Multimodal Sparse Coding for Event Detection

Youngjune Gwon William M. Campbell Kevin Brady Douglas Sturim
MIT Lincoln Laboratory, Lexington, MA 02420, USA

Miriam Cha H. T. Kung
Harvard University, Cambridge, MA 02138, USA

Abstract

Unsupervised feature learning methods have proven effective for classification tasks based on a single modality. We present multimodal sparse coding for learning feature representations shared across multiple modalities. The shared representations are applied to multimedia event detection (MED) and evaluated in comparison to unimodal counterparts, as well as other feature learning methods such as GMM supervectors and sparse RBM. We report the cross-validated classification accuracy and mean average precision of the MED system trained on features learned from our unimodal and multimodal settings for a subset of the TRECVID MED 2014 dataset.

1 Introduction

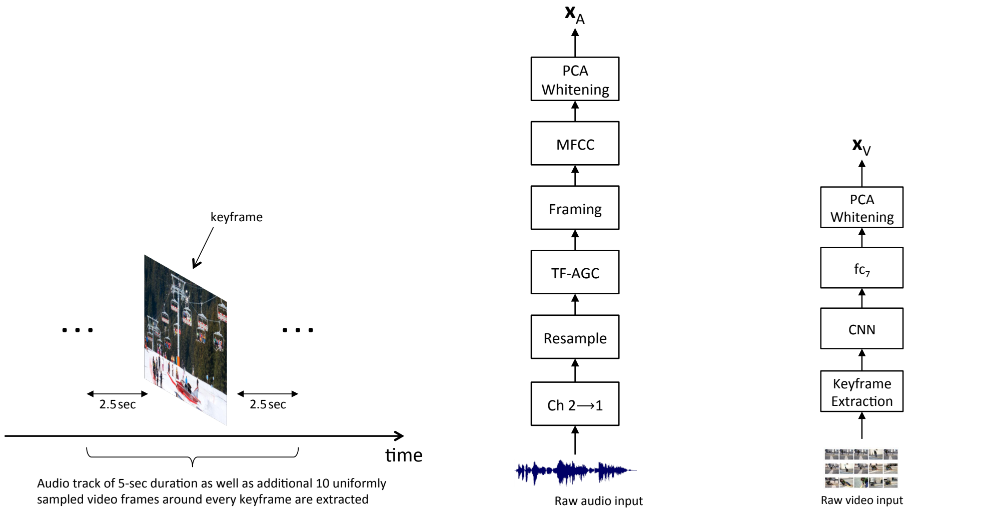
Multimedia Event Detection (MED) aims to identify complex activities occurring at a specific place and time involving various interactions of human actions and objects. MED is considered more difficult than concept analysis such as action recognition and has received significant attention in computer vision and machine learning research. In this paper, we propose the use of sparse coding for multimodal feature learning in the context of MED. Originally proposed to explain neurons encoding sensory information [8], sparse coding provides an unsupervised method to learn basis vectors for efficient data representation. More recently, sparse coding has been used to model the relationship between correlated data sources. By jointly training a dictionary using audio and video tracks from the same multimedia clip, we can force the two modalities to share a similar sparse representation whose benefit includes robust detection and cross-modality retrieval.

In the next section, we will describe audio-video feature learning in various unimodal and multimodal settings for sparse coding. We then present our experiments with TRECVID MED dataset. We will discuss the empirical results, compare them to other methods, and conclude.

2 Audio-video Feature Learning

In summary, our approach is to build feature vectors by sparse coding on the low-level audio and video features. Multiple feature vectors (*i.e.*, sparse codes) are aggregated via max pooling. The resulting pooled feature vectors can scale to describe the entire multimedia file. We use them to train an array of classifiers for MED events.

[*] This work was sponsored by the Department of Defense under Air Force Contract FA8721-05-C-0002. Opinions, interpretations, conclusions, and recommendations are those of the authors and are not necessarily endorsed by the United States Government.



(a) Keyframe extraction (b) Audio preprocessing (c) Video preprocessing

Figure 1: Preprocessing audio and video data from multimedia clip

2.1 Low-level feature extraction and preprocessing

We begin by locating the keyframes of a given multimedia clip. We apply a simple two-pass algorithm that computes color histogram difference of any two successive frames and determines a keyframe candidate based on the threshold calculated on the mean and standard deviation of the histogram differences. Using 256 bins in the histogram, we examine the number of nonzero bins (reflecting the degree of color variation) in the keyframe candidates and discard the ones with less than 26 nonzero bins ($\approx$ 10% of 256). This ensures that our keyframes are not all-black or all-white blank images.

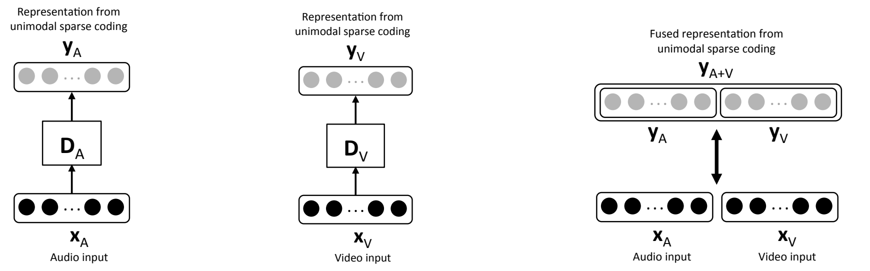
Around each keyframe, we extract 5-sec audio data and additional 10 uniformly sampled video frames within the duration as illustrated in Figure 1a. If extracted audio is stereo, we take only the left channel. The audio waveform is resampled to 22.05 kHz and regularized by the time-frequency automatic gain control (TF-AGC) to balance the energy in sub-bands. We form audio frames using a 46-msec Hann window with 50% overlap between successive frames for smoothing. For each frame, we compute 16 the Mel-frequency cepstral coefficients (MFCCs) as the low-level audio feature. In addition, we append 16 delta cepstral and 16 delta-delta cepstral coefficients, which make our low-level audio feature vectors 48 dimensional. Finally, we apply PCA whitening before unsupervised learning. The complete audio preprocessing steps are described in Figure 1b.

For video preprocessing, we take a deep learning approach. We have tried out pretrained convolutional neural network (CNN) models for the ImageNet Large-scale Visual Recognition Challenge (ILSVRC), namely GoogLeNet `imagenet-googlenet-dag` [11], the Oxford Visual Geometry Group (VGG) VD models `imagenet-vgg-verydeep-16` and `imagenet-vgg-verydeep-19` [10], and a Berkeley Caffe reference model `imagenet-caffe-alex` [6]. We have ended up choosing `imagenet-vgg-verydeep-19`. As depicted in Figure 1c, we run the CNN feedforward passes with the extracted video frames. For each video frame, we take 4,096-dimensional hidden activation from fc_7, the highest hidden layer before the final ReLU (*i.e.*, the rectification non-linearity). By PCA whitening, we reduce the dimensionality to 128.

2.2 High-level feature modeling via sparse coding

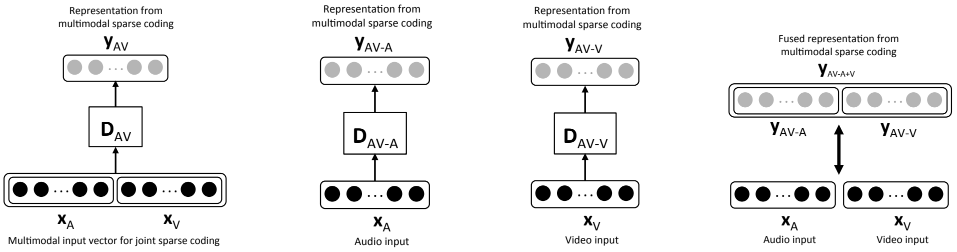
We use sparse coding to model high-level features that can train classifiers for event detection.

Unimodal feature learning. A straightforward approach for sparse coding with two heterogeneous data modalities is to learn a *separate* dictionary of basis vectors for each modality. Figure 2 depicts unimodal sparse coding schemes. Recall the preprocessed audio and video input vectors $\mathbf{x}_\text{A}$ and $\mathbf{x}_\text{V}$.


Representation from unimodal sparse coding
y_A
D_A
x_A
Audio input
Representation from unimodal sparse coding
y_V
D_V
x_V
Video input
Fused representation from unimodal sparse coding
y_A+V
y_A
y_V
x_A
x_V
Audio input
Video input

(a) Audio only (b) Video only (c) Union of unimodal features

Figure 2: Unimodal sparse coding and feature union


Representation from multimodal sparse coding
y_AV
D_AV
x_A
x_V
Multimodal input vector for joint sparse coding
Representation from multimodal sparse coding
y_AV-A
D_AV-A
x_A
Audio input
Representation from multimodal sparse coding
y_AV-V
D_AV-V
x_V
Video input
Fused representation from multimodal sparse coding
y_AV-A+V
y_AV-A
y_AV-V
x_A
x_V
Audio input
Video input

(a) Joint sparse coding (b) Cross-modal audio (c) Cross-modal video (d) Union of cross-modal features

Figure 3: Multimodal sparse coding and feature formation possibilities

Audio-only sparse coding is done by

$$\min_{\mathbf{D}_{\mathrm{A}},\mathbf{y}_{\mathrm{A}}^{(i)}} \sum_{i=1}^{n_{\mathrm{A}}} \|\mathbf{x}_{\mathrm{A}}^{(i)} - \mathbf{D}_{\mathrm{A}}\mathbf{y}_{\mathrm{A}}^{(i)}\|_2^2 + \lambda\|\mathbf{y}_{\mathrm{A}}^{(i)}\|_1 \tag{1}$$

where we feed n_{A} unlabeled audio examples to simultaneously learn the unimodal dictionary $\mathbf{D}_{\mathrm{A}}$ and sparse codes $\mathbf{y}_{\mathrm{A}}^{(i)}$ under the sparsity regularization parameter λ. (We denote $\mathbf{x}_{\mathrm{A}}^{(i)}$ the ith training example for audio.) Similarly, using n_{V} unlabeled video examples, we learn

$$\min_{\mathbf{D}_{\mathrm{V}},\mathbf{y}_{\mathrm{V}}^{(i)}} \sum_{i=1}^{n_{\mathrm{V}}} \|\mathbf{x}_{\mathrm{V}}^{(i)} - \mathbf{D}_{\mathrm{V}}\mathbf{y}_{\mathrm{V}}^{(i)}\|_2^2 + \lambda\|\mathbf{y}_{\mathrm{V}}^{(i)}\|_1. \tag{2}$$

We can form $\mathbf{y}_{\mathrm{A+V}} = [\mathbf{y}_{\mathrm{A}}\ \mathbf{y}_{\mathrm{V}}]^\top$, a union of the audio and video feature vectors from unimodal sparse coding illustrated in Figure 2c.

Multimodal feature learning. The feature union $\mathbf{y}_{\mathrm{A+V}}$ encapsulates both audio and video sparse codes. However, the training is done in a parallel, unimodal fashion such that sparse coding dictionary for each modality is learned independently of the other. To remedy the lack of joint learning, we propose a multimodal sparse coding scheme described in Figure 3a. We use the joint sparse coding technique used in image super-resolution [13]

$$\min_{\mathbf{D}_{\mathrm{AV}},\mathbf{y}_{\mathrm{AV}}^{(i)}} \sum_{i=1}^{n} \|\mathbf{x}_{\mathrm{AV}}^{(i)} - \mathbf{D}_{\mathrm{AV}}\mathbf{y}_{\mathrm{AV}}^{(i)}\|_2^2 + \lambda'\|\mathbf{y}_{\mathrm{AV}}^{(i)}\|_1. \tag{3}$$

Here, we feed the concatenated audio-video input vector $\mathbf{x}_{\mathrm{AV}}^{(i)} = [\frac{1}{\sqrt{N_{\mathrm{A}}}}\mathbf{x}_{\mathrm{A}}^{(i)}\ \frac{1}{\sqrt{N_{\mathrm{V}}}}\mathbf{x}_{\mathrm{V}}^{(i)}]^\top$, where N_{A} and N_{V} are dimensionalities of $\mathbf{x}_{\mathrm{A}}$ and $\mathbf{x}_{\mathrm{V}}$, respectively. As an interesting property, we can decompose the jointly learned dictionary $\mathbf{D}_{\mathrm{AV}} = [\frac{1}{\sqrt{N_{\mathrm{A}}}}\mathbf{D}_{\mathrm{AV-A}}\ \frac{1}{\sqrt{N_{\mathrm{V}}}}\mathbf{D}_{\mathrm{AV-V}}]^\top$ to perform the following audio-only and video-only sparse coding

$$\min_{\mathbf{D}_{\mathrm{AV-A}},\mathbf{y}_{\mathrm{AV-A}}^{(i)}} \sum_{i=1}^{n_{\mathrm{A}}} \|\mathbf{x}_{\mathrm{AV-A}}^{(i)} - \mathbf{D}_{\mathrm{AV-A}}\mathbf{y}_{\mathrm{AV-A}}^{(i)}\|_2^2 + \lambda''\|\mathbf{y}_{\mathrm{AV-A}}^{(i)}\|_1, \tag{4}$$

$$\min_{\mathbf{D}_{\mathrm{AV-V}},\mathbf{y}_{\mathrm{AV-V}}^{(i)}} \sum_{i=1}^{n_{\mathrm{V}}} \|\mathbf{x}_{\mathrm{AV-V}}^{(i)} - \mathbf{D}_{\mathrm{AV-V}}\mathbf{y}_{\mathrm{AV-V}}^{(i)}\|_2^2 + \lambda''\|\mathbf{y}_{\mathrm{AV-V}}^{(i)}\|_1. \tag{5}$$

In principle, joint sparse coding via Eq. (3) combines the objectives of Eqs. (4) and (5), forcing the sparse codes $\mathbf{y}_{\mathrm{AV-A}}^{(i)}$ and $\mathbf{y}_{\mathrm{AV-V}}^{(i)}$ to share the same representation. Note the relationship between the regularization parameters $\lambda' = (\frac{1}{N_{\mathrm{A}}} + \frac{1}{N_{\mathrm{V}}})\lambda''$. Ideally, we could have $\mathbf{y}_{\mathrm{AV}}^{(i)} = \mathbf{y}_{\mathrm{AV-A}}^{(i)} = \mathbf{y}_{\mathrm{AV-V}}^{(i)}$, although empirical values determined by the three different optimizations differ in reality. Feature formation possibilities on multimodal sparse coding are explained in Figure 3.

3 Evaluation

3.1 Dataset, task, and experiments

We use the TRECVID MED 2014 dataset [1] to evaluate our schemes. We consider the event detection and retrieval tasks using the 10Ex and 100Ex data scenarios, where 10Ex includes 10 multimedia examples per event, and 100 examples for 100Ex. There are 20 event classes (E021 to E040) with event names such as "Bike trick," "Dog show," and "Marriage proposal." For evaluation, we compute classification accuracy and mean average precision (mAP) metrics according to the NIST standard on the following experiments:

1. Cross-validation on 10Ex;
2. 10Ex/100Ex (train with 10Ex and test on 100Ex).

We use the number of basis vectors $K = 512$ same for all dictionaries $\mathbf{D}_{\mathrm{A}}$, $\mathbf{D}_{\mathrm{V}}$, and $\mathbf{D}_{\mathrm{AV}}$. We aggregate sparse codes around each keyframe of a training example by max pooling to form feature vectors for classification. We train linear, 1-vs-all SVM classifiers for each event whose hyperparameters are determined by 5-fold cross-validation on 10Ex. We use the INRIA SPAMS (SPArse Modeling Software) [2], VOICEBOX Speech Processing Toolkit [3], MatConvNet [12] to drive the pretrained deep CNN models, and LIBSVM [5].

3.2 Other feature learning methods for comparison

We consider other unsupervised methods to learn audio-video features for comparison. We evaluate the performance of Gaussian mixture model (GMM) and restricted Boltzmann machine (RBM) [9] under similar unimodal and multimodal settings. For GMM, we use the expectation-maximization (EM) to fit the preprocessed input vectors $\mathbf{x}_{\mathrm{A}}$, $\mathbf{x}_{\mathrm{V}}$, $\mathbf{x}_{\mathrm{AV}}$ in 512 mixtures and form GMM supervectors [4] as feature that contain posterior probabilities with respect to each Gaussian. The max-pooled GMM supervectors are applied to train linear SVMs. We adopt the shallow bimodal pretraining model by Ngiam *et al.* [7] for RBM. Activations from the hidden layer of a size 512 are also max pooled before SVM. We have applied a target sparsity of 0.1 to both GMM and RBM. For GMM, this means that a GMM supervector is left with only the highest 10% elements (posterior probabilities) while the rest being zeroed.

3.3 Results

Table 1 presents the classification accuracy and mAP performance of unimodal and multimodal sparse coding schemes. For the 10Ex/100Ex experiment, we have used the best parameter setting from the 10Ex cross-validation to test 100Ex examples. Indicated by the accuracy degradation in 10Ex/100Ex, the results from 5-fold cross-validation on 10Ex are optimistic. This is expected since hyperparameter optimization via cross-validation includes the test samples, and 10Ex is a substantially smaller dataset.

In general, we observe that the union of audio and video feature vectors perform better than using only unimodal or cross-modal features. The union of cross-modal features (Figure 3d) results better performance than joint sparse coding in 3a). We remark that the union of unimodal features has also led to better performance. The union schemes, however, double feature dimensionality (*i.e.*, from 512 to 1,024) since our union operation concatenates the two feature vectors. Joint feature vector is an economical way of combining both the audio and video features while keeping the same dimensionality as audio-only or video-only.

In Table 2, we report the mean accuracy and mAP for GMM and RBM under the union and joint feature learning schemes on the 10Ex/100Ex experiment. Our results show that sparse coding is

Table 1: Mean accuracy and mAP performance of sparse coding schemes

	Unimodal			Multimodal			
	Audio-only	Video-only	Union	Audio	Video	Joint	Union
	(Fig. 2a)	(Fig. 2b)	(Fig. 2c)	(Fig. 3b)	(Fig. 3c)	(Fig. 3a)	(Fig. 3d)
Mean accuracy (cross-val. 10Ex)	69%	86%	**89%**	75%	87%	90%	**91%**
mAP (cross-val. 10Ex)	20.0%	28.1%	**34.8%**	27.4%	33.1%	35.3%	**37.9%**
Mean accuracy (10Ex/100Ex)	56%	64%	**71%**	58%	67%	71%	**74%**
mAP (10Ex/100Ex)	17.3%	28.9%	**30.5%**	23.6%	28.0%	28.4%	**33.2%**

Table 2: Mean accuracy and mAP performance for GMM and RBM on 10Ex/100Ex

Feature learning schemes	Mean accuracy	mAP
Union of unimodal GMM features (Figure 2c)	66%	23.5%
Multimodal joint GMM feature (Figure 3a)	68%	25.2%
Union of unimodal RBM features (Figure 2c)	70%	30.1%
Multimodal joint RBM feature (Figure 3a)	72%	31.3%

better than GMM by 5–6% in accuracy and 7–8% in mAP. However, we find that the performance of RBM is on par with sparse coding. This leaves a good next step to explore further with RBM and develop joint feature learning schemes for it.

4 Conclusion

We have presented multimodal sparse coding for MED. Our approach can build joint sparse feature vectors learned from different modalities and scale to file-level descriptors suitable for training classifiers in a MED system. Using the TRECVID MED 2014 dataset, we have empirically validated our approach and achieved promising performance measured in accuracy and precision metrics recommended by the NIST standard.

We envision a fuller version of this work that will address the following. First of all, we have tested a limited set of parameterizations for each model. For example, sparse coding crucially depends on the number of basis vectors K in a dictionary, input (patch) dimension N, and sparsity parameter λ. Similarly for GMM and RBM, determining the number of mixtures or hidden units and regularization parameters among other factors would be critical. Our choice has been typical according to our media processing expertise, but not comprehensive. We plan to report a broader set of results and analyze model-specific parameter sensitivity along the effect of hyperparameter choices. Our video feature extraction uses a pretrained CNN model for detecting objects only. We are considering CNN models for detecting scenes as well. For audio, integrating with contextual detectors such as speech activity detection (SAD), language or speaker ID, and environmental noise detection are being discussed.

References

[1] 2014 TRECVID Multimedia Event Detection & Multimedia Event Recounting Tracks. http://nist.gov/itl/iad/mig/med14.cfm.

[2] SPArse Modeling Software. http://spams-devel.gforge.inria.fr/.

[3] VOICEBOX: Speech Processing Toolbox for MATLAB. http://www.ee.ic.ac.uk/hp/staff/dmb/voicebox/voicebox.html.

[4] W. M. Campbell, D. E. Sturim, and D. A. Reynolds. Support Vector Machines Using GMM Supervectors for Speaker Verification. *IEEE Signal Processing Letters*, 13(5):308–311, May 2006.

[5] C.-C. Chang and C.-J. Lin. LIBSVM: A Library for Support Vector Machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011.

[6] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional Architecture for Fast Feature Embedding. *arXiv preprint arXiv:1408.5093*, 2014.

[7] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Y. Ng. Multimodal Deep Learning. In *International Conference on Machine Learning (ICML)*, 2011.

[8] B. A. Olshausen and D. J. Field. Sparse Coding with an Overcomplete Basis Set: A Strategy Employed by V1? *Vision research*, 37(23):3311–3325, 1997.

[9] R. Salakhutdinov, A. Mnih, and G. Hinton. Restricted Boltzmann Machines for Collaborative Filtering. In *International Conference on Machine Learning (ICML)*, 2007.

[10] K. Simonyan and A. Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. *CoRR*, abs/1409.1556, 2014.

[11] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going Deeper with Convolutions. In *CVPR 2015*, 2015.

[12] A. Vedaldi and K. Lenc. MatConvNet—Convolutional Neural Networks for MATLAB. In *ACM Multimedia*, 2015.

[13] J. Yang, J. Wright, T. S. Huang, and Y. Ma. Image Super-Resolution via Sparse Representation. *IEEE Transactions on Image Processing*, 19(11):2861–2873, Nov 2010.