NEW TECHNIQUES FOR IMPROVING THE PRACTICALITY OF AN SVM-BASED SPEECH/MUSIC CLASSIFIER

Chungsoo Lim, Seong-Ro Lee, and Yeon-Woo Lee

Mokpo National University
Mokpo, South Korea
{clim, srlee, ylee}@mokpo.ac.kr

Joon-Hyuk Chang

Hanyang University
Seoul, South Korea
jchang@hanyang.ac.kr (Corresponding author)

ABSTRACT

Variable bit-rate coding introduced for effective utilization of limited communication bandwidth requires accurate classification of input signals. This paper investigates implementation of a support vector machine (SVM)-based speech/music classifier in the selectable mode vocoder (SMV) framework, which is a standard codec adopted by the Third-Generation Partnership Project 2 (3GPP2). A support vector machine is well known for its superior pattern recognition capability; however, it is accompanied by a high computational cost. In order to achieve a more practical system, three techniques are proposed for the SVM-based speech/music classifier. The first is to prune support vectors that least contribute to the output of the SVM, while the other two are aimed at reducing the number of classification requests to the SVM-based classifier by eliminating or redirecting some of the classification requests to the classifier.

Index Terms— Codecs, classification algorithm, embedded software.

1. INTRODUCTION

Recently, a number of variable bit-rate speech codecs have been proposed as efficient utilization of limited bandwidth gains more attention due to increased demand for multimedia services through wireless communication devices. To assign variable bit-rate according to input signal type, accurate classification of signals should be performed first. Among many classifications, discriminating speech from music is of prime importance because a different bit-rate should be assigned to music in order to preserve its quality. One of the codecs supporting variable bit-rate coding is the selectable mode vocoder (SMV) adopted by the Third-Generation Partnership Project 2 (3GPP2) [1]. Recently, a speech/music classifier based on a support vector machine (SVM) in the SMV framework was proposed, showing acceptable performance [2]. Specifically, the SVM-based classifiers utilize parameters generated inside the SMV without any further processing and considerably ameliorate the classification performance. However, SVMs typically require a tremendous amount of computation because of the inner vector products with a large number of support vectors. The SVM parameters directly related to the computational intensity are the number and dimensionality of the support vectors. A great deal of research has been conducted to optimize these parameters, including selection of relevant features and simplification of support vectors [3]. Note that these approaches operate during the design and/or learning phases of SVMs. Another major contributor to the high computational intensity of the SVM-based classifier is the number of times the classifier executes classifications.

Based on the above observations, we propose three effective techniques to reduce the high computational cost with little impact on the classification performance. The first technique is to simplify the trained support vectors by pruning the support vectors that have small contribution to the final output of the classifier. The second technique is to reduce the number of classifications that the classifier has to perform by filtering out classification requests that can be inherently handled by the built-in speech/music classifier in the SMV codec. The final technique also lessens the classification burden on the SVM-based classifier by skipping some of the input frames based on the strong inter-frame correlations in speech and music frames.

2. PRUNING SUPPORT VECTORS BASED ON THEIR CONTRIBUTIONS TO CLASSIFICATION

In this section, a novel way to simplify support vectors based on their contributions to final outcomes is proposed. The decision function of SVMs has the following form.

$$f(\bar{x}(t)) = \sum_{i-1}^{M} \alpha_i^* y_i K(\bar{x}_i^*, \bar{x}(t)) + b^* \qquad (1)$$

where $\bar{x}_i^*$ is the i^{th} vector of M support vectors, and $\bar{x}(t)$ is the t^{th} input frame vector. Optimization bias b^* and La-

This research was supported by the MKE, Korea, under the ITRC support program supervised by the NIPA(NIPA-2011-C1090-1121-0007) and by Priority Research Centers Program through the NRF funded by the Ministry of Education, Science and Technology(2011-0022980)

grange multiplier α^* are obtained by solving a quadratic programming problem. The kernel function $K(\bar{x}_i^*, \bar{x}(t))$ is used when input vectors are not linearly separable. If a radial basis function (RBF) is used, it is defined as the following:

$$K(\bar{x}_i^*, \bar{x}(t)) = exp(-\gamma \|\bar{x}_i^* - \bar{x}(t)\|^2) \quad (2)$$

where γ is the kernel parameter of RBF associated with the width of RBF. Since RBF is frequently used for linearly inseparable cases and related work [2] choose RBF as the kernel function, RBF is the kernel function in this study.

As seen from the above equations, the contribution of each support vector is determined by its corresponding α^* value and $K(\bar{x}_i^*, \bar{x}(t))$. From (2), it is found that the output of the kernel function depends on the distance between a support vector $\bar{x}_i^*$ and an input frame vector $\bar{x}(t)$ [2]. This means that if a support vector is a far distance from most of the input frames, its corresponding kernel function outputs are very small because of the small contribution of the exponential function in (2) to the final classifications. Thus, support vectors with small Lagrange multipliers or ones far from the input vectors should be found. To find support vectors that are a great distance from the other vectors, it is assumed that support vectors which are far from other support vectors are likely to also be a great distance from the input vectors. With this assumption, the problem of finding support vectors that have a large distance from possible input vectors changes to the problem of finding outliers among the support vectors.

In order to find outliers among given support vectors, an outlier detection algorithm based on the distribution of distances among data points [4] is used. This outlier detection algorithm is based on a frequency function defined as

$$g(d) = g^n(d) - g^u(d) \quad (3)$$

where $g^n(d)$ is the frequency distribution of given data points, and $g^u(d)$ is the corresponding distribution of uniformly distributed points of the same dimensionality. If the points (distances in our case) in the given data-set are uniformly distributed, the frequency function produces values near zero for all distances. If the points are not uniformly distributed, the highest values produced by the frequency function indicate the most frequent and most characteristic inter-point distances, and the lowest values indicate atypical distances. For a support vector i, the outlier factor is defined as

$$R_i = \frac{1}{m} \sum_{j=1 j \neq i}^{m} g(d(\bar{x}_i, \bar{x}_j)) \quad (4)$$

where m is the number of given support vectors, $d(\bar{x}_i, \bar{x}_j)$ is the distance between the i^{th} and j^{th} support vectors, and $g(d)$ is the difference function introduced in (3). The outlier support vectors tend to have the lowest outlier factors because the distances between the outlier support vectors and the other support vectors are rare, and the corresponding frequency function values given in (3) are low. Hence, the two conditions for determining whether a support vector needs to be pruned (H_P) or not (H_{NP}) are as follows.

$$|\alpha_i^*| \underset{H_P}{\overset{H_{NP}}{\gtrless}} \eta_\alpha, \qquad |R_i| \underset{H_P}{\overset{H_{NP}}{\gtrless}} \eta_R \quad (5)$$

where η_α and η_R are thresholds for identifying candidates for support vector pruning. Subsequently, if at least one of these two conditions is met, the corresponding support vector is eliminated. When η_α and η_R are set to 0.05 and 0.095, respectively, 16.2% of the trained support vectors are pruned without any detrimental impact on classification accuracy.

3. FILTERING CLASSIFICATION REQUESTS BY HIERARCHICAL CLASSIFICATION

In this section, a filtering mechanism that reduces the usage of the SVM-based classifier is presented. The basic idea behind this technique is to filter out classification requests that can be successfully handled by a much simpler speech/music classifier, lessening the burden on the SVM-based classifier. In other words, a hierarchical classifier is formed by combining the SVM-based classifier as the second level classifier and a simpler classifier as the first level classifier. The original speech/music classifier in the SMV is a good candidate for the first level classifier because of the following properties. First, it is already a built-in classifier in the SMV so there is no need for any extra processing. Second, it is much simpler than the SVM-based classifier because it simply requires the comparison of two parameters with two given threshold values. Third, frames classified as music, are actually music frames with high accuracy. Note that the rate between the number of correctly classified frames as music and the total number of music frames is very low (17%), but the ratio between the number of frames correctly classified as music and the number of classifications as music is quite high (90%).

Nevertheless, the original classifier needs to be modified to maximize the impact of filtering while maintaining the accuracy. To be more specific, the original thresholds, the running average of the periodicity counter and the music continuity counter, need to be adjusted. Fig. 1 shows an example of the changes in accuracy of the first level classifier and the ratio between the filtered requests and the total number of requests as the thresholds for the classifier vary. The x-axis represents the threshold pair (music continuity counter/running average of the periodicity counter), and the y-axis shows both the accuracy of the classifier and the filtering ratio. Note that the thresholds for the music continuity counter and the running average of the periodicity counter used in the original speech/music classifier were 200 and 18, respectively [1]. As the thresholds decrease, the accuracy decreases slowly at the outset but decreases rapidly at the end of the process. The filtered ratio reacts in the opposite manner to the accuracy. Therefore, threshold values should correspond to a good

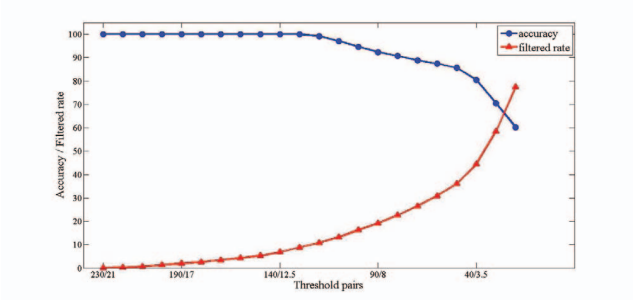

Fig. 1. Impact of the thresholds for the original speech/music classifier on its accuracy and the filtering ratio

tradeoff between the accuracy and the filtered rate. In our experiments, an accuracy comparable to that of the SVM-based classifier [2] is chosen to maintain the overall classification accuracy of the SVM-based classifier, while reducing the execution time and energy consumption via filtering.

4. SKIPPING CLASSIFICATION REQUESTS BY VIRTUE OF INTER-FRAME CORRELATIONS

In this section, an efficient technique to reduce the classification load on the SVM-based classifier by skipping classification requests based on inter-frame correlations generally found in speech/music frames is introduced. Generally, strong correlations exist in speech/music frames because of their inherent structure. The speech/music signals used in our experiments are made up of three distinct segments: speech, music, and silence. Each segment tends to lasts for at least several seconds, thereby existing as a group of frames. Accordingly, it is highly probable that the current frame belongs to the same class as the previous frames when considering the inter-frame correlation of consecutive frames. With this property, one frame is classified and, its classification is used for the predefined number of next frames, which thus require no actual classification. Unfortunately, however, we do not have *apriori* information about the class of each frame. Therefore, the classifications made by our classifier are used, which might decrease the classification accuracy.

To avoid performance degradation, a simple threshold mechanism is adopted. If the output of the SVM-based classifier is less than the threshold, the classification lacks confidence, and vice versa. For a diffident classification, skipping is abandoned and normal classification is performed; however, for a confident classification, a predefined number of subsequent frames are skipped. The condition for skipping (H_S) or not (H_{NS}) is described as

$$|f(\bar{x}(t))| \underset{H_{NS}}{\overset{H_S}{\gtrless}} \eta_f \tag{6}$$

where η_f is a given threshold.

In order to examine the skipping mechanism, the number of skipped frames per classification is varied, and how this affects the classification accuracy and the ratio between the skipped requests and the total number of requests is shown in Fig. 2. In the figure, the x-axis indicates the number of skipped frames between two classified frames, and the y-axis represents the classification accuracy and skipping ratio. The threshold η_f is set to 0.5, The results are averaged values from 20 test files described in the next section with η_f set to 0.5. Note that the accuracies are normalized values relative to the accuracy of the original SVM-based classifier. It can be readily seen that the classification accuracy is well maintained with only a 0.3% decrease as the number of skipped frames increases. The skipping ratio increases rapidly when the number of skipped frames is small, but it starts to be saturated when the number of skipped frames is six.

5. EXPERIMENTS AND RESULTS

To evaluate the proposed techniques, execution time, energy consumption, and classification accuracy are measured. The SVM-based classifier in the SMV [2] was used as the base case where the proposed techniques were applied. The experiments were performed on the TIMIT speech database [5] and commercial music CDs. From the CDs, songs from five different genres were collected, each of them being about 5 min in length. For training, 20 min of data from the speech and music database was used, while 20 files were constructed (4 files per genre) for testing with no overlap with the data used for training. Each test file was about 5 min long and had alternating speech and music segments, and the length of each segment varied from 5 to 30 s. All data was sampled at 8 kHz with a frame size of 20 ms.

The six features adopted for the SVM-based classifier [2] were used with no modification. Note that these six features were the parameters originally generated inside the SMV. As mentioned before, RBF was used as the kernel function for the SVM with its width parameter set to 0.01.

To measure the execution time and energy consumption, a

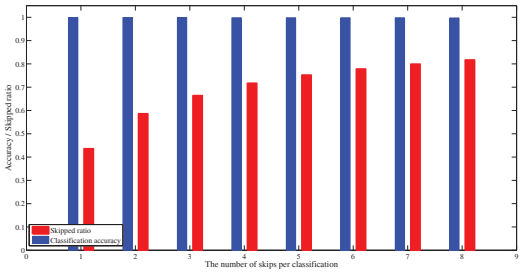

Fig. 2. Impact of the number of skipped frames per classification on the classification accuracy and the skipping ratio

Table 1. Parameters for the processor simulator.

ISA	ARMv4T
Clock Freq	800 Mhz
Pipeline	5 stages, in-order, dual issue
Branch	Dynamic prediction (bimodal)
L1 cache	64 KB separate Inst/Data Cache
	4way, 32 B line, 1 cycle latency
L2 Cache	256 KB unified Inst/Data Cache
	4way, 64 B line, 10 cycle latency
Memory	64 bit bus, 100 cycle latency, 1 port
FPU	1 adder (2 cycle, pipelined)
	1 multiplier (4 cycle, pipelined)

processor simulator called sim-panalyzer [6] was used. Since sim-panalyzer models both processor architecture and power consumption in detail, execution time and energy consumption can be measured simultaneously. The simulation environment was configured to simulate a system with ARM instruction set architecture (ISA). Table 1 captures the processor configuration used for the experiment.

The three techniques were applied individually and simultaneously to test every possible combination of techniques. The effectiveness of each combination was evaluated in terms of classification accuracy, execution time, and energy consumption, as shown in Table 2. Note that the execution time and the energy consumption were those for a single frame. '$Base$' is the SVM-based classifier introduced in [2], and 'P,' 'F,' and 'S' represented the support vector pruning technique described in Section 2, the classification request filtering technique explained in Section 3, and the skipping technique described in Section 4, respectively. For the 'S' technique, the number of skipped frames per classification was set to eight.

For an individual technique, the skipping technique was the best in terms of execution time and energy consumption. For combinations of two techniques, the combination of the skipping technique and the filtering mechanism was the best. When all three techniques were combined, the shortest execution time and the lowest energy consumption were achieved. This behavior indicated that the techniques could be combined with accumulated benefits.

Table 2. Effectiveness of the proposed techniques in terms of classification accuracy, execution time, and energy consumption for processing a frame.

Techniques	Accuracy	Execution time [s]	Energy [j]
Base [2]	0.8184	0.0165	0.0588
Base-P	0.8228	0.0138	0.0488
Base-F	0.8239	0.0130	0.0462
Base-F-P	0.8270	0.0109	0.0385
Base-S	0.8159	0.0030	0.0105
Base-S-P	0.8202	0.0025	0.0093
Base-F-S	0.8197	0.0023	0.0082
Base-F-S-P	0.8225	0.0019	0.0068

6. CONCLUSIONS

This paper proposes three techniques that reduce the computational complexity and energy consumption of an SVM-based speech/music classifier in the SMV framework in order to facilitate effective implementation of the classifier. Firstly, a pruning technique that removes unnecessary support vectors from a trained set of support vectors can reduce execution time and energy consumption by 21%. Secondly, a hierarchical classifier is introduced, in which the first level classifier can filter out a significant portion of input frames, reducing the burden on the second level classifier. Lastly, a skipping mechanism based on inter-frame correlation can considerably reduce the number of classification requests to the SVM-based classifier.

7. REFERENCES

[1] Y. Gao, E. Shlomot, A. Benyassine, J. Hyssen, Huan yu Su, and C. Murgia, "The SMV algorithm selected by TIA and 3GPP2 for CDMA applications," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, May 2001, pp. 709–712.

[2] S.-K. Kim and J.-H. Chang, "Speech/music classification enhancement for 3GPP2 SMV codec based on support vector machine," *IEICE Trans. Fundamentals of Electronics, Communications and Computer Sciences*, vol. E92-A, no. 2, 2009.

[3] D. Nguyen and T. Ho, "An efficient method for simplifying support vector machines," in *Proc. International Conference on Machine Learning*, Aug. 2005, pp. 617–624.

[4] V. Saltenis, "Outlier detection based on the distribution of distances between data points," *Informatica*, pp. 399–410, May 2005.

[5] W. M. Fisher, G. R. Doddington, and K. M. Goudie-Marshall, "The DARPA speech recognition research database: Specification and status," in *Proc. DARPA Workshop Speech Recognition*, Feb. 1986, pp. 93–99.

[6] T. Austin, T. Mudge, and D. Grunwald, "Sim-panalyzer," *http://www.eecs.umich.edu/ panalyzer/.*