Deep Learning Driven Hypergraph Representation for Image-Based Emotion Recognition

Yuchi Huang
yuchi.huang@foxmail.com

Hanqing Lu
luhq@nlpr.ia.ac.cn

National Lab of Pattern Recognition
Institute of Automation, Chinese Academy of Sciences
95 Zhongguancun East Road, 100190, Beijing, China

ABSTRACT

In this paper, we proposed a bi-stage framework for image-based emotion recognition by combining the advantages of deep convolutional neural networks (D-CNN) and hypergraphs. To exploit the representational power of D-CNN, we remodeled its last hidden feature layer as the 'attribute' layer in which each hidden unit produces probabilities on a specific semantic attribute. To describe the high-order relationship among facial images, each face was assigned to various hyperedges according to the computed probabilities on different D-CNN attributes. In this way, we tackled the emotion prediction problem by a transductive learning approach, which tends to assign the same label to faces that share many incidental hyperedges (attributes), with the constraints that predicted labels of training samples should be similar to their ground truth labels. We compared the proposed approach to state-of-the-art methods and its effectiveness was demonstrated by extensive experimentation.

CCS Concepts

•Computing methodologies → Activity recognition and understanding; Object recognition;

Keywords

Emotion Recognition, Hypergraph, Deep Convolutional Networks

1. INTRODUCTION

Recognizing facial expression is a prime facet of building emotionally intelligent systems. Earlier work [20] [22] in this area is mostly based on the Facial Action Coding System (FACS) [4] proposed by Paul Ekman, in which facial expressions were decomposed into visual attributes called 'action units' (AUs), i.e. movements of face regions such as human's eyes, nose and month. These AU-based methods are very dependent on carefully hand-engineered features to ensure good performance. Another category of work for expression analysis used human's general appearance information such as facial shape and texture features to model a person's facial emotion [2] [24] [27]. Due to the success of deep learning in various computer vision problems, recent work in this category applied deep convolutional neural networks (D-CNN) as appearance-based classifier to detect action unit occurrence [5] or emotion classes [13] [19] and achieved encouraging results. The common ground of [5] [13] and [19] is that they all trained an end-to-end D-CNN system to predict AU/expression classes.

Previous work [26] [25] on visualization of D-CNN for object recognition revealed that shallow layers in deep networks learn low-level visual patterns such as edges or corners, whereas last several layers extract abstract or semantic attributes such as object parts. [13] also illustrated that in a D-CNN model trained for emotion recognition task, activations of its high-level hidden units resemble facial action units or variants of them. Inspired by these observations, in this paper we proposed a bi-stage model (as shown in Figure 1) to tackle the image-based emotion prediction problem, instead of using an end-to-end network to recognize facial expression. In the first stage we exploited the representational power of D-CNN to extract visual attributes by remodeling its last hidden feature layer (the first full connected layer) as the 'attribute' layer. Initially we trained our D-CNN which is able to predict emotion labels directly, by pre-training on a face recognition dataset and fine-tuning on the target face expression datasets. Then the classification layers were removed and the node values of the last hidden layer was exponentially normalized. We argue that each of these hidden units contains specific semantic information and represents an facial attribute critical to expression classification; the normalized value produced by each node can be taken as the probability that an input sample has the corresponding attribute.

To utilize generated attributes, in the second stage we formulated the task of emotion recognition as a hypergraph partition problem in which face images are taken as vertices and computed visual attributes are used to form hyperedges. As defined in [29], a hypergraph is a graph in which an edge (i.e. hyperedge) can connect (or contain) more than two vertices; it takes account of the high-order relationship that three or more vertices having a same attribute. For example, multiple expressive images may contain a same facial action unit. This relationship cannot be described by ordinary pairwise graphs in which an edge only links two vertices. Considering that each attribute computed in the

ICMI'16, November 12–16, 2016, Tokyo, Japan
© 2016 ACM. 978-1-4503-4556-9/16/11...$15.00
http://dx.doi.org/10.1145/2993148.2993185

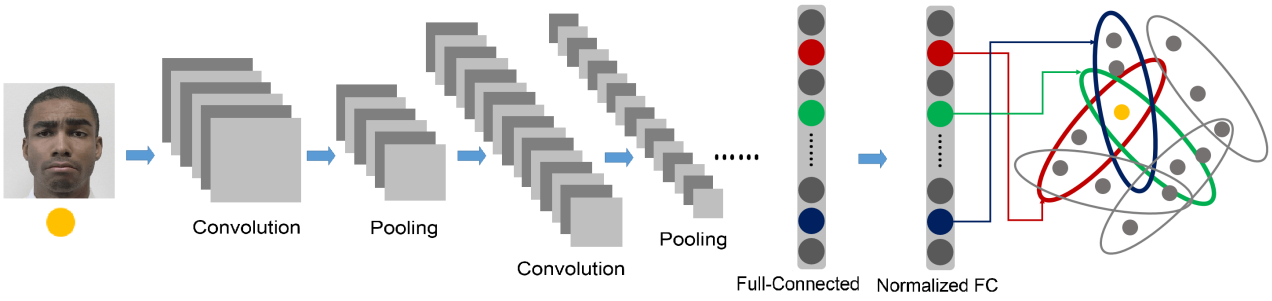

Figure 1: Our framework. The input image (denoted as a yellow vertex) is processed by the trained D-CNN. The output layers are removed and the units in the last hidden feature layer (the first full-connected layer) are exponentially normalized. Each node in this layer is taken as an 'attribute' and is used to form a hyperedge (denoted as an ellipse) in a hypergraph. In this example, three attributes (illustrated as red, green and blue nodes) are activated and the yellow vertex is assigned to three corresponding hyperedges.

first stage represents multiple images, it is more natural to describe the relationship among facial expression images as a hypergraph. Our hyeredge construction process is different from [11], in which local clusters computed from handcrafted low-level features (e.g. SIFT [16]) were used to build hyperedges. Our attributes learned from D-CNN also differ from those of [9] in that the semantic attributes of [9] were predefined by image annotators.

Based on the built hypergraph, our method generates emotion labels by a transductive inference approach, which tends to assign the same label to images that share many incidental hyperedges (attributes), with the constraints that predicted labels of training images should be similar to their ground truth labels. We compared the proposed approach to state-of-the-art methods and its effectiveness was demonstrated by extensive experimentation.

2. REVISIT TO HYPERGRAPH

Let V represent a finite set of vertices and E a family of subsets of V such that $\bigcup_{e\in E} = V$. $G = (V, E, w)$ is called a hypergraph with the vertex set V and the hyperedge set E, and each hyperedge e is assigned a positive weight $w(e)$. A traditional hypergraph can be represented by a $|V| \times |E|$ incidence matrix H_t:

$$h(v_i, e_j) = \begin{cases} 1, & \text{if } v_i \in e_j \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

This hypergraph model has proven to be beneficial to various clustering/classification tasks [1] [18] [10] [21] [9], because it can represent the information that three or more vertices have a same attribute (belong to a same hyperedge), which usual graphs can not describe. However, as illustrated in Equation 1, this structure assigns a vertex v_i to a hyperedge e_j with a binary decision, which causes some information loss. [11] proposed a fuzzy hypergraph model by taking each vertex as a 'centroid' vertex and forming a hyperedge with a centroid and its k-nearest neighbors (k-NN); $h(v_i, e_j)$ is defined as the similarity between v_i and v_j, where v_j is the centroid of e_j. Although this method avoided the hard assignment in Equation 1, such a hyperedge built by k-NN clustering does not represent a semantic attribute; it only describes the local affinity relationship computed from low-level visual features (e.g. SIFT [16]).

In this paper, we utilized D-CNN to construct semantic hyperedges and defined our hypergraph model as follows:

$$h(v_i, e_j) = \begin{cases} O(j, i), & \text{if } O(j, i) > t \\ 0, & \text{otherwise.} \end{cases} \quad (2)$$

where $O(j, i)$ denotes the probability of an image i having an attribute j, which will be explained in the next section; t is a threshold. Here each attribute is corresponding to a hyeredge. According to this formulation, v_i is 'softly' assigned to e_j based on the output probability $O(j, i)$ of trained D-CNN. For a vertex $v \in V$, its degree is defined to be $d(v) = \sum_{e\in E} w(e)h(v, e)$. For a hyperedge $e \in E$, its degree is defined as $\delta(e) = \sum_{v\in e} h(v, e)$. D_v, D_e and W are used to denote the diagonal matrices of the vertex degrees, the hyperedge degrees and the hyperedge weights respectively.

3. THE BI-STAGE FRAMEWORK

3.1 Our D-CNN Model

Network architecture. In this paper we constructed our D-CNN as in Table 1. The architecture consists of four convolutional layers containing 96, 128, 128 and 192 filters, respectively. Except those hidden units in the first layer (inputs of which are 49×49 gray-level images), all other convolutional layers use 3D convolutional filters. We adopted the rectified linear unit (ReLU) as the activation function. We used two consecutive convolutional layers (Layer 3 and Layer 4) to increase the representational power. The size of all filters on each feature map is 3×3 to capture more detailed texture variation. We used 3×3 pooling with stride 2 for all pooling layers. The last convolutional layer is followed by a full-connected layer containing 512 hidden units. We implemented this D-CNN model on the famous deep learning library CAFFE created by Jia [12].

Training Protocols. We trained our models using stochastic gradient descent with a batch size of 128 examples; we set momentum = 0.9 and weight decay = 0.001; we initialized all the weights from zero-mean Gaussian distribution with a standard deviation 0.005. We also used dropout [7] and various forms of data augmentation to regularize our networks and reduce overfitting. We applied dropout to all the convolutional layers and fully-connected layers with a prob-

Table 1: Structure of our deep convolutional neural networks. The last line shows the number of feature maps for convolutional layers and feature dimensions for full connected layers.

	1	2	3	4	5	6	7	8	9	10
Layer	Conv.	Pool.	Conv.	Conv.	Pool.	Conv.	Pool.	FC	FC	SoftM.
Kernel size	3×3	3×3	3×3	3×3	3×3	3×3	3×3	–	–	–
# of feat. maps or dims	96	–	128	128	–	192	–	512	# of Emotions	–

ability of 0.5. For data augmentation, we applied the following transformations to each image: translations, horizontal flips, rotations, scaling and pixel intensity augmentation.

Pre-training and fine-tuning. We discriminatively pre-trained our D-CNN on the Labeled Faces in the Wild dataset (LFW) [8] which contains 5,749 subjects and 13,233 cropped face images. During this phase we used 4,000 output corresponding to the number of selected training subjects till the networks converge. Then we replaced the 4000-way classification with randomly initialized N-way classification, where N is the number of emotions presented in a specific expression dataset. Stochastic gradient descent training of the D-CNN parameters was continued by using the corresponding expression dataset. We found the pre-training and followed domain-specific fine-tuning is very effective to boost performance of our task. So far we have built an end-to-end model which is able to predict emotion labels directly.

3.2 Hyperedge Construction

As shown in Table 1, the first full-connected (FC) layer has 512 hidden units and each unit produces specific semantic information beneficial to our task. We took these units as facial expression 'attributes' and used them to construct hyperedges. During this stage, the classification layers (Layer 9 and 10) were removed and the 512 'attribute' units were exponentially normalized as follows:

$$O(j,i) = \frac{e^{C(i,j)}}{\sum_{k=1}^{512} e^{C(i,k)}}, \tag{3}$$

where $C(i,j)$ denotes the value of jth unit (of the 8^{th} layer) for input image i. In this way the term of $O(j,i)$ in Equation 2 is defined and each hidden unit here represents a **hyperedge**. Input all the images (including both the training and testing samples) of a dataset into our D-CNN model and a hypergraph can be constructed accordingly. Note that t in Equation 2 was empirically set to the average of all normalized node values of the 8^{th} layer. The hyperedge weights were all set to 1 for simplicity.

4. HYPERGRAPH LEARNING

In the classical work of hypergraph learning, the normalized cost function [29] $\Omega(f)$ of a bi-partition problem is defined as follows:

$$\begin{aligned}\Omega(f) &= \frac{1}{2}\sum_{e\in E}\sum_{u,v\in e}\frac{w(e)h(u,e)h(v,e)}{\delta(e)}\left(\frac{f(u)}{\sqrt{d(u)}} - \frac{f(v)}{\sqrt{d(v)}}\right)^2 \\ &= f^T(I-\Theta)f, \end{aligned} \tag{4}$$

where the vector f is the image labels to be learned; $\Theta = D_v^{-\frac{1}{2}} H W D_e^{-1} H^T D_v^{-\frac{1}{2}}$ and $I - \Theta$ is a positive semi-definite matrix called the hypergraph Laplacian. By minimizing this cost function, images sharing many incidental hyperedges are guaranteed to obtain similar labels. In [11], it was verified that derivation in Equation 4 also holds for the fuzzy hypergraph. In an unsupervised framework, Equation 4 can be optimized by the eigenvector related to the smallest nonzero eigenvalue of $I-\Theta$. In the transductive learning setting [29], a vector y can be defined to represent the ground truth information: $y(v) = \frac{1}{|Pos|}$, if a vertex v is in the positive set Pos, $y(v) = -\frac{1}{|Neg|}$, if it is in the negative set Neg. If v is unlabeled, $y(v) = 0$. To force the assigned labels to approach the initial labeling y, a regularization term can be added to the cost function:

$$\Phi(f) = f^T(I-\Theta)f + \mu\|f-y\|^2, \tag{5}$$

where $\mu > 0$ is the regularization parameter. Above equation can be solved by differentiating $\Phi(f)$ with respect to f:

$$f = (1-\gamma)(I-\gamma\Theta)^{-1}y, \tag{6}$$

where $\gamma = \frac{1}{1+\mu}$. This is equivalent to solving the linear system $((1+\mu)I - \Theta)\, f = \mu y$.

In our application, we constructed a hypergraph for all images with N different initial labeling vectors y, where N is the number of emotions present in a specific dataset. In each of these labeling vectors a positive/negative label denotes the presence/non-presence of one expression on a training sample; an initial label 0 denotes that the corresponding image is in the testing set. With N different ys above linear system will be solved for N times and the final learned emotion of a test image is decided by the maximum value of N predicted scores.

5. EXPERIMENTS

We chose two representative facial expression datasets in our experiments: the dataset for Facial Expression Recognition Challenge 2013 (FER2013) [3] and the extended Cohn-Kanade database (CK+) [17]. We compared our proposed method to two baseline approaches: 1) D-CNN + SVM in which normalized output features of the 8^th layer are fed to a linear SVM classifier and 2) D-CNN + N-way Softmax in which N emotion labels are predicted directly (by using the configuration of N-way expression classification in the last two layers). We also compared to state-of-the-art deep learning based methods on each dataset to show the advantage of our approach.

For the parameter γ in Equation 6, we followed the original work of Zhou [28] and fix it as 0.1 for the best performance. Other parameters were directly computed from

Figure 2: Training data of FER2013. Each row consists of faces of the same expression: starting from the first row: Angry, Disgust, Fear, Happy, Sad, Surprise, Neutral.

experimental data.

5.1 Performance on FER2013

FER2013 consists of 28,709 48×48 training, 3,589 validation and 3,589 testing images of faces under 7 different types of emotions: Angry, Disgust, Fear, Happy, Sad, Surprise, Neutral. This dataset was used for ICML 2013 Challenge for Representation Learning. As shown in Figure 2, the face images of this dataset are not frontalized. So without frontalization we only rescaled the detected LFW faces to 49×49 and used them for pre-training of our D-CNN model mentioned in Section 3.1. The training and validation images of FER2013 were padded to 49×49 and employed for fine-tuning. This pre-training and fine-tuning was performed 10 times independently to generate the average accuracies in Table 2.

To form a hypergraph as introduced above, all the images including the testing samples were input to our D-CNN model. The training and validation images with 7 emotion labels were used as the 'training' set to get 7 different initial labeling vectors. As illustrated in Table 2, 1) based on the same D-CNN model, our hypergraph learning approach outperforms D-CNN + SVM and D-CNN + Softmax by more than 3%; 2) results of all these three approaches are better than that of DLSVM [19], the winning method of ICML 2013 Challenge for Representation Learning. This illustrated the effectiveness of pre-training and fine-tuning techniques used to build our D-CNN model.

5.2 Performance on CK+

The extended Cohn-Kanade database (CK+) [17] contains 593 sequences across 123 subjects. All sequences are from the neutral face to the peak expression. ONLY 327 of the 593 sequences have emotion labels and each of which is assigned one of 7 expressions: Anger, Contempt, Disgust, Fear, Happy, Sad and Surprise. To build the expression dataset for comparison of results, we followed the protocol of [14] [13] to extract last three frames of 327 sequences with emotion labels. We also followed [14] [13] to use the first frame of each sequence as a neutral expression. In this way we formed a set of 1308 images with 8 different emotion labels. A face detection algorithm [23] was applied on these images and the cropped faces were all rescaled into 49×49 images.

Table 2: Performance comparison on FER2013. DLSVM [19] is the winning method of ICML 2013 Challenge for Representation Learning.

Method	Accuracy
DLSVM [19]	71.2%
D-CNN + SVM	73.5% ± 1.8 %
D-CNN + Softmax	73.7% ± 1.7 %
D-CNN + Hypergraph	**76.9% ± 1.4%**

Table 3: Performance comparison on CK+.

Method	Accuracy
AUDN [15]	93.7%
Zero-bias CNN + AD [13]	96.4% ± 3.1%
D-CNN + SVM	96.5% ± 2.5%
D-CNN + Softmax	95.7% ± 1.9%
D-CNN + Hypergraph	**97.3% ± 2.3%**

To train our deep neural networks, at first we pre-trained our model as described in Section 3.1. Since detected faces in CK+ images are all frontal, we used frontalized LFW data provided by [6] for the pre-training. To fulfill the fine tuning and the followed hypergraph learning, we split 1308 images into 10 subject independent subsets in the manner presented by [14] and performed 10 fold cross-validation. Each time, images from the training set (9 out of 10 subsets) were employed to fine-tune the pre-trained D-CNN model. Data augmentation techniques were utilized to optimize the tuning results. In this way we have built a D-CNN model which is able to predict the occurrence of an emotion directly; the average prediction accuracy of 10 fold experiments is 95.7%.

To perform hypergraph learning, in each experiment 8 different ys w.r.t. 8 emotions were used and the final learned expression was decided by the maximum value of 8 predicted scores. By using the same D-CNN model, our hypergraph learning approach outperforms D-CNN + SVM and D-CNN + N-way Softmax by 0.8% and 1.6%, respectively. Results of all these three approaches are also better than those of reported by start-of-the-art methods [15] and [13].

6. CONCLUSION

We introduced a transductive learning framework for image-based emotion recognition, in which fuzzy hypergraph was used to represent the relevance relationship among faces. Using last hidden feature layer's nodes of our D-CNN model as semantic cues, we took each image as a vertex and formed hyperedges according to those deep learning driven semantic attributes. In this way, the task of facial expression classification was converted to a transductive learning problem which can be solved by the hypergraph partition algorithm. The effectiveness of our proposed method was demonstrated by extensive experimentation on two popular databases.

7. REFERENCES

[1] S. Agarwal, J. Lim, L. Zelnik Manor, P. Perona, D. Kriegman, and S. Belongie. Beyond pairwise clustering. In *CVPR'05*.

[2] M. S. Bartlett, G. Littlewort, M. Frank, C. Lainscsek, I. Fasel, and J. Movellan. Recognizing facial expression: Machine learning and application to spontaneous behavior.

[3] P.-L. Carrier and A. Courville. Facial expression recognition dataset. In *ICML 2013 Challenges in Representation Learning*, 2013.

[4] P. Ekman and W. Friesen. Facial action coding system: A technique for the measurement of facial movement. *Consulting Psychologists Press*, 1978.

[5] A. Gudi, H. E. Tasli, T. M. den Uyl, and A. Maroulis. Deep learning based FACS action unit occurrence and intensity estimation. In *11th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition, FG 2015, Ljubljana, Slovenia, May 4-8, 2015*, 2015.

[6] T. Hassner, S. Harel, E. Paz, and R. Enbar. Effective face frontalization in unconstrained images. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[7] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *CoRR*, 2012.

[8] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst, October 2007.

[9] S. Huang, M. Elhoseiny, A. M. Elgammal, and D. Yang. Learning hypergraph-regularized attribute predictors. *CVPR*, 2015.

[10] Y. Huang, Q. Liu, and D. Metaxas. Video object segmentation by hypergraph cut. CVPR'09.

[11] Y. Huang, Q. Liu, S. Zhang, and D. N. Metaxas. Image retrieval via probabilistic hypergraph ranking. In *CVPR*, pages 3376–3383. IEEE Computer Society, 2010.

[12] 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.

[13] P. Khorrami, T. L. Paine, and T. S. Huang. Do deep neural networks learn facial action units when doing expression recognition? *CoRR*, abs/1510.02969, 2015.

[14] M. Liu, S. Li, S. Shan, and X. Chen. Au-aware deep networks for facial expression recognition. In *10th IEEE International Conf. on Automatic Face and Gesture Recognition, FG*, 2013.

[15] M. Liu, S. Li, S. Shan, and X. Chen. Au-inspired deep networks for facial expression feature learning. *Neurocomputing*, 159:126–136, 2015.

[16] D. Lowe. Object recognition from local scale-invariant features. In *ICCV'09*.

[17] P. Lucey, J. F. Cohn, T. Kanade, J. Saragih, Z. Ambadar, and I. Matthews. The extended cohn-kande dataset: A complete facial expression dataset for action unit and emotion-specified expression. In *CVPR4HB*, 2010.

[18] L. Sun, S. Ji, and J. Ye. Hypergraph spectral learning for multi-label classification. In *SIG KDD '08*.

[19] Y. Tang. Deep learning with linear support vector machines. *Workshop on Representational Learning, ICML*, 2013.

[20] Y.-l. Tian, T. Kanade, and J. F. Cohn. Recognizing action units for facial expression analysis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 23(2):97–115, Feb. 2001.

[21] Z. Tian, T. Hwang, and R. Kuang. A hypergraph-based learning algorithm for classifying gene expression and array cgh data with prior knowledge. *Bioinformatics*, July 2009.

[22] Y. Tong, W. Liao, and Q. Ji. Facial action unit recognition by exploiting their dynamic and semantic relationships. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(10):1683–1699, 2007.

[23] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *CVPR*, pages 511–518, 2001.

[24] J. Whitehill and C. W. Omlin. Haar features for facs au recognition. In *FG*, pages 97–101. IEEE Computer Society, 2006.

[25] J. Yosinski, J. Clune, A. Nguyen, T. Fuchs, and H. Lipson. Understanding neural networks through deep visualization. In *Deep Learning Workshop, International Conference on Machine Learning (ICML)*, 2015.

[26] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *ECCV 2014 - 13th European Conference on Computer Vision*.

[27] G. Zhao and M. Pietikainen. Dynamic texture recognition using local binary patterns with an application to facial expressions. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(6):915–928, 2007.

[28] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schökopf. Learning with local and global consistency. In *NIPS'03*.

[29] D. Zhou, J. Huang, and B. Schökopf. Learning with hypergraphs: Clustering, classification, and embedding. In *NIPS'06*.