Sound Ranking Using Auditory Sparse-Code Representations

Martin Rehn, Richard F. Lyon, Samy Bengio, Thomas C. Walters, Gal Chechik

May 13, 2009

The task of ranking sounds from text queries is a good test application for machine-hearing techniques, and particularly for comparison and evaluation of alternative sound representations in a large-scale setting. We have adapted a machine-vision system, "passive-aggressive model for image retrieval" (PAMIR) [2], which efficiently learns, using a ranking-based cost function, a linear mapping from a very large sparse feature space to a large query-term space. Using this system allows us to focus on comparison of different auditory front ends and different ways of extracting sparse features from high-dimensional auditory images. In addition to two main auditory-image models, we also include and compare a family of more conventional Mel-Frequency Cepstral Coefficients (MFCC) front ends. The experimental results show a significant advantage for the auditory models over vector-quantized MFCCs. The two auditory models tested use the adaptive pole-zero filter cascade (PZFC) auditory filterbank and sparse-code feature extraction from stabilized auditory images via multiple vector quantizers. The models differ in their implementation of the strobed temporal integration used to generate the stabilized image. Using ranking precision-at-top-k performance measures, the best results are about 72% top-1 precision and 35% average precision, using a test corpus of thousands of sound files and a query vocabulary of hundreds of words.

1 Modeling Sounds

Sparse representations of signals have received a lot of attention recently. When appropriately extracted, such representations can then be very efficiently coupled to linear models and still yield highly scalable state-of-the-art performance in many applications.

We are thus looking for a method to extract a sparse representation of sound files in order to use them as input to a linear ranking system. Our baseline auditory feature extraction system is based on MFCCs. We calculated the standard 13 MFCC coefficients, together with their first and second differences, and removed the (first) energy component, yielding a vector of 38 features per time frame. We used standard parameters for calculating the MFCCs, resulting in that each sound file was represented by a series of a few hundred 38-dimensional vectors. In order to obtain a sparse representation, we then took the following approach: we used k-means to cluster the set of all MFCC vectors extracted from our training data. We used various codebook sizes, up to 8000, making 8000-dimension feature vectors, with each component being the number of times the corresponding codeword was found in the sound file, and most components usually being zero.

Our novel front ends are based on models of the mammalian auditory system, particularly using cochlear-model filterbanks from which a "stabilized auditory image" (SAI) [3] is formed. Computing auditory features follows three steps: SAI computation, box cutting, and vector quantization.

The SAI stage of processing produces a series of two-dimensional frames of real-valued data. These frames can be thought of as images, with the cochlear channel number on the vertical axis and time lags relative to identified strobe times on the horizontal axis.

To arrive at the sparse code we first cut up each SAI frame into smaller, overlapping rectangles, or "boxes." These have different sizes in order to capture information at multiple scales in the SAI frame. The horizontal and vertical sizes are varied independently. We also vary the vertical position of the boxes. We then rescale the subimages in these boxes into a fixed size, the same for all boxes, regardless of their original sizes. For each box we then optionally compute the horizontal and vertical marginals – the average values for each column and row in the box – and concatenate the marginals into a single vector per box. Alternatively, we keep each box contents as a single vector, without reducing it to its marginals.

In the final step, we approximate each of the vectors that represent the boxes in the SAI frame with sparse codes, using either vector quantization or matching pursuit. Vector quantization means that a box is approximated by the best matching vector from a codebook (in the Euclidean sense). Once the best match has been chosen, the representation can be encoded as a sparse code vector, with a length equal to the size of the codebook, that consists of all zeros, except for a single "one" at the index position of the chosen code word. Matching pursuit means that we project a vector (representing a box) onto the codebook vectors, find the largest projection, add the signed scalar value of that projection to the sparse vector representation (in the appropriate index position), and subtract the vector valued projection from the original vector, producing a residual vector. The process is then repeated until the magnitude of the largest projection becomes smaller than a given threshold. For both matching pursuit and vector quantization we learn individual codebooks using k-means, in order to represent the boxes at each specific position in the SAI.

Once each box has been converted into a sparse code (using vector quantization or matching pursuit) they are concatenated into one high-dimensional sparse vector, representing the entire SAI frame. For example, a frame might have 100 boxes, each with a codebook of size 256, for a total feature vector of 25600 dimensions, with only 100 nonzero elements.

To represent an entire sound fragment or file, we combine the sparse vectors representing individual frames into a sparse vector representing the entire fragment, by simply summing them up. The resulting vector will be less sparse than the vectors for individual frames, but it will still be sparse.

2 Ranking Sounds Given Text Queries

The black-box view of the ranking task is that a user enters a search query, and in response is presented with an ordered list of sound documents, ranked by relevance to the query. Inside the black box, the ordering of documents is based solely on acoustic content: no annotations or other metadata are available to the system at retrieval time. Rather, at training time, a set of annotated sound documents (sound files with associated textual tags) is made available. In this way, a small labeled set can be used to enable content-based retrieval from a much larger, unlabeled set.

We extracted sparse features from all sound documents. We then trained a machine learning system to rank the documents using the extracted features. We used PAMIR [2], an efficient online large-margin approach with a ranking criterion that was used effectively to rank images from text queries. In a previous paper on sound ranking [1], we compared PAMIR to various other machine learning approaches using MFCC based features and concluded that PAMIR was by far the most scalable one and yet obtained the best performance most of the time.

3 Dataset and Experiments

Our data set consists of 8638 sound effects, collected from several sources. Close to half of the sounds (3855) come from commercially available sound effect collections. The remaining 4783 sounds are taken from a variety of web sites; *www.findsounds.com*, *partners in rhyme*, *acoustica.com*, *ilovewavs.com*, *simplythebest.net*, *wav-sounds.com*, *wavsource.com*, and *wavlist.com*. Most of the sounds contain only a single 'auditory object', and contain the 'prototypical' sample of an auditory category. Many sounds are short (a few seconds) but there are also some that are several minutes in length.

We applied 3-Fold cross-validation on the set of sound files in order to estimate the performance of the compared systems. For each front end, we varied a number of parameters from a baseline condition, and evaluated the resulting system's precision-at-top-k (for k ranging from 1 to 20) and average precision.

4 Results

Figure 1 shows that auditory representations are found to be significantly better than MFCC representations for all operating points k of the precision-at-top-k curve. Smaller differences include: vector quantization better than matching pursuit; box reduction to "marginals" better than full box subimages; more boxes better than fewer; bigger codebooks better than smaller.

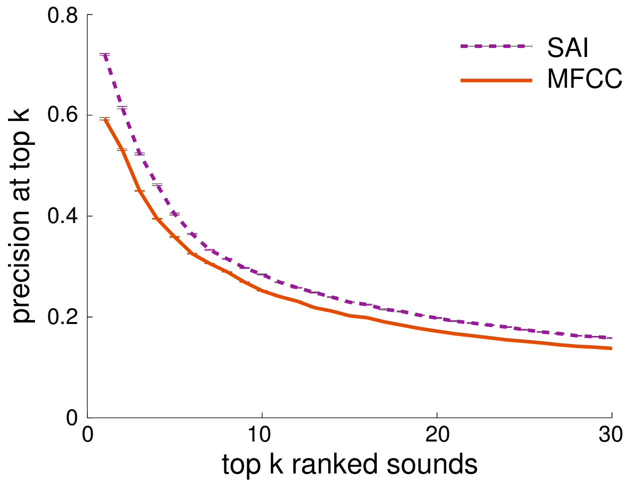

Figure 1: Precision at top k of SAI- and MFCC- based systems, as a function of k, the number of top ranked documents. Error bars denote the standard error of the mean over 1073 test queries.

References

[1] G. Chechik, E. Ie, M. Rehn, S. Bengio, and D. Lyon. Large-scale content-based audio retrieval from text queries. In *ACM International Conference on Multimedia Information Retrieval, MIR*, 2008.

[2] D. Grangier and S. Bengio. A discriminative kernel-based model to rank images from text queries. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 30(8):1371–1384, 2008.

[3] R. D. Patterson, K. Robinson, J. Holdsworth, D. McKeown, C. Zhang, and M. Allerhand. Complex sounds and auditory images. In *Auditory physiology and perception, Proceedings of the 9h International Symposium on Hearing*, pages 429–446, 1992.