Vector Quantisation Mappings for Speaker Verification

Anthony Brew, Pádraig Cunningham
University College Dublin
anthony.brew@ucd.ie, padraig.cunningham@ucd.ie

Abstract

In speaker verification several techniques have emerged to map variable length utterances into a fixed dimensional space for classification. One popular approach uses Maximum A-Posteriori (MAP) adaptation of a Gaussian Mixture Model (GMM) to create a supervector. This paper investigates using Vector Quantisation (VQ) as the global model to provide a similar mapping. This less computationally complex mapping gives comparable results to its GMM counterpart while also providing the ability for an efficient iterative update enabling media files to be scanned with a fixed length window.

1. Introduction

In most speaker verification systems, some form of spectral-based parametrisation is used to encode speech into machine readable form. Typically short-term analysis is used to compute a sequence of magnitude spectra or frames [1]. This creates two challenges for a classifier. First for each utterance there are a variable number of frames available for classification. Second the learner must somehow make a combined decision on the set of frames provided to it. For this reason generative classifiers have become popular in speaker verification as they provide a natural framework for combining the likelihood of multiple frames.

By far the most successful generative classifier applied to speaker verification has been the Gaussian Mixture Model (GMM). This is in part due to the potential for GMMs to model the underlying phonetic components that make up the human voice [8]. However as speaker verification is a discriminative problem it would seem natural to employ a discriminative classifier such as a Support Vector Machine (SVM).

Three main strategies for discriminative classification in this task have become popular. Firstly one can train the discriminative classifier directly on the cepstral frames and fuse individual frame scores to come to an overall decision. Secondly one can build a generative distribution for each utterance and compare this to other distributions built on the training set. Lastly models built on the training set can be used to project utterances into a fixed dimensional space for classification. The last two approaches have generally relied on GMMs.

A method of particular interest for this paper is a projection based on GMM MAP adaptation that spans the last two categories. For each utterance a global GMM is updated and the means of the individual mixtures are weighted and stacked to create a fixed-dimensional GMM super-vector (see section 3 for details).

Work by [5] has shown an adaptation-style technique for Vector Quantisation (VQ), an early method used in speaker verification [9, 11]. Inspired by this, this paper investigates how to use a VQ codebook to create a fixed dimensional projection for classification. This method gives comparable results to MAP adapted supervectors while also providing a simple and fast implementation that can be updated efficiently in an online manner by passing a sliding window over the length of a media file.

This paper starts by introducing the technical aspects associated with classification in this domain and how at a high level the sequence kernels presented operate. The paper then proceeds with a background of how MAP adaptation works with GMMs, it then provides an overview of VQ and the proposed projection based on an update to the VQ codebook. A comparison of results on two well-known datasets (YOHO and King) is then given. YOHO provides a dataset with short fixed vocabulary utterances while King provides a dataset with free vocabulary conversational speech.

2. Classification Overview

Speech utterances can be represented for classification in various forms with different levels of compactness as shown in Figure 1 [6]. The data rate of the raw acoustic speech signal is extremely high and so
speaker verification systems rely on frequency analysis of a sequence of windowed portions of the signal to provide information about the spectral content of the waveform at a much lower data rate. This sequence of cepstral feature vectors are usually computed via a mel-frequency filterbank, leading to mel-frequency cepstral coefficients (MFCCs), or via linear prediction, yielding the linear predictive cepstral coefficients (LPCCs) [1]. These cepstral coefficients are often augmented with the Δ cepstral features, which are the first order difference between the current frame and the surrounding frames to capture temporal information.

The above process produces a sequence of cepstral frames to provide to the classifier for prediction. In this paper we refer to this sequence of cepstral frames as an utterance. Traditional approaches employ generative classifiers such as Gaussian mixture models to build a generative model for the target class and the non-target classes. The predicted class for an utterance is provided by the generative model from which the sequence scored highest. While this approach is suitable for generative models, the combination of classification scores from individual cepstral frames by discriminative classifiers such support vector machines has had limited success [10].

In order to classify multiple cepstral frames via a discriminative classifier, several techniques combine the cepstral frames via a set of basis functions into a fixed dimensional characterisation vector for the utterance as a whole. The two approaches in this paper use a large prebuilt clustering of training frames to build a characterisation vector from the cepstral feature vectors. The main difference in the two approaches is that the traditional approach uses a soft partitioning provided by a Gaussian Mixture Model, while the proposed alternative uses a hard partitioning provided by Vector Quantisation.

3. GMM Supervectors

A GMM Universal Background Model (UBM) is a generative mixture model built on a large amount of speech from multiple speakers made up of \( N \) Gaussian components,

\[
g(x) = \sum_{i=1}^{N} \lambda_i N(x, \mu_i, \Sigma_i)
\]

where \( \lambda_i \) are the mixture weights, \( N(\cdot) \) is a Gaussian with \( \mu_i \) and \( \Sigma_i \) as the mean and covariance matrices. In speaker verification it is common that the covariance matrices are restricted to being diagonal. This constraint of diagonal covariance matrices is also used in this work.

Unlike the standard client-vs-UBM approach [1] of separately training a client model for each speaker independently of the UBM, the basic idea in Maximum A-Posteriori (MAP) adaptation is to derive the clients model by updating the well trained parameters of the UBM via a form of Bayesian adaptation. It has been found empirically that best performance is obtained when only the means are updated [8].

In an approximation of Kullback-Liebler (KL) divergence to compare the distributions from which two utterances \( \mathbf{Y} \) and \( \mathbf{Z} \) are drawn, Campbell et al. proposes that a UBM have its means \( \mu_i \) MAP adapted for each utterance to become \( \mu_i^Y \) and \( \mu_i^Z \) while leaving the covariances and mixture weights unchanged [4]. The following linear kernel (GSLK) between two utterances \( \mathbf{Y} \) and \( \mathbf{Z} \) is an approximation of the KL divergence between the two utterances that can be used in an SVM classifier

\[
k(\mathbf{Y}, \mathbf{Z}) = \sum_{i=0}^{N} \lambda_i \mu_i^Y^T \Sigma_i^{-1} \mu_i^Z
\]

where \( \Sigma_i \) is the \( i^{th} \) covariance matrix of the GMM UBM and \( \lambda_i \) is the \( i^{th} \) mixture weight [4]. While this is an example of a kernel that compares distributions it can also be viewed as a fixed dimension projection

\[
\left( \sqrt{\lambda_i \Sigma_i^{-\frac{1}{2}}} \mu_i \right)_{i=1}^{N}
\]

where the means are weighted and stacked to form the supervector and a linear kernel is then employed.
4. Vector Quantisation

Vector Quantisation is a lossy compression technique that can be summarised as follows [11], given an input set of training vectors $X = \{x_1, \ldots, x_l\}$, a set of codebook vectors $C = \{c_1, \ldots, c_k\}$ is found that partitions the input space. An input sequence can then be encoded as $e = \{e_0, \ldots, e_l\}$ where $e_i$ is an integer index in the range $1 \leq e_i \leq k$ for the closest codebook vector $c_{e_i}$ used to encode $x_i$.

An example of a Vector Quantisation (VQ) codebook can be seen in Figure 2 where the diamonds are the input sequence, these input vectors would be represented by their closest codebook vector.

The codebook is constructed so that some measure of distortion from the training sequence to the codebook vectors is minimised [9]. A common measure of distortion is the average squared Euclidean distance of the input frames from their codebook vectors.

One popular method for finding codebook vectors is the Linde-Buzo-Gray (LBG) algorithm [7] which has been used in the past for speaker verification [9, 11]. This algorithm operates as follows: an initial codebook vector is chosen to be the mean of the input data points, this codebook vector is split to create two codebook vectors by shifting the codebook slightly left and right, and then an iterative $k$-means clustering that takes the codebooks as seeds is used until the codebook meets a convergence criterion. The resulting codebook vectors are then each split as before and the $k$-means process is again run until the convergence criterion is met. This process of split and converge is repeated until the required number of codebook vectors is found. The resulting number of codebook vectors is a power of two, i.e., an exact number of bits to encode each frame of the input sequence to a codebook vector.

4.1 VQ Supervector

In MAP adapted UBMs it is normal that only the mean vectors of the GMM are adapted while the globally trained covariances and weights are used for all speakers [8]. This raises the question of whether the variances and weights are needed [5]. In the MAP adapted GMM supervectors described in [4] only the updated means are stacked to create a projection for a given utterance (see equation 1). Hence it is reasonable to believe that comparable performance can be found by creating a supervector based on a global VQ model.

In order to build a VQ based supervector the following logic can be applied. On a large pool of training data (similar to the GMM UBM) a VQ codebook $C$ is built. If this VQ codebook was updated based on an utterance $X = \{x_1, \ldots, x_l\}$ each $x_j$ would be assigned to its nearest codebook vector $c_j$. Let $S_i$ be the set of input vectors from $X$ assigned to codebook vector $c_i$ and $\mu_i$ be the mean of the vectors in $S_i$. Then if $S_i$ is non-empty and the VQ codebook was incremented via $k$-means by including the new input sequence $X$, codebook vector $c_i$ would move in the direction $v_i = \mu_i - c_i$.

By considering the codebook $C$ as a vocabulary of sounds for human speech, each individual speaker will use this vocabulary in a slightly different manner. By stacking the variations $v_i$ into a supervector the classifier can discriminate between how individuals use the vocabulary. However, $v_i$ does not have information pertaining to the quantity of speech that was encoded in its respective cell, and so these $v_i$ are prone to noise as shown with codebook 2 in figure 2. In order to mitigate for this we scale the vector $v_i$ by the proportion of input frames that fell into that cell. Thus cells with small amounts of data have their shift vectors scaled back. This results in codebook adaption similar to the proposed adaptation of a client’s vocabulary proposed in early VQ work [11].

4.2 Efficient computation

The construction of the VQ world model is less computationally complex than its GMM counterpart. As with its GMM counterpart, the calculation of the model is trivial to parallelise. Through the use of an efficient nearest neighbour technique such as KD-Trees, VQ supervectors are less computationally expensive to compute than their GMM counterpart.

As with the GMM supervector method, the VQ supervector method can be made more efficient at classification time by using a trick that depends on its use of a linear kernel [4]. This is because of a model com-
GMM and VQ supervectors were built
value of the sub-
coefficients where used

As in [4], the adaptation coefficient $r$ for the GMM MAP adaptation was set to be 16. The $C$ value of the SVM was set to be infinite forcing the SVM to find a hard margin solution. Given that the cepstral frames were made up of a total of 12 Mel and 12 \( \Delta \) coefficients, each component of the model contributed 24 features, for a model of 1024 this yields a feature space of 24,576 in dimensionality. For each speaker in the King dataset all 9 target utterances and 432 non-target utterances were converted to supervectors for training and for the YOHO dataset all 96 target utterances and 1,500 randomly sampled non-target utterances were used to train the SVM. It should be noted that speech from unknown impostors used in testing were not present during training.

**Observations:** GMM and VQ supervectors were built based on different model sizes as shown in Figure 4(a). Improvements for both methods over the baseline method (figure 4(b) and 4(c)) show that the improvement is of the same order. The fact that GMM supervectors perform marginally better in the short utterance (YOHO) experiments may be due to the VQ model creating a hard partition on the input vectors, and so a small proportion of the cepstral vectors make up each \( v_i \) subcomponent of the supervector. As the GMM supervector has a soft partitioning this results in all input vectors contributing to all subcomponents of the supervector.

On longer free speech utterances provided by the King dataset, both methods perform equally well. For longer utterances larger numbers of cepstral frames make up each subcomponent of the VQ supervector, so the soft clustering provided by the GMM does not provide a considerable de-noising benefit.

In order to “soften” the partitions for shorter utterances while maintaining the iterative scanning of files,

\[
f(x) = \left( \sum_{i=1}^{s} \alpha_i \phi(s_i) \right) \phi(x) + b
\]

where \( \phi(x) \) is the mapping of the utterance to the supervector and \( s_i \) and \( \alpha_i \) are the support vectors and their corresponding weights. Thus \( \left( \sum_{i=1}^{s} \alpha_i \phi(s_i) \right) \) can be compacted to be a single model vector to be compared against at classification time.

This efficiency could be taken a step further if one wished to pass a sliding window of a fixed number of frames across an audio file to identify sections where a speaker appears. For the VQ model each frame \( f \) contributes to only one small subsection of the supervector, whereas for the GMM supervector each frame contributes to all the elements of the supervector. This means as one frame is added from the front of the scanning window and last frame is dropped (see Figure 3) at most two sections of the VQ supervector need updating. This enables a small diff in the overall dot product to be calculated (corresponding to the two updated subsections) and so only a small amount of computation is required to update the match score for the window. The combination of these two efficiencies should allow VQ supervectors to be applied to continuous audio streams.

**5 Evaluation**

**Experimental Setup:** A comparison of these two techniques has been performed on two well-known datasets (YOHO and King) is is provided. YOHO provides a dataset of 138 speakers with short fixed vocabulary utterances (3 double digit figures) while King provides a dataset of 51 speakers with longer free vocabulary conversational speech. The evaluation provided in this paper examined the performance of MAP adapted GMM supervectors and VQ supervectors across a range of model sizes, (number of components for the GMM and number of codebook vectors for the VQ model). In this work, MFCCs and their \( \Delta \) coefficients where used to parameterise each utterance. Parameterisation, the train-test protocol and the building of the GMM-UBM followed the procedure shown in [2]. The VQ models were built using the same setup as the GMM-UBM. For each VQ model, the cepstral features were normalised to mean zero and unit variance so that features with higher variance would not have more influence in the placement of the VQ codebook vectors.

As a baseline the classic client-vs-UBM model GMM as implemented in [2] was used to compare performance improvements. Errors are reported using the Equal Error Rate (EER) found over all the speakers [1].

As in [4], the adaptation coefficient $r$ for the GMM MAP adaptation was set to be 16. The $C$ value of the SVM was set to be infinite forcing the SVM to find a hard margin solution. Given that the cepstral frames were made up of a total of 12 Mel and 12 \( \Delta \) coefficients, each component of the model contributed 24 features, for a model of 1024 this yields a feature space of 24,576 in dimensionality. For each speaker in the King dataset all 9 target utterances and 432 non-target utterances were converted to supervectors for training and for the YOHO dataset all 96 target utterances and 1,500 randomly sampled non-target utterances were used to train the SVM. It should be noted that speech from unknown impostors used in testing were not present during training.
it may be worth considering allowing each input vector to be assigned to its $k$-nearest codebook vectors for the VQ model. This may provide a more stable mapping for shorter utterances.

To find a comparison of the time required to build each supervector, we took 400 utterances from the King dataset and built the corresponding supervector on a single processor. For models of size 512, the VQ supervector was 70% faster to compute and for models of size 1024 the VQ supervector was 133% faster.

6. Conclusions and Further Work

This paper has demonstrated an adaptation projection based on Vector Quantisation. While this technique is less computationally complex than its GMM counterpart, it provides equivalent performance. This paper shows that while GMM projections provide state-of-the-art performances, it is worth re-investigating other models. Its has been shown that other models such as VQ can provide equally rich projections to those given by GMMs with lower computational costs.

As with GMMs, model compaction can be used with SVMs to speed up classification times. An additional benefit of the VQ mappings match score is that it can be partially iteratively incremented and decremented to allow a fixed length window to be passed across the contents of a media file.

Due to the lower complexity of this approach it may provide a good building block for other techniques. An examination of the impact of the VQ model projection in conjunction with other techniques that have used MAP GMM supervectors would be interesting.

References

[1] F. Bimbot, J. Bonastre, C. Fredouille, G. Gravier, I. Magrin-Chagnolleau, S. Meignier, T. Mer-