Utterance Verification using an Optimized k-Nearest Neighbour Classifier

R. Paredes, A. Sanchis, E. Vidal, A. Juan

DSIC/ITI, Universitat Politècnica de València
Camí de Vera s/n, E-46071 València (Spain)
`{rparedes,asanchis,evidal,ajuan}@iti.upv.es`

Abstract

Utterance verification can be seen as a conventional pattern classification problem in which a feature vector is obtained for each hypothesized word in order to classify it as either correct or incorrect. In this paper, we study the application to this problem of an optimized version of the *k-Nearest Neighbour* decision rule which also incorporates an adequate feature selection technique. Experiments are reported showing that it gives comparatively good results.

1. Introduction

Current speech recognition systems are not error-free and, in consequence, it is desirable for many applications to predict the reliability of each hypothesized word. From our point of view, this can be seen as a conventional pattern recognition problem in which a feature vector is obtained for each hypothesized word in order to classify it as either correct or incorrect [7]. The problem can then be properly approached using pattern classification techniques [1].

We have recently proposed a new feature, called *Word Trellis Stability* (WTS), that performs relatively well in comparison with several well-known features [8]. Also, we have recently developed a *smoothed naive Bayes* classification technique to profitably combine these features [9]. From an empirical viewpoint, this simplistic classification technique was good enough for us to show that certain (naive) feature combinations achieve better results than each feature alone [9]. However, as it has nothing to do with feature selection, it is only part of the solution. Unfortunately, the unsolved, feature selection problem is of great importance in utterance verification since many of the features proposed in the literature are notably redundant.

In this paper, we follow a holistic approach to the utterance verification problem; i.e., we consider a classification technique that also has a built-in, class-dependent feature selector. More precisely, the classification technique considered here is an optimized version of the *k-Nearest Neighbour (k-NN)* decision rule [1]. It is an optimized version in the sense that the standard Euclidean distance is extended to include a discriminatively-trained non-negative weight for each class-feature pair [5, 6]. As said above, it can be considered as an integrated approach to feature selection and classifier design; e.g., a null weight for a certain class-feature pair (c, d) means that feature d is not relevant for class c. Apart from this and, more importantly, our weighted k-NN approach has been successfully applied to many pattern recognition tasks [5, 6] and we show here that utterance verification is not an exception. More specifically, experiments are reported showing that it gives results that are similar to (or even better than) those of the naive Bayes model.

2. Predictor Features

In the utterance verification problem, a large number of predictor features have been proposed in the last years; however, it is unclear which of them are more informative.

Different kind of features can be obtained depending on the knowledge source. On the one hand, features can be derived directly from the speech recognizer. This kind of features can be classified into *acoustic* features, that are derived exclusively from acoustic information, *language model* features, that are based on language model information, and *combined* features, which try to benefit from both acoustic and language knowledge. The latter typically relay on the viterbi search decoding graph or on a compact representation of the best alternative hypotheses; e.g., word graphs or n-best lists. Many authors have proposed a large number of these types of features [2, 3, 4, 8, 10, 14]. On the other hand, there are a different kind of features that are based in more *heuristic* observations, e.g., the log of the number of times a word was observed in the training material, or the number of phones in a word [3]. In this work we have explored various kinds of features.

***Acoustic* features**

- *PercPh*: The percentage of hypothesized word phones that match the phones obtained in a "phone-only" decoding [3].

- *AbsPh*: The number of hypothesized word phones that match the phones obtained in a "phone-only" decoding.

- *NormACscore*: The acoustic log-score of the word divided by its number of phones [10].

- *ACscore*: The acoustic log-score of the word.

***Language model* features**

- *LMProb*: Language model probability [3].

***Combined* features**

- *Acoustic stability* (AS): Number of times that a hypothesized word appears at the same position (as computed by Levenshtein alignment) in K alternative outputs of the speech recognizer obtained using different values of the *Grammar Scale Factor* (GSF), i.e. a weighting between acoustic and language model scores [2].

- *Duration*: The word duration in frames divided by its number of phones [3].

- *Hypothesis density* (HD): The average number of the active hypotheses within the hypothesized word boundaries [4].

- *Word Activity* (WAc): The number of times per frame that the word is active in different partial hypotheses within its boundaries.

- *Word Trellis Stability* (WTS): We have recently introduced this feature. Let w be a word of the recognized sentence and let s_w, e_w be the starting and ending frames of w, $0 \le s_w < e_w < N$, where N is the number of frames of the given utterance. The WTS of w is computed as:

$$WTS(w) = \frac{1}{e_w - s_w + 1} \sum_{t'=s_w}^{e_w} \frac{C(w,t')}{\sum_{w'} C(w',t')}$$

$$C(w,t') = \sum_{t=t'}^{N-1} \sum_{h \in \mathcal{H}_t(w,t')} (\alpha_f - \alpha_i)$$

where $\mathcal{H}_t$ is a set of word-boundary partial hypotheses that are most probable at time t for a certain range of GSF values $[\alpha_i, \alpha_f]$. In addition, in each hypothesis of $\mathcal{H}_t(w,t')$ the word w must be active at time frame t'. More details about the WTS can be found in [8].

- WTS$_{max}$: A variant of the WTS feature, defined as:

$$WTS_{max}(w) = \max_{s_w \le t' \le e_w} \frac{C(w,t')}{\sum_{w'} C(w',t')}$$

Heuristic **features**

- *NumPh*: The number of phones found in the word's dictionary pronunciation [3].

3. Nearest Neighbour Approach

Let $P = \{(\boldsymbol{p}_1, c_1), \ldots, (\boldsymbol{p}_n, c_n)\}$ be a training data set (prototypes), where each variable pair represents a recognized word: $\boldsymbol{p}_i$ is a vector of features in a vector space E, and c_i denotes the true nature of the word ($c_i = 0$ for correct and $c_i = 1$ for incorrect); and let $d(\cdot, \cdot)$ be a dissimilarity measure defined in E.

Given a feature vector $\boldsymbol{x} \in E$ that represents a hypothesized word w, the *Nearest Neighbour* (NN) classification rule assigns the class of a prototype $p \in P$ to w such that $d(p, x)$ is minimum. The NN rule can be extended to the k-NN rule by classifying w in the class which is more heavily represented by the labels of its k nearest neighbours.

Unfortunately, in practice, the (k-)NN classification accuracy decreases dramatically with the number of available prototypes. To circumvent this problem, we use a weighted dissimilarity measure that helps improving the (k-)NN classification performance in small data set situations [5, 6]. The weights of this dissimilarity measure are learnt by a gradient descent algorithm [5] to optimize the error rate of the NN classifier. Moreover, the optimal weights values are easily profitable to select the most significant features, by discarding those that are not good predictor features.

3.1. Weighted dissimilarity measure

The proposed weighted distance can be seen as a generalization of the standard Euclidean distance. In its simplest form, the proposed weighted distance si:

$$d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_d \sigma_d^2 (x_d - y_d)^2} \qquad (1)$$

where σ_d is the weight associated with the d-th feature. Assuming a classification problem into different classes, a natural extension of (1) is:

$$d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_d \sigma_{cd}^2 (x_d - y_d)^2} \qquad (2)$$

where $c = class(\mathbf{x})$. We will refer to this extension as *"WL2 dissimilarity"*.

Let us now consider the problem of classifying new words output by the speech recognizer as either *correct* or *incorrect*. This is a classical two-category classification problem in which we are interested in finding an $2 \times D$ weight matrix

$$W = \left(\begin{array}{ccc} \sigma_{01} & \cdots & \sigma_{0D} \\ \sigma_{11} & \cdots & \sigma_{1D} \end{array} \right)$$

which optimizes the WL2-based NN classification performance.

3.2. Learning the optimal weights

In order to learn the weights of W, we consider the following criterion *index*

$$J(W) = \sum_{\mathbf{x} \in P} \frac{d(\mathbf{x}_{nn}^{=}, \mathbf{x})}{d(\mathbf{x}_{nn}^{\neq}, \mathbf{x})} \qquad (3)$$

where $\mathbf{x}_{nn}^{=}$ is the nearest neighbour of $\mathbf{x}$ in the same class and $\mathbf{x}_{nn}^{\neq}$ is the nearest neighbour of $\mathbf{x}$ in a different class. Note that this index is minimized when a matrix of weights W is chosen so that each prototype is close to prototypes from its own class and far away from prototypes that belong to other classes.

To search for a minimizer of (3), we use a gradient descent procedure:

$$\sigma_{cd}^{(k+1)} = \sigma_{cd}^{(k)} - \mu_{cd} \frac{\partial (J(W))}{\partial \sigma_{cd}^{(k)}} \qquad (4)$$

where $\sigma_{cd}^{(k)}$ denotes the value of σ_{cd} at iteration k of the descent algorithm and μ_{cd} is a step factor (or "learning rate") associated with feature d in class c (typically $\mu_{cd} = \mu$ for all c and d).

By developing the partial derivatives in (4) the following update equations are obtained:

$$\sigma_{cd}^{(k+1)} = \sigma_{cd}^{(k)} - \mu_{cd} \frac{\sigma_{cd}^{(k)} (x_{nn_d}^{=} - x_d)^2}{d(x, x_{nn}^{=}) d(x, x_{nn}^{\neq})} \qquad (5)$$

for each prototype from class c, and

$$\sigma_{cd}^{(k+1)} = \sigma_{cd}^{(k)} + \mu_{cd} \frac{d(x, x_{nn}^{=}) \sigma_{cd}^{(k)} (x_{nn_d}^{\neq} - x_d)^2}{d(x, x_{nn}^{\neq})^3} \qquad (6)$$

for each prototype from another class ($\neq c$) whose nearest neighbour in a different class is from class c. These equations are iteratively applied until no significant change in $J(W)$ is observed.

3.3. Feature Selection

Once the weights are estimated using the proposed gradient descent approach, a feature pruning can be performed by discarding features with low average weights:

$$\bar{\sigma}_d = \frac{\sigma_{0d} + \sigma_{1d}}{2} \qquad (7)$$

Then, we consider the features in order from lowest to highest average weights and each feature is selected or discarded depending on whether it improves or not a Leaving One Out error estimation. The final feature subset corresponds to the subset with the lowest Leaving One Out error.

Once the feature subset is obtained, the weights are re-trained, as in 3.2, using only this feature subset.

4. Experimental results

4.1. Experimental setup

We carried out experiments using the *FUB task*, an Italian speech corpus of phone calls to the front desk of a hotel, acquired in the context of the EUTRANS project [12]. The *FUB* corpus involves highly spontaneous speech data and contains many non-speech artifacts. Basic statistics of the (disjoint) training and test sets are summarized in table 1.

Table 1: FUB speech corpus

	training	test
speakers	276	24
running words	52, 511	5, 381
vocabulary size	2, 459	–
bigram perplexity	–	31

The training set was used to train Italian context-dependent phone models. The acoustic models were left-to-right continuous density HMMs, trained using Linear Discriminant Analysis (LDA) and a Viterbi approximation [13]. Decision-tree clustered generalized triphones (CART with $1,500$ tied states plus silence) were used as phone-units. A smoothed trigram language model was estimated using the transcriptions of the training utterances. The test-set Word Error Rate was 27.5%.

4.2. Experimental results

To perform the experimental study, a conventional continuous speech recognizer based on Viterbi beam search has been used with the language and acoustic models described in the previous section.

Once a word has been classified as either correct or incorrect, two different types of errors can occur. The first is produced when a correct word is classified as incorrect (*false rejection*) and the second is when an incorrect word is classified as correct (*false acceptance*).

We use the classification error rate (CER) as the metric for the evaluation of the classification accuracy. The CER is simply defined as the number of the two-type classification errors divided by the total number of recognized words. It should be noted that a baseline CER is obtained assuming that all recognized words are tagged as correct. This is equivalent to the number of insertions and substitutions, divided by the number of recognized words.

A first set of experiments has been performed using the k-NN rule with $k = 1$. Both the Euclidean and the WL2 distances, with the whole set of predictor features were used. The WL2 distance improve the classification error from the 21.3% of the Euclidean distance, to 20.2%, as shown in table 3. The weights of the WL2 distance were learnt following the approach presented in section 3. Table 2 shows the average weights estimated for each feature. It can be seen that AS appears as the most important feature, which confirms many previous observations [2, 3, 4, 8, 9].

Table 2: *Estimated weights for each feature. The features above the line were selected by the feature selection process.*

Feature	Weight
AS	1.115
NormACscore	1.010
LMProb	1.001
Duration	1.0
WTS	1.0
WTS_{max}	1.0
PercPh	0.999
AbsPh	0.995
WAc	0.988
NumPh	0.977
ACscore	0.976
HD	0.842

The next step was to select the most significant features following the procedure explained in section 3.3. A subset with the seven first best features (those above the line in table 2) was finally chosen. This selection suggests that our WTS and WTS_{max} features can be useful to improve the classification accuracy. With this feature subset and the weights associated to these selected features, we performed a 1-NN classification using the WL2 distance which reduced again the error rate down to 18.8% (table 3).

At this point the weights of the WL2 distance are re-trained using only the selected feature subset. Using the 1-NN rule and the WL2 distance with the new weights reduced the error rate down to 16.8% (table 3).

Finally, the k-NN classification rule with $k > 1$ was used. The value of k was estimated by leaving one out over the training data using the WL2 distance and the re-trained weights. With the best estimated value ($k = 30$) the test set error rate was further reduced down to 15.7%.

Table 3: CER for k-NN techniques.

Technique	CER (%)
Baseline	21.0
k-NN with $k = 1$	21.3
+WL2	20.2
+feature selection	18.8
+weights re-trained	16.8
+$k > 1$	15.7

It should be emphasized that feature selection and weight retraining have had the highest impact in improving the results. The WL2 improves the results significantly only after the feature set is reduced to the most important features. But, of course, the first weight estimation over the whole set of features was necessary for the feature selection procedure.

4.3. Comparative Results

The proposed technique can be compared with the Naive Bayes approach reported in [9]. Being x a feature vector to represent a hypothesized word, the Naive Bayes approach was used to estimate the posterior probability $P(c_i|x)$. To compare this method
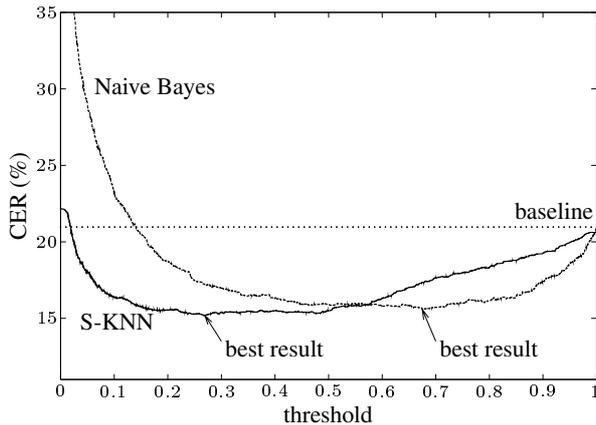
Figure 1: *Classification Error Rates as a function of posterior probability threshold.*

with the k-NN WL2 approach presented here, a smoothed estimation of the k-NN class posterior probability (S-KNN) of the word is used:

$$P(c_i|\boldsymbol{x}) = \frac{\sum\limits_{i \in S_c} \frac{1}{d(\boldsymbol{x}, \boldsymbol{y_i})}}{\sum\limits_{i=1}^{k} \frac{1}{d(\boldsymbol{x}, \boldsymbol{y_i})}}$$

where S_i is the set of indices of the prototypes from class c_i among the k nearest neighbours retrieved $y_1, \ldots, y_k$.

Once these posterior probabilities are computed, the word can be classified as either correct or incorrect, depending on whether its probability exceeds a certain threshold τ or not. CER values obtained for both techniques as a function of τ are shown in the figure 1.

With the class posterior probabilities estimated by Naive Bayes a minimum CER of 15.6% is achieved while, using the smoothed k-NN posterior probabilities, the minimum CER is 15.1%.

5. Conclusions

We have studied the application of an optimized version of the k-Nearest Neighbour decision rule to utterance verification. In contrast to other, simpler approaches to utterance verification, it provides an integrated solution to both feature selection and classifier design. The experimental results show that this new approach can achieve results similar to (or even better than) those of a smoothed naive Bayes classifier.

6. Acknowledgments

7. References

[1] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis*. John Wiley and Sons, New York, 1974.

[2] T. Zepenfeld, M. Finke, K. Ries, M. Westphal, and A. Waibel, "Recognition of conversational telephone speech using the JANUS speech engine," in *ICASSP*, 1997, pp. 1815–1818.

[3] L. Chase, *Error-responsive feedback mechanisms for speech recognizers*, Ph.D. thesis, School of Computer Science, Carnegie Mellon University, USA, 1997.

[4] T. Kemp and T. Schaaf, "Estimating confidence using word lattices," in *EUROSPEECH*, 1997, pp. 827–830.

[5] R. Paredes and E. Vidal, "A Nearest Neighbor Weighted Measure In Classification Problems," in *VII SNRFAI*, 1999.

[6] R. Paredes and E. Vidal, "A class-dependent weighted dissimilarity measure for nearest neighbor classification problems," in *VII SNRFAI*, 1999. *Pattern Recognition Letters*, vol. 21, pp. 1027–1036, 2000.

[7] A. Sanchis, V. Jiménez, and E. Vidal, "Efficient Use of the Grammar Scale Factor to Classify Incorrect Words in Speech Recognition Verification," in *ICPR*, 2000, vol. 3, pp. 278–281.

[8] A. Sanchis, A. Juan, and E. Vidal, "Estimating confidence measures for speech recognition verification using a smoothed naive bayes model," in IbPRIA'2003. (accepted)

[9] A. Sanchis, A. Juan, and E. Vidal, "Improving Utterance Verification using a Smoothed Naive Bayes Model" in *ICASSP*, 2003. (accepted)

[10] T. Schaaf and T. Kemp, "Confidence measures for spontaneous speech recognition," in *ICASSP*, 1997, pp. 875–878.

[11] J.C. Amengual, J.M. Benedí, F. Casacuberta, M.A. Castaño, A. Castellanos, V.M. Jiménez, D. Llorens, A. Marzal, M. Pastor, F. Prat, E. Vidal, and J.M. Vilar, "The EuTrans-I speech translation system," *Machine Translation*, vol. 15, pp. 75–103, 2000.

[12] Instituto Tecnológico de Informática, Fondazione Ugo Bordoni, RWTH Aachen, and ZERES Gmbh, "Final report," 2000.

[13] H. Ney, L. Welling, S. Ortmanns, K. Beulen, and F. Wessel, "The RWTH large vocabulary continuous speech recognition system," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 1998, pp. 853–856.

[14] D. Vergyri, "Use of word level side information to improve speech recognition," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2000, pp. 1823–1826.