Deep Action Classification via Matrix Completion

Sushma Bomma and Neil M Robertson
Institute of Sensors, Signals and Systems
Heriot-Watt University, United Kingdom
Email: (sb6, N.M.Robertson)@hw.ac.uk

***Abstract*—Matrix completion is the task of predicting unknown or missing entries in a data matrix. The estimation of the missing entries is based on the assumption that the underlying matrix is a low rank one. Deep learning has evolved as an efficient tool for feature extraction in many large-scale image based applications. Exploiting the techniques from both domains, we propose a novel solution to the problem of simultaneous classification of actions from multiple test videos with deep features using matrix completion methods. Learned features from a convolutional neural network and corresponding labels from data are concatenated to form a big matrix with unknown or missing entries in the place of test data labels. Convex rank minimization algorithms are used to complete this matrix. The proposed method achieves stable performance even in situations with more than 50% of features and labels missing.**

I. Introduction

Real world data is rarely perfect or clean. This can be due to human error, equipment failure, system generated error etc. This requires learning with missing or incomplete data in many applications. Missing data leads to missing features. In supervised learning another issue is the availability of labelled data. Labelling is manual and laborious for large datasets. With only some of the data labelled, the problem would be semi-supervised learning, where a classifier is trained on labelled and unlabelled training data. We therefore seek a classifier which is robust to deficiencies in features and labels. To this end we propose a solution using the matrix completion framework addressing the issues of missing features and labels applied to the problem of action recognition. Human action recognition is useful in assisted living, security surveillance, human-computer interfaces, patient-monitoring systems etc. To recognize ongoing activities from an unknown video is a challenging task due to unavoidable occlusions, view-point variations of cameras, anthropometric differences and so on. Despite these issues, past decade has made remarkable progress in this area [1] [2]. In surveillance applications, it is required to simultaneously recognize or classify multiple actions. This was addressed as joint classification of actions with matrix completion [3]. In that, the missing entries were only test data labels. The problem when some of the features missing besides test labels is not addressed. In this paper, we propose robust joint action classification within matrix completion setting, dealing with missing features or labels or both from training data together with unknown test data labels. In this context we show that learned features from a convolutional neural network perform better than the state of the art hand-crafted features. Brief background on matrix completion and deep learning is given in Section I-A and Section I-B.

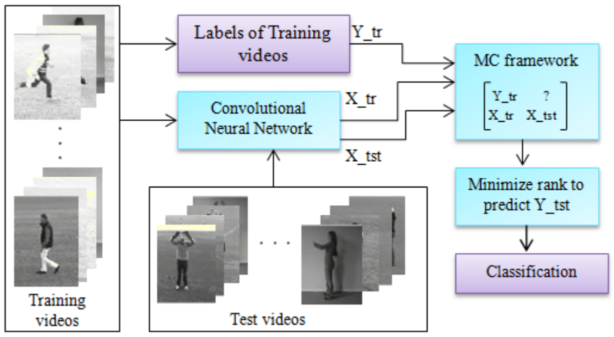
Labels of Training videos
Y_tr
MC framework
Y_tr ?
X_tr X_tst
Convolutional Neural Network
X_tr
X_tst
Minimize rank to predict Y_tst
Classification
Training videos
Test videos

Fig. 1. Proposed approach : A convolutional neural network is trained to learn class specific features from given training videos. The trained network is used to extract features X_tr and X_tst from train and test videos. Label vectors Y_tr corresponding to the training videos are generated. Features and labels are arranged as shown in Matrix completion framework with zeroes replaced in the place of unknown entries. Classification is achieved by completing the matrix using rank minimization techniques.

A. *Matrix Completion*

Matrix completion (MC) is the problem of estimating unknown entries of a matrix generated by partial observations of data [4]. MC is associated with popularly known *Netflix* challenge [5] to come up with a *recommender system* using highly insufficient observed ratings. The solution to matrix completion problem relies on the assumption that the underlying matrix is a low-rank matrix. If Z is the incomplete matrix with some observed entries A_{ij} where (i, j) are specified by a set Ω, then it can be recovered by solving the problem of rank minimization as shown in Equation (1). The rank function is non-convex and is difficult to optimize.

$$\begin{aligned} \text{minimize} \quad & \operatorname{rank}(\mathbf{Z}) \\ \text{subject to} \quad & \mathbf{Z}_{ij} = \mathbf{A}_{ij} \ \forall \ (i,j) \in \Omega \end{aligned} \tag{1}$$

Replacing the rank function by nuclear norm [6], its convex equivalent Equation (2) is obtained.

$$\begin{aligned} \text{minimize} \quad & \| \mathbf{Z} \|_* \\ \text{subject to} \quad & \mathbf{Z}_{ij} = \mathbf{A}_{ij} \ \forall \ (i,j) \in \Omega \end{aligned} \tag{2}$$

Rank function is the count of singular values of the matrix while $\| \ \|_*$ is the nuclear norm which is the sum of singular values. These are analogous to l_0 and l_1 norms respectively. Linear classification requires finding a decision boundary defined by the parameters $[\mathbf{W^T}, \mathbf{b}]$ such that $\mathbf{y}_i = \mathbf{W^T}\mathbf{x}_i + \mathbf{b}$

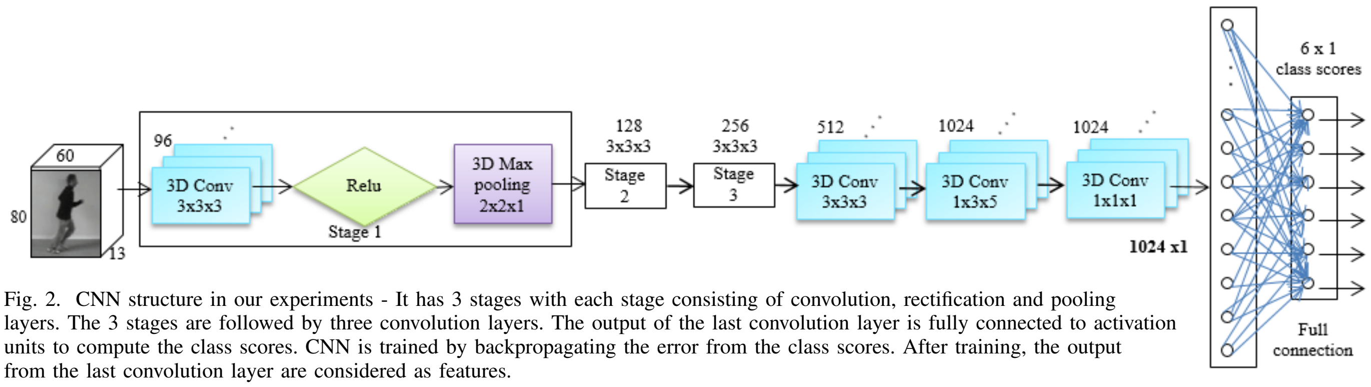

Fig. 2. CNN structure in our experiments - It has 3 stages with each stage consisting of convolution, rectification and pooling layers. The 3 stages are followed by three convolution layers. The output of the last convolution layer is fully connected to activation units to compute the class scores. CNN is trained by backpropagating the error from the class scores. After training, the output from the last convolution layer are considered as features.

where $(\mathbf{x}_i, \mathbf{y}_i)$ is the feature and label pair from the training data for $i = 1, 2, \ldots N_{tr}$. This parameter set is then used for classifying test data. The matrix formed by the concatenation of labels and features will be of low rank based on the linear dependence of labels on features. Transduction classification with matrix completion was introduced by Goldberg et al in [7]. Mathematically, let $\mathrm{X_{tr}}$ be the feature matrix with columns representing each training sample and $\mathrm{Y_{tr}}$ be the label matrix whose columns are labels of the corresponding training samples. Similarly let $\mathrm{X_{tst}}$ and $\mathrm{Y_{tst}}$ be those pertaining to test data. Then combining all, leads to a low rank matrix, say Z as shown in (3).

$$\mathbf{Z} = \begin{bmatrix} \mathbf{Y}_{tr} & \mathbf{Y}_{tst} \\ \mathbf{X}_{tr} & \mathbf{X}_{tst} \end{bmatrix} \Rightarrow \begin{bmatrix} \mathbf{Y}_{tr} & \mathbf{?} \\ \mathbf{X}_{tr} & \mathbf{X}_{tst} \end{bmatrix} \tag{3}$$

With the completely unknown labels and probably corrupted or missing training features and corresponding labels, the matrix Z is incomplete. The classification problem can now be cast as rank minimization problem as in (2) and can be solved using techniques for rank minimization. Since features and labels span different spaces, Equation (2) is further constrained with *least-squares* feature loss and *logistic* label loss. Following this, matrix completion was applied to multi-label classification tasks. In [8] and [9], matrix completion is used for multi-label image classification. This is extended to handle images with multiple viewpoints in [10]. Also relation extraction problem of natural texts in [11] is solved using matrix completion.

B. Deep Learning and Convolutional Neural Networks

A sub-field of machine learning which has gained immense attention recently is *deep learning*. Deep learning methods tend to *automatically* learn powerful non-linear representations of the data in a hierarchical manner unlike conventional hand-crafted methods. These methods have shown to beat the state-of-the-art in text [12], audio [13] [14] and image applications [15]. There are generative models such as Deep Belief Nets (DBNs) which learn features from input in an unsupervised manner [16] and supervised discriminative models such as Convolutional Neural Networks (CNNs) [17]. CNNs on large datasets like Imagenet [18] have shown to outperform than DBNs. CNNs are more advanced extensions of fully connected neural networks which learn features from input at multiple levels in a supervised manner i.e. from labelled data. The learning process is achieved through online backpropagation [12] with gradient descent optimization.
The supervised learning of CNNs results in discriminative features such that they cluster according to the classes. This makes CNNs well suited for classification and recognition tasks. A basic CNN is comprised of one or more blocks of convolution, rectification, pooling, operations. Convolutional layer consists of a filter bank, with trainable parameters or weights. Each filter adapts to a specific set of features from the input in the learning process and the output of the filter is known as *feature map*. These feature maps are then applied to a non-linear operator through rectification layer. The rectification or *Relu* is more biologically inspired to mimic human brain's neurons' activity, to fire on certain inputs only. Pooling is a *downsampling* operator producing translational invariance to the features. Typically *average* or *mean* pooling and *max* pooling is used in classification or recognition tasks.

II. Deep Action Classification with Matrix Completion

Matrix completion (MC) and deep learning techniques have been successfully applied to various vision problems dealing with images. Fusing both techniques, we propose robust classification of multiple actions, each represented with learned deep features, within the matrix completion framework. We focus on *multi-class classification* of actions from videos rather than *multi-label classification*. Figure 1 illustrates our proposed approach. We formalize the classification problem as matrix completion problem next. Under the setting where the unknown entries are only the test labels, the problem can be formulated as Equation (4) adapted from [9]. Let $\mathbf{Z_X} = [\mathbf{X}_{tr} \quad \mathbf{X}_{tst}]$ be the feature sub-matrix, $\mathbf{Z_Y} = [\mathbf{Y}_{tr} \quad \mathbf{Y}_{tst}]$ be the label sub-matrix, $\mathbf{Z_1} = \mathbf{1}^{\mathrm{T}}$ is used to handle bias and thus $\mathbf{Z} = [\mathbf{Z_Y}; \mathbf{Z_X}; \mathbf{Z_1}]$. The parameters μ and λ are positive weights for better adaptation of features and labels and γ is a regularizer used to smooth the log loss. For the case where some of the features of train and test data and some of the labels of train data are missing, the summations in (5) would be over the observed entries only. This problem is solved using *Fixed Point Continuation (FPC)* method from [19]. Mainly there are two alternating steps of gradient computation and shrinkage operation.

$$\begin{aligned} \text{minimize} \quad & \mu\|\mathbf{Z}\|_* + l_X(\mathbf{Z_X}) + \lambda\, l_Y(\mathbf{Z_Y}) \\ \text{subject to} \quad & \mathbf{Z_X} \geq \mathbf{0} \text{ and } \mathbf{Z_1} = \mathbf{1}^{\mathrm{T}} \end{aligned} \tag{4}$$

where

$$\begin{aligned} l_X(\mathbf{Z_X}) &= \sum_{ij \in \mathbf{Z_X}} (\mathrm{z}_{ij} - \mathrm{z}_{0ij})^2 \\ l_Y(\mathbf{Z_Y}) &= \sum_{ij \in \mathbf{Y}_{train}} \frac{1}{\gamma} log(1 + exp(-\gamma \mathrm{z}_{ij} \mathrm{z}_{0ij})) \end{aligned} \tag{5}$$

To speed-up the algorithm, these two steps are iterated until a specified error tolerance for a sequence of fixed values of a parameter (μ, nuclear norm weight in this case). For every parameter value, the initial objective value is the final value obtained from the previous parameter iteration. The gradient computation of the matrix is computed in two steps. Since the loss on features and labels are modelled using different functions, the gradients for each part i.e. feature part and label part are computed independently and then combined to get the gradient of the complete matrix. Matrix completion relies on the assumption that the underlying matrix is a low rank matrix implying that either the rows or columns span a low dimensional subspace. In multi-label classification with matrix completion as in [7], [9], [10], [11], the correlation between labels aids the underlying low rank structure. For multi-class classification, the labels are distinct. In fact if labels are in $\{0, 1\}$, then they are orthogonal. In such cases, the low rank structure predominantly relies on features. For this reason, we train a CNN to learn features which are common within a specific class and distinct from other classes. The trained CNN will output features from the samples, which will be clustered according to the classes. We refer to these learned features as *deep features*. To extract spatio-temporal features, convolution and pooling must be 3 dimensional as in [20] and [21].

III. Experiments

We evaluate our approach in two ways (1) Compare the performance of MC classification with deep features from CNN and with state of the art hand-crafted features (2) Compare the performance of MC classification with missing features and labels to linear SVM. We use KTH dataset [22] in our experiments. Figure 3 shows the snapshot of this dataset. It is comprised of videos from 6 action classes - boxing, hand-clapping, handwaving, jogging, running, walking - recorded in 4 scenarios - s1(outdoors), s2(outdoors with scale variation), s3(outdoors with different clothing), s4(indoors). There are 25 persons performing the actions under each setting. Each class has around 400 sequences and 2391 sequences on the whole. As per [22], videos of 16 persons are used for training and the remaining 9 for testing. For our experiments, the frames are reduced to 60×80 from 120×160 and a sequence of 13 frames is considered. So a video is represented as $60 \times 80 \times 13$ spatio-temporal cube. Apart from the resizing, no other pre-processing like human centric bounding box extraction and local contrast normalization as in [21] or foreground and optical flow extraction as in [20] are made in our experiments.

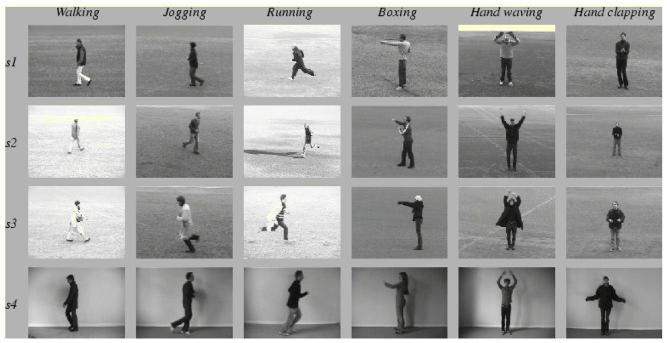

Fig. 3. A snapshot of KTH dataset showing six actions in a row for each of the four scenarios.

A. CNN training and deep features

The CNN used in this work is shown in Figure 2. This is comprised of 3 stages of convolution, rectification and pooling operations. Stage 1 has a filter bank of 96 with the size of each convolution filter being $(3 \times 3 \times 3)$. Stage 2 and 3 result in 128 and 256 feature maps respectively with same size filters. The three stages are followed by two convolution layers of filter sizes $(3 \times 3 \times 3)$, $(1 \times 3 \times 3)$ with 512 and 1024 feature maps. Following which there are two fully connected layers of 1024 and 6 (for each class of KTH dataset) neurons. The pooling is max-pooling with a window size is $(2 \times 2 \times 1)$ in all the 3 stages. The convolution stride for all convolution layers is fixed as 1 spatially and temporally. And the pooling stride is 2 spatially and 1 temporally. This CNN is built with pre-defined layers in Matconvnet toolbox [23]. 3D convolution and pooling layers from [24] which are compatible with Matconvnet toolbox are adapted for our experiments. The training parameters weight decay, momentum are set to 0.005 and 0.9 respectively. Initially the learning rate η is chosen as $1e-4$. For training purposes, we extend the dataset by including horizontal and vertical flipped data of the original version. The network is trained on the training data i.e. video cubes of 16 persons using online backpropagation with weight sharing and momentum [12]. The training is stopped after 35 epochs. The output from the second last layer 1024×1 is considered as the feature vector representing each video.

B. Dense trajectory features

We also extract dense trajectory feature descriptors proposed by Wang et.al. in [25]. These are state-of-the-art hand-crafted features in action recognition. For each video, trajectories are tracked based on optical flow at densely sampled pixels. For each trajectory, we compute only motion boundary histogram (MBH) descriptors. Each descriptor is 192 with 96 in x and 96 in y directions. This is reduced by two using PCA. A Gaussian Mixture Model (GMM) with $K = 256$ is learnt on the PCA reduced descriptors from training data. Each of these descriptors are fisher-vector encoded using the GMM. The dimension of this encoded MBH descriptor is 49152. We have further reduced the dimension with *Random Projection* to 3072.

We first construct sub-matrix $\mathbf{Z_X}$ using the features of train

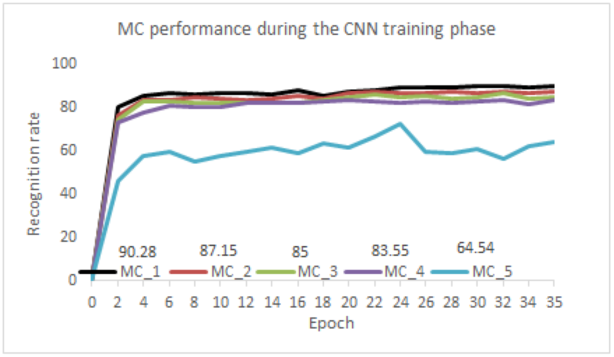

Fig. 4. Comparison of CNN performance and MC performance during the entire training phase of CNN.

data and test data respectively. Training data features are arranged class-wise from all classes and classification for each class test data is computed separately to avoid memory issues with Matlab. We chose binary labels in $\{0, 1\}$ for each class with each label being a 6×1 binary vector. So we have $\mathbf{Z_Y} = [\mathbf{Y}_{\text{tr}} \;\; \mathbf{Y}_{\text{tst}}]$ with zero vectors in the place of $\mathbf{Y}_{\text{tst}}$ and $\mathbf{Z} = [\mathbf{Z_Y}; \mathbf{Z_X}; \mathbf{Z_1}]$.

C. Results

Before doing our comparisons we have checked the performance of MC classification during the entire training phase of CNN. For every 2 epochs, the performance of MC is recorded and it is observed that MC method closely follows CNN throughout for different percentages of missing features and labels. Figure 4 shows the results these as curves labelled MC_1, MC_2 ,MC_3, MC_4 and MC_5. The case where only labels of test data are missing is MC_1. To check the robustness of the MC classification, we create random masks applied in an overlapping manner to observe only some percentage of the available data. Using these masks, we compute $\mathbf{Z_2}$, $\mathbf{Z_3}$, $\mathbf{Z_4}$, $\mathbf{Z_5}$ which are obtained by randomly missing 30%, 50%, 70% and 90% data respectively of the actual $[\mathbf{Z_Y}; \mathbf{Z_X}]$. MC_2 corresponds to $\mathbf{Z_2}$, MC_3 corresponds to $\mathbf{Z_2}$ and so on. It is seen that even with 50% missing data, the recognition rate does not deteriorate much validating the robustness of the MC method of classification.

Our first experiment justifies the choice of deep features for MC classification. We test the performance of MC method of classification with deep features and dense trajectory features. We construct the matrix $\mathbf{Z}$ in the same way as mentioned in the previous experiment with dense trajectory features from each video. The results are summarized in Figure 5 showing the performance of deep features (referred to as Deep_feat) and dense trajectory features (referred to as Dense_feat) against percentage of missing entries. It is clearly seen that the deep features are more stable than the dense trajectory features in the context of MC method. With only test labels unknown, deep features and dense features perform equally well. As the percentage of the observed data decreases, performance with dense features degrades faster than deep features.

For our second experiment, a linear SVM is trained separately for different percentages of the missing data. In this case, it is

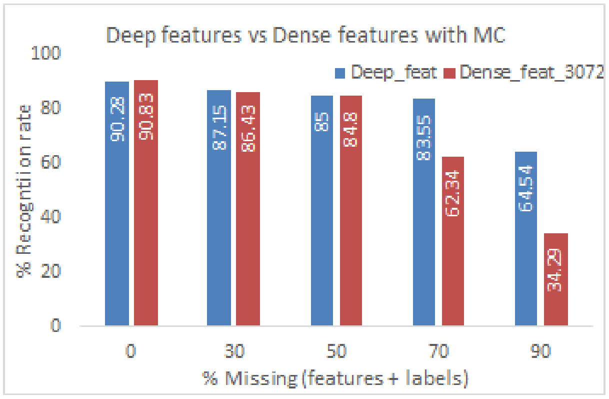

Fig. 5. Performance comparison of deep features and dense features with increasing percentages of missing entries.

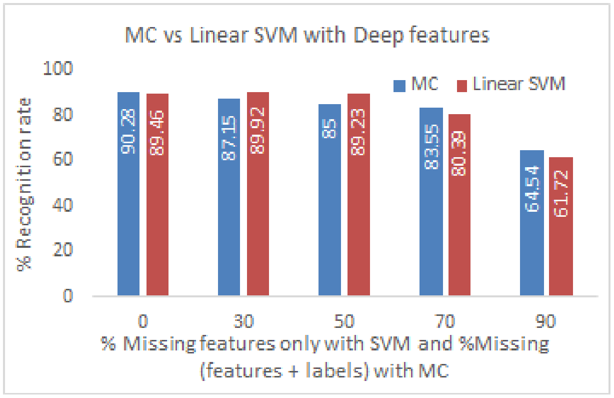

Fig. 6. Performance comparison of MC and Linear SVM using deep features with increasing percentages of missing entries.

only the features which are missing in training the SVM. As can be seen from Figure 6, without any training MC performs better than SVM in most cases. The best recognition rate with our approach is **90.28%** where we jointly classify multiple test samples in a class simultaneously. This is comparable state of the art of 92.17% in [21] or 90.20% in [20] on the same dataset where CNN features are used. But we have not used any further training as in [21] where a recurrent neural network is trained on features from CNN or used any hand crafted inputs as in [20] where the CNN is trained on image gradients, optical flow along with input frames.

IV. Conclusion

In this paper we proposed matrix completion as classification tool with deep features from a convolutional neural network. From the experiments it is observed that the performance of MC with deep features is more consistent than dense features. The performance deteriorates only under extreme conditions where more than 70% of the data is missing. This proves that the data matrix with deep features preserves the low-rank structure. In future, our work is directed towards (1) learning features from more deeper architectures so as to improve the recognition rate in matrix completion scenario and (2) validate on more complicated datasets like Youtube and UCF.

References

[1] J. Aggarwal and M. Ryoo, "Human activity analysis: A review," *ACM Comput. Surv.*, pp. 16:1–16:43, 2011.

[2] R. Poppe, "A survey on vision-based human action recognition," *Image Vision Comput.*, pp. 976–990, 2010.

[3] S. Bomma and N. Robertson, "Joint classification of actions with matrix completion," in *Image Processing (ICIP), 2015 IEEE International Conference on*, Sept 2015, pp. 2766–2770.

[4] E. J. Candes and B. Recht, "Exact matrix completion via convex optimization." *Foundations of Computational Mathematics*, no. 6, pp. 717–772, 2009.

[5] J. Bennett, S. Lanning, and N. Netflix, "The netflix prize," in *In KDD Cup and Workshop in conjunction with KDD*, 2007.

[6] B. Recht, M. Fazel, and P. A. Parrilo, "Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization," *SIAM review*, vol. 52, no. 3, pp. 471–501, 2010.

[7] A. B. Goldberg, X. Zhu, B. Recht, J.-M. Xu, and R. D. Nowak, "Transduction with matrix completion: Three birds with one stone." in *NIPS*. Curran Associates, Inc., 2010, pp. 757–765.

[8] R. S. Cabral, F. D. la Torre, J. P. Costeira, and A. Bernardino, "Matrix completion for multi-label image classification." in *NIPS*, 2011, pp. 190–198.

[9] R. S. Cabral, F. De la Torre, J. P. Costeira, and A. Bernardino, "Matrix completion for weakly-supervised multi-label image classification," *IEEE Transactions Pattern Analysis and Machine Intelligence (PAMI)*, 2014.

[10] M. Liu, Y. Luo, D. Tao, C. Xu, and Y. Wen, "Low-rank multi-view learning in matrix completion for multi-label image classification." in *AAAI*. AAAI Press, 2015, pp. 2778–2784.

[11] M. Fan, D. Zhao, Q. Zhou, Z. Liu, T. F. Zheng, and E. Y. Chang, "Distant supervision for relation extraction with matrix completion." Baltimore, Maryland: Association for Computational Linguistics, June 2014, pp. 839–849.

[12] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," in *Proceedings of the IEEE*, vol. 86, no. 11, 1998, pp. 2278–2324.

[13] H. Lee, P. T. Pham, Y. Largman, and A. Y. Ng, "Unsupervised feature learning for audio classification using convolutional deep belief networks." in *NIPS*. Curran Associates, Inc., 2009, pp. 1096–1104.

[14] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury, "Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups," *Signal Processing Magazine, IEEE*, no. 6, pp. 82–97, Nov. 2012.

[15] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks." in *NIPS*, 2012, pp. 1106–1114.

[16] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, no. 7, pp. 1527–1554, July 2006.

[17] 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*. IEEE, 2015, pp. 1–9.

[18] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, June 2009, pp. 248–255.

[19] S. Ma, D. Goldfarb, and L. Chen, "Fixed point and bregman iterative methods for matrix rank minimization." *Math. Program.*, vol. 128, no. 1-2, pp. 321–353, 2011.

[20] S. Ji, W. Xu, M. Yang, and K. Yu, "3d convolutional neural networks for human action recognition." Omnipress, 2010, pp. 495–502.

[21] M. Baccouche, F. Mamalet, C. Wolf, C. Garcia, and A. Baskurt, "A.: Sequential deep learning for human action recognition," 2011.

[22] C. Schuldt, I. Laptev, and B. Caputo, "Recognizing human actions: a local svm approach," in *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, vol. 3. IEEE, 2004, pp. 32–36.

[23] A. Vedaldi and K. Lenc, "Matconvnet – convolutional neural networks for matlab," 2015.

[24] P. Sun, "3d convolution and pooling operation - mexconv3d," 2015. [Online]. Available: https://github.com/pengsun/MexConv3D

[25] H. Wang, A. Klaser, C. Schmid, and C.-L. Liu, "Action recognition by dense trajectories," in *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition*, ser. CVPR '11. Washington, DC, USA: IEEE Computer Society, 2011, pp. 3169–3176. [Online]. Available: http://dx.doi.org/10.1109/CVPR.2011.5995407