Comparison of Distance Metrics for Phoneme Classification based on Deep Neural Network Features and Weighted k-NN Classifier

Muhammad Rizwan & David V. Anderson

Georgia Institute of Technology

mrizwan@gatech.edu, anderson@gatech.edu

Abstract

K-nearest neighbor (k-NN) classification is a powerful and simple method for classification. k-NN classifiers approximate a Bayesian classifier for a large number of data samples. The accuracy of k-NN classifier relies on the distance metric used for calculating nearest neighbor and features used for instances in training and testing data. In this paper we use deep neural networks (DNNs) as a feature extractor to learn discriminative internal structure of the data. We compared different distance metrics for calculating nearest neighbor in our speaker similarity score algorithm for phoneme classification and their execution time. The city block distance metric is computationally efficient and provides good phoneme classification accuracy.

Index Terms: Phoneme classification, feature learning, distance metrics

1. Introduction

The goal of a speech recognition system is to decode words for a given acoustic waveform. This is achieved by splitting the acoustic waveform into small fragments of speech known as "frames." Classifying these frames is known as phoneme classification. Thus, phoneme classification is the task of deciding the phonetic identity of a very short speech utterance (frame) [1]. The overall performance of a speech recognition system is highly dependent on the accuracy of framewise phoneme classification. Improving the phoneme classification accuracy will result in the overall improvement of speech recognition system [2]. The difference between phoneme classification and phoneme recognition is that in phoneme classification we classify phonemes on frame by frame basis while phoneme recognition consists of identifying individual phonemes in a sentence [3]. Dynamic programing techniques are used to convert the phoneme classification to the phoneme recognition.

Recently deep neural networks (DNNs) have become popular in various signal processing applications because of their strong ability to learn internal representation [4]. These DNNs cannot be directly applied to speech recognition systems as DNNs require fixed size input while various samples of phonemes/words are of variable length. In order to overcome dynamic nature of the speech, hidden Markov models (HMMs) are used. HMMs have strong sequential modeling capabilities. HMMs require observation probabilities for a given acoustic observation. These observation probabilities are estimated by DNNs by using a softmax layer at the output, where each neuron at the output layer corresponds to different phoneme class. These observation probabilities are converted to scaled likelihood by using Bayes theorem taking into account the prior probability of each phoneme class. HMMs use these scaled likelihood to find the most probable sequence of phonemes. By using lexicon and language models these phonemes are converted to word sequences [5].

Authors in [2] trained bi-directional Long Short Term Memory (LSTM) on the TIMIT database. They achieved a phoneme classification accuracy of 69.5%. Steve Renals, *et al.*, applied radial basis functions (RBF) for phoneme classification [6]. They varied the number of radial basis functions from 64 to 256 with input feature vector comprising three different sets: cepstral coefficients, formants tracks (with frequency, amplitude, and bandwidth information), and bark-scaled formants tracks. The best phoneme classification accuracy of 73.3% was obtained using the cepstral coefficients with 256 radial basis functions. The radial basis function–based phoneme classification system training is two to three times faster than neural networks trained with the back-propagation algorithm [6]. In [7], they used histograms reconstruction phase space (RPS) technique. They achieved an overall phoneme classification accuracy of 61.59%, 34.49%, and 30.21% for fricatives, vowels, and nasals respectively. The advantage of using the reconstruction phase space technique over cepstral coefficients is that it preserves non-linear information present in the speech utterance. In [8], they combined cepstral features with the RPS technique to achieve an overall classification accuracy of 57.85%. In [9], the author compared Gaussian mixture models (GMMs) and HMMs for phoneme classification. He showed that GMMs performed slightly better as compared with three–state HMMs. The most specific phoneme information is in the middle portion of the phoneme waveform. They achieved a phoneme classification accuracy of 60.43% with 256 Gaussian models. In [3], they used support vector machines (SVMs) and achieved a classification accuracy of 71.4%. Through experiments, they showed that better generalization performance is achieved by using Gaussian kernel as compared with linear or polynomial kernels. In [10], they learned distance metric for k-NN based phonetic frame classification and obtained an accuracy of 63.3%.

Current speech recognition systems are based on HMMs. HMMs have inherent deficiencies and weaknesses because of data generalization. In data generalization, original data is not retained, but rather approximated by statistical models. Various attempts have been made to overcome these problems by using all information from the data and building local models in original representation space [11, 12, 13, 14, 15]. This paper is an extension of our previous work on phoneme classification based on k-NN that utilizes all the information available with the speech utterance. In [16], we used k-NNs to learn a similarity score of a target speaker with speakers in our training set. We then used the speaker similarity score to weight k-NN for phoneme classification. In [17], we trained different DNNs architectures for feature learning and tested our speaker similarity score based approach on DNN features. In this paper we investigate various distance metrics and their computation time for our speaker similarity score based approach for phoneme classification. The outline of the paper is as follows. In Section 2, we discuss k-nearest neighbor and various distance metrics. Section 3 presents our speaker similarity score based phoneme classification method. In Section 4 we present the experimental results. Section 5, we discuss the conclusion of this paper and future work.

2. K-Nearest Neighbor

K-nearest neighbor (k-NN) is one of the most popular and simple supervised learning algorithms used in various pattern recognition and classification problems. The rationale behind k-NN is that the class of a query data instance (testing data) should be the same as the class of the nearest instance from the training data (instance space) or the majority of the nearest k instances. k-NN is non-parametric and does not require fitting a model [18]. Because of its non-parametric nature it does not generalize training samples with statistical models. This helps in doing fine comparison between training samples and incoming data samples and results in better classification performance [16]. Despite its simplicity, it works reasonably well with different types of data. The asymptotic error rate of k-NN approaches the optimal Bayes error rate, when the number of training samples N and the number of of neighbors k tend to infinity [19].

During the learning process, k-NN stores the entire collection of training samples along with class label information. In order to classify query data instance (testing data), the distance between query data and training samples is calculated based on some distance metric. Based on the value of "k", samples with least distances are selected, and the decision is made using some decision rule [20]. Thus, k-NN classification performance is dependent on:

- Input features
- Decision rule
- Distance metric
- Value of "k"

The k-NN has an inherent assumption that two instances that are close in space are likely to belong to the same class. But in reality this does not hold for most of the datasets [21]. The performance of k-NN can be improved by feature learning, clever decision rules, finding the appropriate distance metric, and experimenting with different values of "k". In our method (Section 3), we have used DNNs for feature learning and weighted our decision by learning speaker similarity score. We compared performance by varying value of "k" and distance metrics mentioned in Table 1 below, where $\mathbf{x}$ is the feature vector in $\mathbf{m}$ dimensional space, $\mathbf{s}$ represents query point, $\mathbf{t}$ represents point from the instance space, $\mathbf{\tilde{x}_s} = \frac{1}{m}\sum_{j=1}^{m}\mathbf{x_{sj}}$, and $\mathbf{\tilde{x}_t} = \frac{1}{m}\sum_{j=1}^{m}\mathbf{x_{tj}}$.

3. Method

Our speaker similarity score algorithm consists of three main steps. First, we train a DNN using a back-propagation algorithm to learn optimal DNN weights. After the DNN has been trained, we removed the output layer. The input vector we use consists of 13 Mel-frequency cepstral coefficients along with the first and second temporal differences (the deltas and delta-deltas) as input. The outputs of hidden layer are used as features. In the second step, we learned a speaker similarity score of the target speaker with the speakers in our instance space (training data).

Table 1: Distance Metrics for kNN

Distance	**Equation**
Euclidean	$\mathbf{d} = \sqrt{\sum_{j=1}^{n}(\mathbf{x_{sj}} - \mathbf{x_{tj}})^2}$
City Block	$\mathbf{d} = \sum_{j=1}^{n} \lvert \mathbf{x_{sj}} - \mathbf{x_{tj}} \rvert$
Chebyshev	$\mathbf{d} = \max_j \{ \lvert \mathbf{x_{sj}} - \mathbf{x_{tj}} \rvert \}$
Cosine	$\mathbf{d} = 1 - \frac{\sum_{j=1}^{n} \mathbf{x_{sj}x_{tj}}}{\sqrt{\sum_{j=1}^{n} \mathbf{x_{sj}x_{sj}}}\sqrt{\sum_{j=1}^{n} \mathbf{x_{tj}x_{tj}}}}$
Correlation	$\mathbf{d} = 1 - \frac{(\mathbf{x_s}-\mathbf{\tilde{x}_s})(\mathbf{x_t}-\mathbf{\tilde{x}_t})'}{\sqrt{(\mathbf{x_s}-\mathbf{\tilde{x}_s})(\mathbf{x_s}-\mathbf{\tilde{x}_s})'}\sqrt{(\mathbf{x_t}-\mathbf{\tilde{x}_t})(\mathbf{x_t}-\mathbf{\tilde{x}_t})'}}$

Finally, we weight our k-NN decision by the target speaker similarity score for phoneme classification decision. For details regarding the speaker similarity score algorithm, see [16, 17].

3.1. Feature Learning using DNN

By using a DNN for feature learning one can discover abstractions from low-level features to high-level concepts with very little human effort. These algorithms can learn non-linear mathematical models with multivariate statistics related to each other by intricate statistical relationship [4]. A typical architecture of deep neural network consists of an input layer, hidden layers, and output layer. The features learned using DNNs tend to eliminate irrelevant variabilities of raw input data and at the same time preserve information that is useful for classification. Deep neural network training is performed by maximizing the log posteriori probability over the training frames [22]. Maximizing the log posteriori probability is equivalent to minimizing the cross-entropy objective function given by Eq. 1.

$$E(w) = -\sum_{n=1}^{N}\sum_{k=1}^{K} T_{kn} \ln Y_k(X_n, w) \tag{1}$$

where Y_k is the network output, X_n is input to the network, w are the weights, T_{kn} is true value (one for correct class and zero for all others), K is the number of phoneme classes at output, N is the number of input samples. A sigmoid is used as a nonlinear activation function. a soft-max layer is used at the output for assigning probabilities to each phoneme class and is given by Eq. 2.

$$f_i(Z) = \frac{\exp(Z_i)}{\sum_{k=1}^{K}\exp(Z_k)}, i = 1, 2, .., K \tag{2}$$

Once the deep neural network is trained using back-propagation, we remove the soft-max layer. The data after the hidden layer activations are used as features.

3.2. Learning Speaker Similarity Score

We used a k-NN classifier to find the speaker similarity score for a target speaker with speakers in training data. The instance space of k-NN comprises phonemes from various sentences and speakers. For each instance in our training data we have phoneme label and speaker information available. We use some portion of the labeled data from the target speaker to find k-nearest phonemes from our instance space. For each correct match of the phoneme, we will find corresponding speakers for that correct match in our instance space. We then increment the score of that particular speaker by one. In this way we will use all the phonemes from the target speaker to find score of the speakers in our training data. The greater the score of a particular speaker in our training data, the more similar it is to the given target speaker.

3.3. Phoneme Classification

In our algorithm we use speaker similarity score information that we learned for our target speaker to weight our decision for phoneme classification. For a phoneme acoustic frame from the target speaker we found the k-nearest neighbors in the instance space using the distance metric. From these "k" nearest neighbor phonemes we will find the corresponding speakers. For each phoneme represented among the k-nearest neighbors, the corresponding speaker similarity scores are added. The classifier then assigns a label to the acoustic frame according to which of the represented phonemes has the highest score.

4. Results

4.1. Dataset

The dataset used in our experiment is TIMIT [23], which consists of 630 speakers from eight different dialect regions of United States. For each speaker we have ten utterances (2-SA, 5-SX and 3-SI). We divided the dataset into training, validation, and testing data. Training data consists of utterances from SX and SI. We used these utterances (SX and SI) to train DNN. Our instance space for k-NN also comprises utterances from SX and SI. Testing data consists of 24 speakers (3 speakers from each dialect region). We used random 4 utterances from each speaker to learn speaker similarity score and tested our approach on random 4 utterances. We used 39 phoneme classes.

4.2. Experiment

For feature learning we trained a DNN with two hidden layers. Hidden layer 1 has 1000 neurons. We varied the number of neurons in the second layer from 50 to 1000. DNN with 250 neurons in second hidden layer gives the best result. Results reported in this paper are based on DNN with 250 neurons in the second hidden layer. The speaker similarity score algorithm involves a training phase and a testing phase. In the training phase, speaker similarity scores are learned using k-NN for a target speaker and in the testing phase, based on the target speaker similarity score, phoneme classification is done using k-NNs with neighbors weighted according to their speaker similarity score.

4.3. Comparison of Distance Metrics

In this experiment we compared distance metrics mentioned in Section 2 for our speaker similarity score algorithm for different values of "k". Increasing the value of "k" improves the phoneme classification performance. The best performance is achieved by using Euclidean distance metric. The classification accuracy achieved by using city block distance metric is very close to Euclidean distance. Both Euclidean and city block distance metrics are a special cases of Minkowski distance metric. Minkowski distance between two points a and b in D dimensional space is given by:

$$d(a,b) = \sqrt[m]{\sum_{j=1}^{D} |a_j - b_j|} \tag{3}$$

where $m = 1$ and $m = 2$ for city block and Euclidean distance metrics respectively. As we increase m it gives more emphasis to large differences in individual dimensions. Fig. 1 shows the comparison of distance metrics.

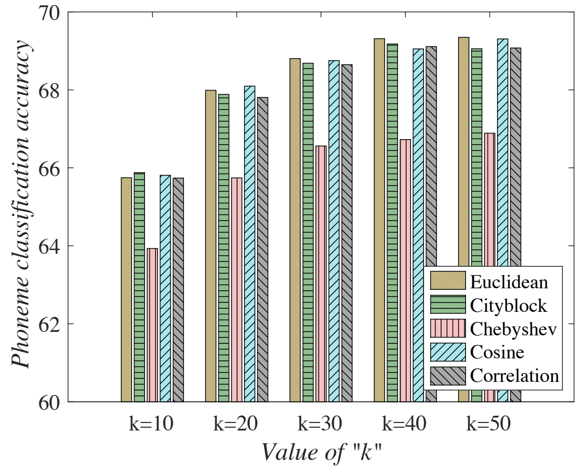

Figure 1: Comparison of distance metrics

4.4. Comparison of Execution Time

The distance metric has a significant impact on the execution time of our speaker similarity score algorithm. The city block distance metric requires less execution time for similarity score calculation and phoneme classification as compare with Euclidean distance metric. Fig.2 shows the speed up comparison of the city block, Chebyshev, cosine, and correlation with Euclidean distance metric.

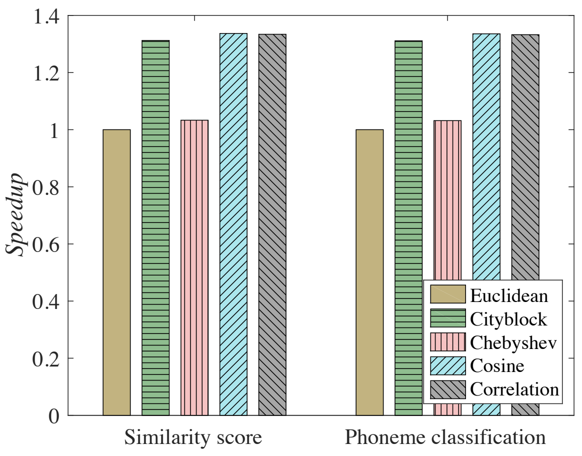

Figure 2: Comparison of execution time

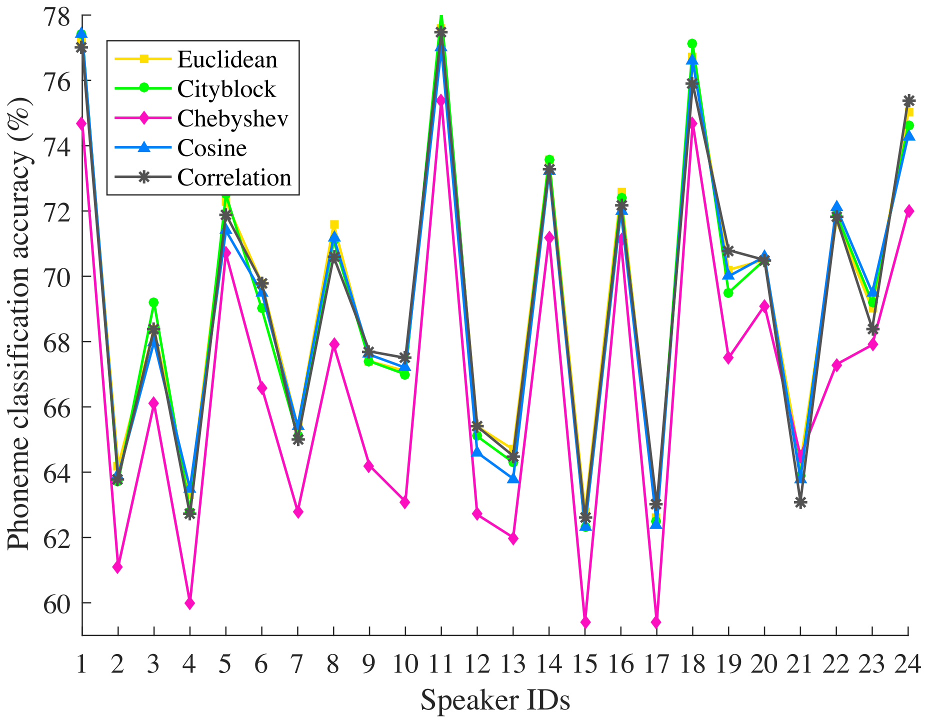

Figure 3: Variation in value of “k” per speaker

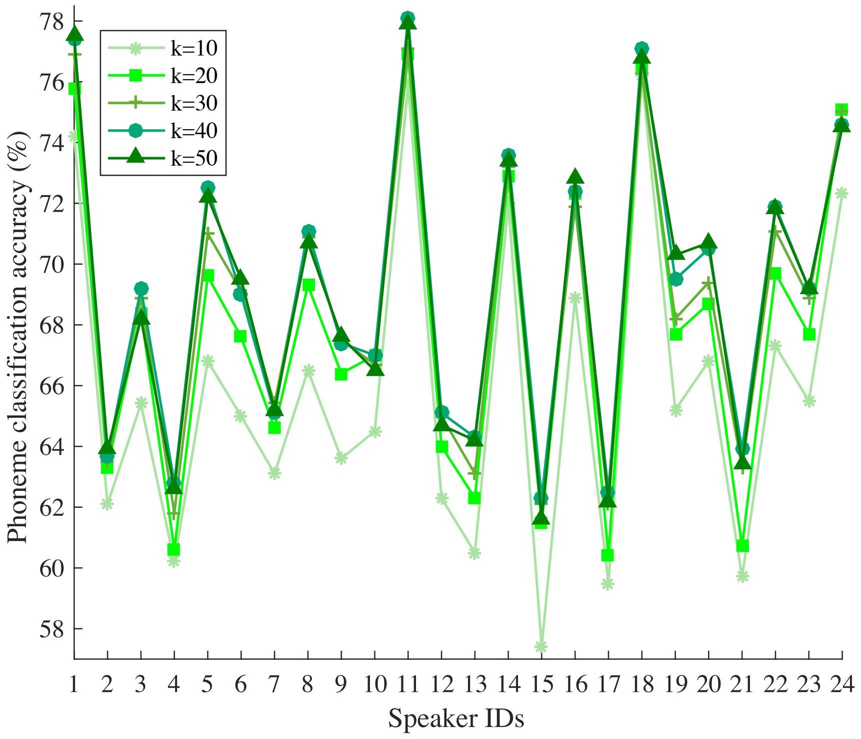

Figure 4: Comparison of distance metrics per speaker

4.5. Speaker wise comparison

In this experiment, we compared variation in value of “k” and distance metrics on individual speaker performance. Increase in value of “k” improves the phoneme classification performance, but as we increase “k” it requires more computation time. In terms of distance metric, the city block works well with most of the speakers. Figs. 3 and 4 shows the comparison of phoneme classification accuracy with distance metrics and value of “k” for 24 different speakers.

5. Conclusions and Future Work

In this paper we compared distance metrics for our speaker similarity score algorithm. Our speaker similarity algorithm is based on k-NNs and features learned through DNN. City block distance metric is computationally efficient and at the same time provides good phoneme classification accuracy. In future work, we are investigating on learning distance metrics based on the internal structure of the data and applying clustering methods to partition instance space into different groups when there is no speaker information available.

6. References

[1] O. Dekel, J. Keshet, and Y. Singer, “An online algorithm for hierarchical phoneme classification,” in *Machine Learning for Multimodal Interaction*. Springer, 2004, pp. 146–158.

[2] A. Graves and J. Schmidhuber, “Framewise phoneme classification with bidirectional LSTM networks,” in *Neural Networks, 2005. IJCNN., International Joint Conference on*, vol. 4. IEEE, 2005, pp. 2047–2052.

[3] J. Salomon, “Support vector machines for phoneme classification,” *Master of science, School of Artificial Intelligence, University of Edinburgh*, 2001.

[4] Y. Bengio, “Learning deep architectures for AI,” *Foundations and trends® in Machine Learning*, vol. 2, no. 1, pp. 1–127, 2009.

[5] H. A. Bourlard and N. Morgan, *Connectionist speech recognition: a hybrid approach*. Springer Science & Business Media, 2012, vol. 247.

[6] S. Renals and R. Rohwer, “Phoneme classification experiments using radial basis functions,” in *Neural Networks, 1989. IJCNN., International Joint Conference on*. IEEE, 1989, pp. 461–467.

[7] J. Ye, R. J. Povinelli, and M. T. Johnson, “Phoneme classification using naive Bayes classifier in reconstructed phase space,” in *Digital Signal Processing Workshop, 2002 and the 2nd Signal Processing Education Workshop. Proceedings of 2002 IEEE 10th*. IEEE, 2002, pp. 37–40.

[8] M. T. Johnson, R. J. Povinelli, A. C. Lindgren, J. Ye, X. Liu, and K. M. Indrebo, “Time-domain isolated phoneme classification using reconstructed phase spaces,” *Speech and Audio Processing, IEEE Transactions on*, vol. 13, no. 4, pp. 458–466, 2005.

[9] M. Antal, “Speaker independent phoneme classification in continuous speech,” *Studia Univ. Babes-Bolyai Informatica*, vol. 49, no. 2, pp. 55–64, 2004.

[10] J. Labiak and K. Livescu, “Nearest neighbors with learned distances for phonetic frame classification.” in *INTERSPEECH*, 2011, pp. 2337–2340.

[11] T. Deselaers, G. Heigold, and H. Ney, “Speech recognition with state-based nearest neighbour classifiers.” in *INTERSPEECH*. Citeseer, 2007, pp. 2093–2096.

[12] M. De Wachter, M. Matton, K. Demuynck, P. Wambacq, R. Cools, and D. Van Compernolle, “Template-based continuous speech recognition,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 4, pp. 1377–1390, 2007.

[13] N. Singh-Miller and M. Collins, “Learning label embeddings for nearest-neighbor multi-class classification with an application to speech recognition,” in *Advances in Neural Information Processing Systems*, 2009, pp. 1678–1686.

[14] L. Golipour and D. D. O’Shaughnessy, “Phoneme classification and lattice rescoring based on a k-nn approach.” in *INTERSPEECH*, 2010, pp. 1954–1957.

[15] T. N. Sainath, B. Ramabhadran, D. Nahamoo, D. Kanevsky, D. Van Compernolle, K. Demuynck, J. F. Gemmeke, J. R. Bellegarda, and S. Sundaram, “Exemplar-based processing for speech recognition: An overview,” *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 98–113, 2012.

[16] M. Rizwan and D. V. Anderson, “Using k-Nearest Neighbor and Speaker Ranking for Phoneme Prediction,” in *Machine Learning and Applications (ICMLA), 2014 13th International Conference on*. IEEE, 2014, pp. 383–387.

[17] ——, “Speaker adaptation using speaker similarity score on DNN features,” in *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 2015, pp. 877–882.

[18] R. Min, D. A. Stanley, Z. Yuan, A. Bonner, and Z. Zhang, “A deep non-linear feature mapping for large-margin k-NN classification,” in *Data Mining, 2009. ICDM’09. Ninth IEEE International Conference on*. IEEE, 2009, pp. 357–366.

[19] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern classification*. John Wiley & Sons, 2012.

[20] J. Gou, L. Du, Y. Zhang, T. Xiong *et al.*, “A new distance-weighted k-nearest neighbor classifier,” *J. Inf. Comput. Sci*, vol. 9, no. 6, pp. 1429–1436, 2012.

[21] W. Ren, Y. Yu, J. Zhang, and K. Huang, “Learning convolutional nonlinear features for k nearest neighbor image classification,” in *2014 22nd International Conference on Pattern Recognition (ICPR)*. IEEE, 2014, pp. 4358–4363.

[22] Z. Huang, J. Li, C. Weng, and C.-H. Lee, “Beyond cross-entropy: towards better frame-level objective functions for deep neural network training in automatic speech recognition.” in *INTERSPEECH*, 2014, pp. 1214–1218.

[23] J. S. Garofolo, L. F. Lamel, W. M. Fisher, J. G. Fiscus, and D. S. Pallett, “DARPA TIMIT acoustic-phonetic continous speech corpus CD-ROM. NIST speech disc 1-1.1,” *NASA STI/Recon technical report n*, vol. 93, 1993.