The success and feasibility of recent nonparametric techniques of phoneme classification directed us to employ a recently proposed similarity search algorithm (SASH) [5] to perform monophone classification. This search algorithm showed remarkable efficiency in terms of computational time in very high dimensions for the classification of text and image data. Representing data in high dimensional space has the potential to induce more discrimination between data classes. In addition, it has been shown that the acoustic-phonetic information in phoneme boundary points can help in discriminating between phoneme classes [10]. Therefore, we extend the window of feature computation for each phoneme beyond its boundaries. Finally, we employ a discriminant probability estimate called k -Nearest Neighbour (k -NN) due to its simplicity and well-known low theoretical classification error. The outline of this paper is as follows. Section 2 describes the approximate k -NN classification method and the SASH similarity search algorithm. In Section 4, we illustrate the proposed phoneme classification algorithm. We also combine this algorithm with a baseline GMM-HMM monophone recognizer through lattice rescoring. Finally, in Section 5, we derive some conclusions. All the experiments are performed on the clean TIMIT database, and the development set is directory 1 of the TIMIT test section.

2. k -Nearest Neighbour Classification Rule

The k -NN-based techniques are widely used in the field of pattern recognition. The well-known advantage of the k -NN classifier is its error rate which can be proved theoretically to be upper bounded by twice the Bayes error rate (optimal). The k -NN density estimate is fully described in [2]. Here, we only explain the volumetric k -NN classification rule for the Euclidean distance measure [13]:

$$\hat{c} = \arg \min_c \left\{ \frac{d}{2} \cdot \log \left((o - y_c^k(o))' (o - y_c^k(o)) \right) - \log \frac{k}{N_c} - \log P(c) \right\}. \quad (1)$$

In the above equation, $y_c^k(o)$ is the k^{th} nearest neighbour of the query point o from class c , N_c is the number of members of class c , d is the dimensionality of the feature space, and $P(c)$ is the probability of class c .

3. Nearest Neighbour Search Algorithm

The similarity search methods for the nearest neighbours can be either “exact” or “approximate”. The difference between the approximate methods and their exact counterparts is the involvement of an error bound that controls the trade-off between the computational time and accuracy of the query search. An exact similarity search needs to access an unacceptably high proportion of the datapoints, which results in the problem of the *curse of dimensionality* in high dimensional data space. In addition, for the real data the accuracy of an exact similarity search is not very different from the approximate one. We employ an approximate similarity search algorithm called Spatial Approximation Sample Hierarchy (SASH), which was recently proposed by Houle [5]. The time complexity of most of the similarity search algorithms is exponential in the number of dimensions. However, the computational time of SASH is not affected by the dimensionality of the data. The SASH search structure shows efficient performance in clustering datasets of 1 million vectors with dimensions of more than 1.1 million. This is a lot more than the ability of the spatial search methods due to the dependence of their computational time on the dimensionality of the data. Moreover, the SASH query search performs almost 100 times faster than the sequential search methods [5].

Computational Complexity of SASH

According to Houle, the time complexity for constructing the SASH structure is $(2p^2n \log_2 n)$ and for the k -NN query is $(\frac{k^{1+\frac{1}{\log_2 n}}}{k^{\frac{1}{\log_2 n}-1}} + 2p^3 \log_2 n)$. n is the number of data points, and each node can have edges directed to it from at most p parent nodes at the level above it and from it to at most c child nodes at the level below it. The most proper choices for these parameters are $c = 4p$ and $p = 4$. As can be seen, the dimension of the feature vector only affects the distance computation and has no direct influence on the computational cost. For a quick comparison, the computational time of constructing a kd-tree is $O(n \log_2 n)$, however its query search has a computational time of $O(n^{1-\frac{1}{d}} + k)$. When the dimensionality increases, these methods perform even worse than a brute-force search algorithm.

SASH Construction:

In order to construct the SASH from the reference samples, all the datapoints are ordered randomly in a hierarchy with the number of datapoints in each level being half of those of the bottom level. All the nodes of the second level are connected to the root node. From the third level on, the procedure below is performed:

For each node u in level i , we find its p “tentative parents” from level $i - 1$. Once for every node in level i the tentative parents are determined, we stand in level $i - 1$ and connect every node of level $i - 1$ to its c “distinctive children” from level i .

The *tentative parents* of a node in level i are the p closest nodes in level $i - 1$ that are distinctive children of level $i - 2$.

The *distinctive children* of a node u in level $i - 1$ are the c closest nodes in level i that have chosen u as their tentative parent.

SASH Query Search:

Once all the nodes are interrelated in the SASH structure, it can be used for the query search: The nearest neighbours of a query point are the k elements of the union of sets of nodes, $P_1 \cup P_2 \cup \dots \cup P_h$; P_1 is composed of only the root node and h is the final level. Each of these sets, P_i , is constructed from one level, i , of the constructed SASH as follows.

We stand in level i and find the distinctive children of level $i - 1$ among the nodes of level i . Then, P_i would be the k closest neighbours of the query point among these distinctive children.

4. Experiments

4.1. Feature Extraction

We use a fixed-length feature representation, but we alter from previous approaches by using a much higher dimension. We use a similar concatenated feature vector to [8]. However, we noticed that if we reduce the number of averaged frames, the discriminative information for the phoneme representations increases, and a better classification rate is achieved. Therefore, we divide each of the 3 sections in 2. We also extend the phoneme duration by 2 frames outside of the phoneme boundaries to include the discriminative information of the boundary frames. Since there are phonemes with duration of 20 ms, we use only 2 frames in order not to extend to the third phoneme. Finally, we add the duration of the phoneme as the last entry to its feature vector. The total number of features reaches 253.

4.2. Performing Phoneme Classification

After producing 253 dimensional feature vectors for the phonemes of training and test databases, we apply a data sharpening procedure similar to [13] to reduce the overlap between phoneme classes. Next, we build the SASH search structure from the training samples. During the test, the SASH query search method and the k -NN classification rule in equation 1 are used to classify an unknown test pattern. We apply the standard 39 phoneme clustering that was proposed by Lee [8]. Since the number of nearest neighbours in vote (k) has to be determined in advance, we evaluate different values of k using the development set. Figure 1 displays the Phoneme Error Rate (PER) versus k .

As can be seen, $k = 10$ is the optimum. We apply the algorithm on the TIMIT test database and the total classification rate is 79.2%. The result for the 3-best classification is also 92.4%. In order to show the effect of employing a higher dimensionality on the classification rate, we examine other division factors for the phoneme duration. The classification results of the development set for different dimensions are displayed in Table 1. As can be seen, when the dimensionality increases, the classification rate increases as well until

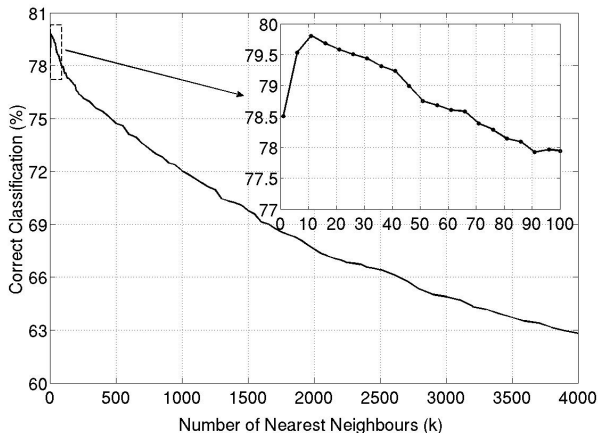


Figure 1: The PER (%) vs. “ k ”, the number of nearest neighbours in vote, for the development set.

Table 1: The classification rate of the proposed method for different dimensions on the development set.

Dimension	1×42	3×42	6×42	9×42
PER(%)	74.4	77.8	79.9	79.4

it reaches the maximum for the division factor of 6. The performance decreases slightly from the maximum for the division factor of 9 due to too much sparsity of the data space for an unchanged amount of training data. There have been a number of experiments on monophone classification using the clean TIMIT database in the literature. Here, we mention the classification results of some of the recent methods. The GMM baseline proposed by Halberstadt [4] produces a classification rate of 76.63% using heterogeneous measurements. In 1995, using support vector machines, Clarkson achieved a maximum classification rate of 77.6% for the polynomial degree 5 [1]. In 2005, Gunawardana applied a model called hidden conditional random fields for which the classification rate was 78.3% [3]. In 2007, Karsmakers’ method of fixed-size kernel logistic regression obtained 78.1% [7], and Rifkin et al’s linear Regularized Least Squares (RLS) method for second-order system resulted in 79.07% correct phoneme classification [11]. The proposed k -NN/SASH phoneme classification method incorporates a simple approximate classifier and a fast search structure, and it achieves a promising classification rate. The computational time and complexity of our phoneme classifier is mainly that of the SASH search algorithm, which is described in Section 3.

4.3. Confusion Matrix

A more detailed behaviour of a classifier can be observed from its confusion matrix. We compare the confusion matrix of the k -NN/SASH phoneme classifier and a baseline GMM-HMM phoneme classifier. The confusion matrices of the two systems are displayed in Figure 2. The level of darkness in these images shows the amount of confusion be-

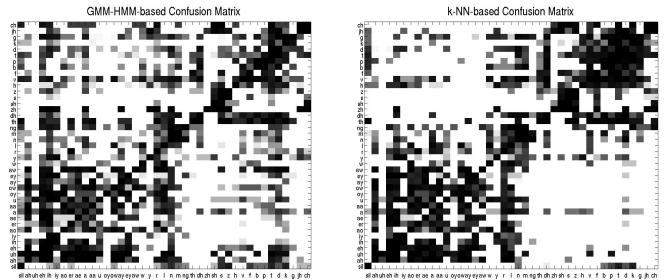


Figure 2: The confusion matrices for the GMM-HMM monophone classification (left) and the k -NN monophone classification (right).

tween different phonemes. According to the figure and as is already known in the literature, phonemes with similar general classes are more confused with each other due to their resembling characteristics. However, for classes which are phonetically distant from each other the k -NN method shows a better classification performance. This can be due to the fact that in the k -NN classification only those classes that are close neighbours of a particular class (and share main decision boundaries with it) are confused with that class. Therefore, the confusion is low between those classes that do not share decision boundaries. One weakness of the non-parametric models, such as k -NN, which we also observed during our experiments, is the problem of imbalance classes. The phoneme classes that have a high population compared to their neighbouring classes bias the recognition results towards themselves. This is an example of *lack of generalization* that exists for k -NN and some other nonparametric-based approaches. Different solutions have been proposed in the literature. Methods of condensing the training datasets tackle this problem by thinning the overpopulated classes and aiming to produce a dataset with more balanced classes, while preserving the boundary points [14].

4.4. Combining the k -NN classifier and the GMM-HMM phoneme recognizer

We incorporate the proposed k -NN classifier into a phoneme recognition task by rescoreing phoneme lattices, produced by a GMM-HMM monophone recognizer. According to our knowledge, the k -NN classification rule has not been tested for lattice rescoreing. We apply the HTK toolkit [6] to perform monophone recognition and produce phoneme lattices. We use 42 dimensional MFCC feature vectors, 3 HMM states, 2 null states, and 5 mixtures per state in the GMM-HMM structure of the HTK toolkit. The statistics of the lattices for both context-independent and context-dependent cases are displayed in Table 2. We classify the segments inside phoneme lattices using the k -NN/SASH method. Then, we calculate the new score based on the comparison between the k -NN/SASH classification result and the GMM-HMM one. If the k -NN phoneme choice is not similar to the GMM-HMM one, we add a cost to the GMM-HMM score,

Table 2: The statistics of lattices generated by the GMM-HMM monophone recognizer.

Averaged Statistics	CI	CD
tokens per state	2	2
arcs per second	85.6	166
max incoming arcs per node	4.7	7.5

Table 3: The recognition results before and after applying the k -NN lattice rescoring technique on the TIMIT test database.

Phoneme Lattices	Correctness(%)	Accuracy(%)
CI-unigram HMM	62.4	55.8
k -NN Rescoring	66.1	58.3
CI-bigram HMM	63.1	60.3
k -NN Rescoring	65.9	62.3
CD HMM	76.9	72.8
k -NN Rescoring	77.8	73.2

otherwise the new score is equal to the GMM-HMM score. The value of the cost needs to be proportional to the duration of the hypothesized phoneme. If not, the score of a long segment and a short segment in the lattice are modified equally while their GMM-HMM scores are influenced by their duration.

$$\text{new acoustic-score} = \begin{cases} \text{HMM acoustic-score} & \text{if } k\text{-NN phoneme} = \text{HMM phoneme} \\ \text{HMM acoustic-score} - \text{cost} \times (\text{phoneme duration}) & \text{Otherwise.} \end{cases} \quad (2)$$

Using the development set, the optimum value of $cost$ is 2 for both context-independent and context-dependent cases. As is shown in Table 3, upon using the k -NN/SASH classifier the accuracy of the stand-alone GMM-HMM monophone recognition improves for all cases. However, for context-dependent lattices, the improvement is small. This is due to the fact that the k -NN classifier uses only the acoustic information of separate monophones to perform classification, while the recognition results in context-dependent lattices are based on triphone information and more accurate. The insertion penalty, which is an ad-hoc parameter for the GMM-HMM recognizer of the HTK toolkit, is tuned to obtain the best possible result in each task. There have been other reports of lattice rescoring using nonparametric approaches which result better compared to the proposed method. Stuhlsatz [12] used support vector machines to perform lattice rescoring. He improved the performance of a baseline GMM-HMM system for the task of context-independent monophone recognition from 56.74% to 60.72% on the TIMIT database. However, he used lattices with 10 incoming edges per node.

5. Conclusion

In this paper, we introduced the k -NN/SASH phoneme classification technique that achieves comparable results with

current methods of phoneme classification. We employ a high-dimensional feature representation for phonemes, apply a fast approximate similarity search algorithm, and incorporate the k -nearest neighbour density estimate. Increasing the number of concatenated sections of feature vectors from 3 (previously practiced) to 6 let the classification rate increase from 77.8% to 79.9% on the development set. The proposed method achieves a classification performance of 79.2% for the TIMIT test database. We also applied our method to rescore phoneme lattices generated by a baseline GMM-HMM recognizer and obtained improvements for both context-independent and context-dependent phoneme recognition tasks.

6. References

- [1] Clarkson P., Moreno P. J., "On the use of support vector machines for phonetic classification", *Proc. ICASSP*, 1999, vol. 2, pp. 585-588.
- [2] Fukunaga K., "Introduction to statistical pattern recognition", *Morgan Kaufmann*, 1990, edition 2.
- [3] Gunawardana L., Mahajan A., Acero A., and Platt J. C., "Hidden conditional random fields for phone classification", *Proc. INTERSPEECH*, 2005, pp. 1117-1120.
- [4] Halberstadt A., and Glass J. R., "Heterogeneous measurements and multiple classifiers for speech recognition", *Proc. ICSLP*, 1998, pp. 995-998.
- [5] Houle M. E., and Sakuma J., "Fast approximate similarity search in extremely high-dimensional datasets", *Proc. 21st International Conference on Data Engineering (ICDE)*, 2005, pp. 619-630.
- [6] The Hidden Markov Model Toolkit (HTK), <http://htk.eng.cam.ac.uk>.
- [7] Karsmakers P., Pelckmans K., Suykens J. A. K., and Van hamme H., "Fixed-size logistic regression for phoneme classification", *Proc. INTERSPEECH*, 2007, pp. 1-4.
- [8] Lee S., and Glass J., "Real-time probabilistic segmentation for segment-based speech recognition", *Proc. ICSLP*, 1998, pp. 1641-1648.
- [9] Lefevre F., "Non-parametric probability estimation for HMM-based automatic speech recognition", *Computer, Speech and Language*, 2003, vol. 17, no. 2-3, pp. 113-136.
- [10] Marcus J., "Phonetic recognition in a segment-based HMM", *Proc. ICASSP*, 1993, pp. 479-482.
- [11] Rifkin R., and Bouvrie J., "Noise robust phonetic classification with linear regularized least squares and second-order features", *Proc. ICASSP*, 2007, pp. 881-884.
- [12] Stuhlsatz A., Meier H. G., Katz M., Krger S. E., and Wendenmuth A., "Support vector machines for postprocessing of speech recognition hypotheses", *Proc. International Conference on Telecommunications and Multimedia (TEMU)*, 2006.
- [13] De Wachter M., "Example based continuous speech recognition", *PhD thesis, K.U.Leuven, ESAT*, May 2007.
- [14] Wilson D. L., "Asymptotic properties of nearest neighbor rules using edited data", *IEEE Transactions on Systems, Man, and Cybernetics*, 1972, vol. 2, pp. 408-420.