HEp-2 Cell Image Classification with Deep Convolutional Neural Networks

Zhimin Gao, Lei Wang, *Senior Member, IEEE,* Luping Zhou, *Senior Member, IEEE,* and Jianjia Zhang

***Abstract*—Efficient Human Epithelial-2 (HEp-2) cell image classification can facilitate the diagnosis of many autoimmune diseases. This paper presents an automatic framework for this classification task, by utilizing the deep convolutional neural networks (CNNs) which have recently attracted intensive attention in visual recognition. This paper elaborates the important components of this framework, discusses multiple key factors that impact the efficiency of training a deep CNN, and systematically compares this framework with the well-established image classification models in the literature. Experiments on benchmark datasets show that i) the proposed framework can effectively outperform existing models by properly applying data augmentation; ii) our CNN-based framework demonstrates excellent generalization capability across different datasets, which is highly desirable for classification under varying laboratory settings. Our system is ranked high in the cell image classification competition hosted by ICPR 2014.**

***Index Terms*—Indirect immunofluorescence, staining patterns classification, deep convolutional neural networks (CNNs).**

I. Introduction

INDIRECT immunofluorescence (IIF) on Human Epithelial-2 (HEp-2) cells is a recommended methodology to diagnose autoimmune diseases [1]. However, manual analysis of IIF images leads to crucial limitations, such as the subjectivity of result, the inconsistence across laboratories, and the low efficiency in processing a large number of cell images [2], [3]. To improve this situation, automatic and reliable cell images classification has become an active research topic.

Many methods have been recently proposed for this topic, especially during the HEp-2 cell classification competitions [3], [4], [5]. Most of them treat feature extraction and classification as two separate stages. For the former, a variety of hand-crafted features are adopted, including local binary pattern (LBP) [6], [7], [8], scale-invariant feature transform (SIFT) [9], histogram of oriented gradients [10], discrete cosine transform, and the statistical features like gray-level co-occurrence matrix [11] and gray-level size zone matrix [12]. For the latter, nearest-neighbour classifier, boosting, support vector machines (SVM) and multiple kernel SVM have been employed [13]. As a result, the performance of these classifiers relies highly on the appropriateness of the empirically chosen hand-crafted features. Moreover, because features and classifier are treated separately, they cannot work together to maximally identify and retain discriminative information.

Very recently, deep convolutional neural networks (CNNs) have consistently achieved outstanding performance on generic visual recognition tasks [14] and this has revived extensive research interest in CNN-based classification model [15]. The CNNs consist of multi-stage processing of an input image to extract hierarchical and high-level feature representations. Many hand-crafted features and the corresponding classification pipelines can be regarded as an approximation to or a special case of the CNNs, by sharing some basic building blocks. Nevertheless, these features and pipelines have to be carefully designed and integrated in order to preserve discriminative information. The excellent performance achieved by deep CNNs on generic visual recognition and the high demand for full automation of HEp-2 cell image classification motivate us to research the CNNs for this classification task.

To this end, we propose an automatic feature extraction and classification framework for HEp-2 staining patterns based on deep CNNs [16]. This framework extracts features from the raw pixels of cell images and avoids using hand-crafted features. Feature representations for each kind of staining patterns are learnt and optimized via training the multi-layer network. Also, the classification layer is jointly learnt with this network to predict the probability of a cell image for each class. The highly non-linear and high-capacity properties [17] make the multi-layer CNNs difficult to train, especially when the number of training samples is not sufficiently large. We explore multiple important aspects in this CNN-based classification system, including network architecture, image preprocessing, hyper-parameters selection, and data augmentation, which are important for CNNs to achieve effective and reliable cell classification. Furthermore, we conduct rigorous experimental comparison with two state-of-the-art hand-designed shallower image representation models, i.e., bag-of-features (BoF) and Fisher Vector (FV), to investigate the advantages and disadvantages of our CNN-based framework on cell image classification. Our system has participated in the *Contest on Performance Evaluation on Indirect Immunofluorescence Image Analysis Systems* hosted by ICPR 2014[1] and won the fourth place among 11 international teams.

The rest of the paper is organized as follows. Section II reviews the classification models of BoF, FV and deep CNNs. In Section III, our CNN-based framework for cell images classification is presented and a set of key factors are discussed. Section IV reports the experimental investigation and comparison, and the conclusions are drawn in Section V.

We were invited by the ICPR 2014 contest organizers to

Zhimin Gao, Lei Wang, Luping Zhou and Jianjia Zhang are with the School of Computer Science and Software Engineering, Uinversity of Wollongong, NSW 2522, Australia (e-mail: zg126@uowmail.edu.au, {leiw, lupingz@uow.edu.au}, jz163@uowmail.edu.au).

[1] Contest website is at http://i3a2014.unisa.it/?page_id=91.

report our system in a workshop short paper [18]. This paper significantly extends that workshop paper in the following aspects: i) a more detailed description of our deep CNN-based classification framework for HEp-2 cell images is presented and multiple key factors for effectively training a reliable deep CNN are discussed and experimentally demonstrated; ii) the role of image rotation as a data augmentation method in helping the deep CNN to achieve robust representations in this classification task is investigated and analyzed; iii) systematic experimental comparisons of our CNN-based framework and the state-of-the-art hand-designed classification models are conducted; iv) the excellent generalization capability of our cell classification system with respect to different laboratory settings is demonstrated by transferring the learned network across two datasets with easy implementation, which makes our system attractive for practical clinical applications.

II. Related Work

Bag-of-features and Fisher Vector Models. The BoF model [19] generally consists of four stages: local feature extraction, dictionary learning, feature encoding, and feature pooling. The dictionary is composed of a set of visual words describing the common visual patterns shared by local descriptors. The relationship between local descriptors and visual words is characterized by feature encoding. A variety of coding methods have been proposed in the literature [20], [21], [22], [23]. On top of these, spatial pyramid matching (SPM) [24] is usually utilized to incorporate the spatial information of an image. The BoF model has been applied to staining patterns classification [13], [25], [26], [27], in which one or more of the above four stages are tailored to obtain better cell image representations for classification. Readers are referred to the review [4] for more details.

In the past several years, FV model has shown superior performance to the BoF model [28], [29], [30]. Their main differences lie at dictionary learning and feature encoding. The dictionary in FV is generated by a probabilistic model, e.g., the Gaussian mixture model (GMM), that characterizes the distribution of local descriptors. Each local descriptor is then encoded by the first- and second-order gradients with respect to the model parameters. FV model has also been applied to cell image classification [31], [32].

Deep Convolutional Neural Networks. CNNs belong to a class of learning models inspired by the multi-stage processes of visual cortex [33]. A pioneering work of CNNs was Fukushima's "neocognitron" [34]. It has a structure similar to the hierarchical model of the visual nervous system discovered by Hubel and Wiesel [35]. Each stage of the network imitates the functions of simple and complex cells in the primary visual cortex. Later on, LeCun et al. extended the neocognitron by utilizing backpropagation algorithm to train the model parameters of CNNs and achieved excellent performance in hand-written digit recognition [16].

With the advent of fast parallel computing, better regularization strategies, and large-scale datasets, deep CNNs models have recently significantly outperformed the models with hand-crafted features on generic object classification, detection and retrieval [15], as well as other visual recognition tasks like face verification [36]. As for HEp-2 cell image classification, Malon et al. [3] adopted a CNN to classify cell images. Buyssens et al. [37] designed a multiscale CNN for cytological pleural cancer cells classification. Our CNN framework presented in this paper is different from their works in terms of both image preprocessing method and network architecture. Moreover, our CNN performs better than the CNN designed in [3] on HEp-2 cell classification.

Although CNNs have been initially applied to cell image classification, the following issues have not been systematically investigated and thus remain unclear: i) what are the key issues when adopting deep CNNs for cells classification? ii) how is the performance of the CNN-based classification model when compared with the well-established classification models in the literature, especially the BoF and FV models? These issues will be carefully investigated and addressed in this work.

III. Proposed Framework

The proposed deep CNN-based HEp-2 cell image classification framework consists of three components: image preprocessing, network training, and feature extraction and classification, which are elaborated in this section. Also, data augmentation which plays an important role in this classification framework will be described and analyzed.

A. Network Architecture

A proper selection of network architecture is crucial to CNNs. Usually, deep CNNs are composed of multiple convolutional layers interlaced with subsampling (pooling) layers, as shown in Fig. 1. Each layer outputs a set of two-dimensional feature maps, each of which represents a specific feature detected from all positions of the input. These feature maps are in turn used as the input of the next layer. Fully-connected layers are usually stacked on the top of the network to conduct classification.

Our deep CNN shares the basic architecture as the classical LeNet-5 [16]. Specifically, it contains eight layers. Among them, the first six layers are convolutional layers alternated with pooling layers, and the remaining two are fully-connected layers for classification.

1) Convolutional Layer: Let's assume that it is the lth layer. Let N^l denote the number of feature maps at this layer, where l is used as a superscript. Accordingly, each feature map is denoted as $\mathbf{h}_j^l$ $(j = 1, 2, ..., N^l)$. This convolutional layer is parametrized by an array of two-dimensional filters $\mathbf{W}_{ij}^l$ associating the ith feature map $\mathbf{h}_i^{l-1}$ in the $(l-1)$th layer with the jth feature map $\mathbf{h}_j^l$ in the lth layer and the bias b_j. Each filter acts as a feature detector to detect one particular kind of features by convolving with every location of the input feature map. To obtain $\mathbf{h}_j^l$, each input feature map $\mathbf{h}_i^{l-1}$ $(i = 1, 2, ..., N^{l-1})$ is firstly convolved with the corresponding filter $\mathbf{W}_{ij}^l$. The results are summed and appended with the bias b_j^l. After that, a non-linear activation function $\phi(\cdot)$, which can be sigmoid, tanh or rectified linear function

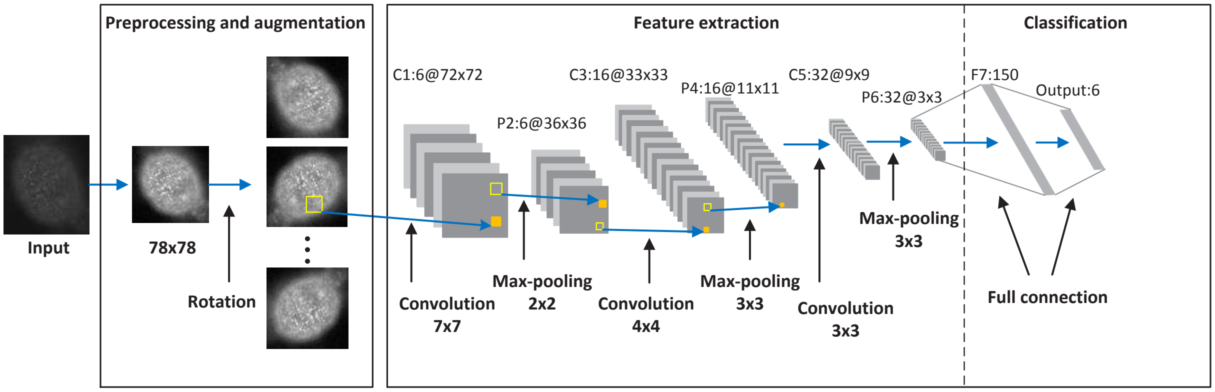


Fig. 1. The architecture of our deep convolutional neural network classification system for HEp-2 cell images. Each plane within the feature extraction stage denotes a feature map. The convolutional layer and max-pooling layer is abbreviated as C and P respectively. C1:6@72×72 means that this is a convolutional layer, and is the first layer of the network. This layer is comprised of six feature maps, each of which has size of 72×72. The symbols and number above the feature maps of other layers have the similar meaning, whereas F7:150 means that this is a fully-connected layer. It is the seventh layer of the network and has 150 neurons. The words and number between two layers stand for: the operation, i.e., convolution or max-pooling, applied to the feature maps of the previous layer in order to obtain the feature maps of this layer; and the size of each filter or the size of pooling region.

[14], is applied in an element-wise manner. Mathematically, the feature maps of the lth layer can be expressed as follows:

$$\mathbf{h}_j^l = \phi\Big(\sum_{i=1}^{N^{l-1}} \mathbf{h}_i^{l-1} * \mathbf{W}_{ij}^l + b_j^l\Big),\ j = 1, 2, ..., N^l. \tag{1}$$

where $*$ denotes the convolution operation.

2) Pooling Layer: A pooling layer down-samples a feature map. This will greatly reduce the computation of training a CNN and also introduces invariance to small translation of input images. Max-pooling or average-pooling is usually applied. The former selects the maximum activation over a small pooling region, while the latter uses the average activation over this region. Max-pooling generally performs better than average-pooling [38].

3) Classification Layer: Classification layers usually involve one or more fully-connected layers at the top of a CNN. Our network contains two fully-connected layers. The first fully-connected layer (F7 in Fig. 1) takes the cascade of all the feature maps of the sixth layer (denoted as $\mathbf{h}^6$) as input. This layer is parametrized by weights $\mathbf{W}^7$ and biases $\mathbf{b}^7$. The output of this layer $\mathbf{h}^7$ is obtained as $\mathbf{h}^7 = \phi(\mathbf{W}^7\mathbf{h}^6 + \mathbf{b}^7)$. The last fully-connected layer is the output layer and parametrized by weights $\mathbf{W}^8$ and biases $\mathbf{b}^8$. It contains n neurons corresponding to n classes of staining patterns, and outputs the probabilities $\hat{\boldsymbol{y}} = [\hat{y}_1, \hat{y}_2, ..., \hat{y}_n]^\top \in \mathbb{R}^n$ via softmax regression as follows:

$$\mathbf{h}^8 = \mathbf{W}^8\mathbf{h}^7 + \mathbf{b}^8,\ \mathbf{h}^8 \in \mathbb{R}^n \tag{2}$$

$$\hat{y}_j = \frac{\exp(h_j^8)}{\sum_{i=1}^{n} \exp(h_i^8)},\ j = 1, 2, ..., n. \tag{3}$$

where $\hat{y}_j$ is the output probability of the jth neuron.

The network architecture of our deep CNN is illustrated in Fig. 1. Specifically, the first layer convolves an input image with each of the six filters of size 7×7 with a stride of one pixel, and then adds a bias to each of them after convolution. We adopt the hyperbolic tangent function $\phi(x) = 1.7159 \tanh(\frac{2}{3}x)$ as the activation function. The second layer takes the output of the first layer as input, and applies max-pooling over non-overlapping regions of size 2×2 for each feature map. The third layer adopts filters of size 4×4, and has 16 feature maps. The fourth layer then applies max-pooling over non-overlapping pooling regions of size 3×3. The fifth layer employs filters of size 3×3 and includes 32 feature maps. The sixth layer employs 3×3 non-overlapping max-pooling to the output maps of the fifth layer. After that, the resulting 32 feature maps of size 3×3 are cascaded and passed to the first fully-connected layer containing 150 neurons.

When a cell image is fed into the network, the spatial resolution of each feature map decreases as the features are extracted hierarchically from one layer to next. The spatial information of each cell is extracted by the feature maps because of the spatial convolution and pooling operations, which are important to distinct different staining pattern types. The features obtained are invariant to small translation or shift of cell images, because the filter weights of the convolutional layers are uniform for different regions of the input maps and max-pooling is robust to small variations.

B. Image Preprocessing

An appropriate image preprocessing method that takes the characteristic of images into consideration is necessary for deep CNNs to obtain good internal feature representation and classification performance.

The brightness and contrast of the HEp-2 cell images provided by the ICPR 2014 contest (ICPR2014 dataset in short) vary greatly. To reduce this variance and enhance the contrast, we normalize each image by first subtracting the minimum intensity value of the image. The resulting intensity is then divided by the difference between the maximum and minimum intensity values. Furthermore, each image is resized to 78×78 to guarantee a uniform scale of all the images used for training. This size is approximately the average size of all the cell images. Examples of six staining patterns in ICPR2014 dataset and the preprocessed images are shown in Fig. 2. In addition, we just use the preprocessed whole cell images to train our network instead of adopting a mask to only use the foreground within each cell as Malon et al. in [3], because the mask information of each cell is usually unavailable in practice.

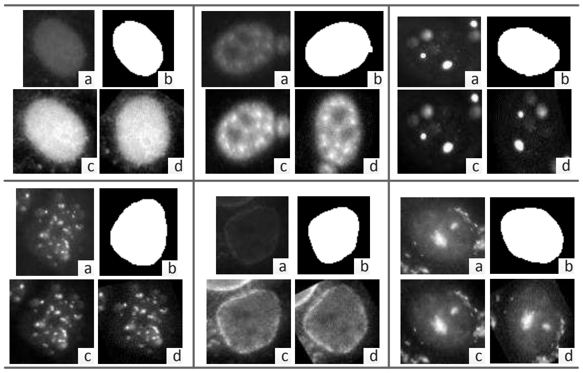


Fig. 2. Example cells of six classes in ICPR2014 dataset and their corresponding preprocessed and aligned images. There are four images for each cell: (a) the original image; (b) the mask of this cell image (we do not take advantage of it for training the CNN); (c) the preprocessed image when the original image is applied contrast normalization and resized; (d) the aligned image when (c) is aligned by PCA.

C. Data Augmentation

Deep CNNs are high-capacity architecture having a large number of parameters to be learnt. It will be difficult to effectively train a CNN when training images are insufficient. Data augmentation [14] has been regarded as a simple and effective way to generate more samples to train a CNN and gain robustness against a variety of variances.

For data augmentation in the cell image classification, we identify the following two points: i) generating new training images by rotating existing ones can effectively boost the classification performance of the CNNs; ii) instead of merely increasing the robustness of the CNNs against the global orientation of a cell, the extra samples generated via such rotation-based augmentation help to show the staining patterns within a cell image in different ways, which is a more important factor contributing to the improvement of the classification performance.

To demonstrate the first point, we keep rotating each training image with respect to its center by a step of θ degree. The newly generated images inherit the class label from the original training image, because rotating a cell image does not change its class label. By doing so, the original training set is enlarged by a factor of $m = \frac{360}{\theta}$, and this augmented training set is used to train the CNN.

To demonstrate the second point, we pre-align each cell image to approximately have the same global orientation. In this way, if the global orientation variance is really the main factor affecting the training performance of the CNN, we shall observe some improvement by using the pre-aligned training set. Also, augmenting this pre-aligned training set by rotation in further shall not lead to significantly better classification performance.

To investigate our hypothesis, we apply principal component analysis (PCA) to each cell's mask to obtain the principal direction of its shape and make all the cell images aligned. Applying this process to all training cell images makes them pre-aligned. Fig. 2 shows some example cell images after alignment. After that, we use the pre-aligned training images to train the CNN and then classify test images which are also pre-aligned.

We find that the CNN learnt in this manner does not show better performance than the CNN trained with the original training images. However, when data augmentation is applied to the pre-aligned training set images, the performance of the trained CNN increases greatly. This indicates that, in terms of cell classification, adequately demonstrating the staining patterns within a cell image via different ways is more important than removing the global orientation variance[2]. Detailed experimental results will be presented in Section IV.

D. Network Training

Due to the non-convex property of the cost surface of CNNs, it is essential to select appropriate network training parameters, e.g., learning rate, and regularization methods, e.g., weight decay and dropout [39] to make the network converge to good solutions fast.

Our deep CNN is parameterized by the weights and biases of different convolutional layers and fully-connected layers $\{\mathbf{W}^l, \mathbf{b}^l\}$, where $l = 1, 3, 5, 7, 8$. The total number of parameters is over $50,000$. The network is trained by minimizing the cross-entropy between the output probability vector $\hat{\boldsymbol{y}} = [\hat{y}_1, \hat{y}_2, ..., \hat{y}_n]^\top$ and the binary class label vector $\boldsymbol{y} = [y_1, y_2, ..., y_n]^\top$ with one non-zero entry "1" corresponding to the true class, which is expressed as follows.

$$E(\boldsymbol{y}, \hat{\boldsymbol{y}}) = -\sum_{j=1}^{n} y_j \log(\hat{y}_j) \tag{4}$$

The weights are initialized from a uniform distribution and the biases are initialized to zero. All these trainable parameters are updated periodically via stochastic gradient descent (SGD) [16] after evaluating the cost function. Let w^l denote a weight of the lth layer, i.e., an element of $\mathbf{W}^l$. Let b^l be a bias of the lth layer (an element of $\mathbf{b}^l$). Each weight w^l and bias b^l are updated by the following rules:

$$w^l := w^l - \eta \cdot \frac{\partial E}{\partial w^l}; \quad b^l := b^l - \eta \cdot \frac{\partial E}{\partial b^l} \tag{5}$$

where $\frac{\partial E}{\partial w^l}$ and $\frac{\partial E}{\partial b^l}$ are the partial derivatives of the cost function with respect to w^l and b^l respectively. They are calculated via back-propagating the output error to the lth layer [40], and η is the learning rate.

To smooth the directions of gradient descent and make the network converge fast, we employ momentum [41] to speed up the learning by guiding the descent direction with past

[2] A good example in contrast is human facial image, for which pre-alignment is generally helpful for recognition. This is because the patterns within a facial image, e.g., eyes, nose and mouth, have a rigid geometric association with the global orientation of the face. Pre-aligning the faces with respect to their global orientations effectively makes the patterns inside align with each other. Nevertheless, it is not such a case for cell images.

gradients. The update rules of w^l and b^l become as the follows:

$$\begin{aligned} v_w^l &:= \alpha \cdot v_w^l - \beta \cdot \eta \cdot w^l - \eta \cdot \frac{\partial E}{\partial w^l}; \quad w^l := w^l + v_w^l \\ v_b^l &:= \alpha \cdot v_b^l - \eta \cdot \frac{\partial E}{\partial b^l}; \quad b^l := b^l + v_b^l \end{aligned} \tag{6}$$

where v_w^l and v_b^l are the momentum variables for w^l and b^l respectively; α and β are the coefficients of momentum term and weight decay term, and their optimal values are experimentally tuned, as shown in Section IV. When training error rate becomes stabilized, the learning rate η will be reduced to achieve finer learning. The whole training process terminates after the classification error rates of both training set and validation set (which is held out from the given training images) plateau at some epochs.

In addition, another newly developed regularization strategy, dropout [39], is also investigated in the network training. It randomly sets a fraction of the activations in the hidden layers to zero to force the hidden units to learn more independent and robust features that could generalize well.

E. Feature Extraction and Classification

When classifying a test image, the same preprocessing and rotation in Section III-B and III-C are applied. This results in m rotated variants in total. Each of them is forward-propagated through the network, and the probability of this image for each of the n classes is obtained. To further improve the robustness of classification, we select four CNNs at different epochs of the training process and use them collectively for classification. The predicted class is the one having the maximum output probability averaged over the $4m$ probabilities, that is,

$$\hat{l} = \arg\max_j \hat{y}_j = \arg\max_j \frac{1}{4m} \sum_{k=1}^{m} \sum_{i=1}^{4} \hat{y}_{ik}, \; j = 1, 2, ..., n. \tag{7}$$

IV. Experimental Result

We evaluate our CNN classification system on two datasets of HEp-2 cell classification competition held by ICPR 2014 and 2012. The evaluation criterion is the mean class accuracy (MCA) newly adopted by ICPR 2014 competition. The average classification accuracy (ACA) used by the previous competition is also calculated for the ease of comparison.

A. Introduction of the HEp-2 Cell Datasets

ICPR2014 cell dataset: This dataset contains $13,596$ training cell images, and the test set is reserved by the competition organizers and not published yet. The cell images are extracted from 83 specimen images captured by monochrome high dynamic range cooled microscopy camera fitted on a microscope with a plane-Apochromat $20\times/0.8$ objective lens and an LED illumination source [5]. These specimen images have been manually segmented and annotated by specialists. Each image belongs to one of the six staining patterns: *homogeneous, speckled, nucleolar, centromere, nuclear membrane* and *golgi*, as shown in the top row of Fig. 3.

ICPR2012 cell dataset: It consists of $1,455$ cell images extracted from 28 specimens, which are acquired with a fluorescence microscope (40-fold magnification) coupled with 50W mercury vapor lamp and with a digital camera [3]. The dataset is pre-partitioned into training set (721 images) and test set (734 images). Each image belongs to one of the six classes: *homogeneous, coarse speckled, nucleolar, centromere, fine speckled* and *cytoplasmatic*, as shown in the bottom row of Fig. 3.

Comparing the two datasets shows that two of the six classes are different. Specifically, two sub-categories of ICPR2012 dataset (*fine speckled* and *coarse speckled*) are merged into one category (*speckled*) in ICPR2014 dataset, and two less frequent staining patterns appearing in daily clinical cases, *golgi* and *nuclear membrane* are introduced in ICPR2014 dataset for developing more realistic HEp-2 cell classification systems. **Moreover, because the images in the two datasets are captured with different laboratory settings, a classification system that can be easily transferred from one dataset to the other one will be highly desired.**

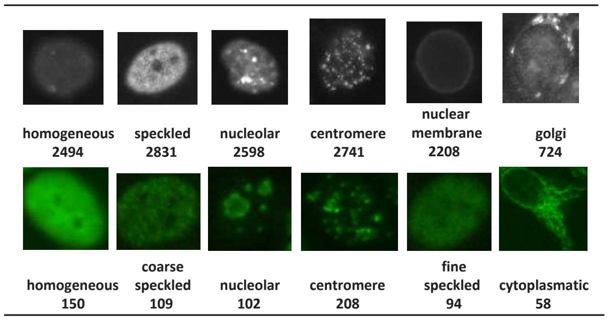


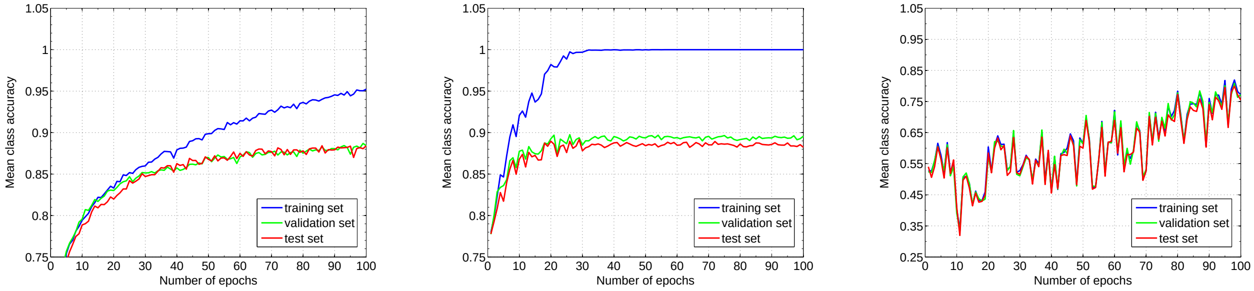
Fig. 3. Comparison of HEp-2 cell images of ICPR2014 dataset (top row) and ICPR2012 dataset (bottom row). The number below the name of each cell is the total number of this kind of cells in the training set of each dataset.

B. Experiments of Hyper-parameters Optimization

This experiment demonstrates the importance of properly tuning the hyper-parameters in the CNN-based system. We categorize the hyper-parameters into two groups: model-relevant and training-relevant, as listed in Tables I and II.

To tune these hyper-parameters, we randomly partition the $13,596$ cell images of ICPR2014 dataset into three subsets, that is, 64% for training (8701 images), 16% for validation (2175 images), and 20% for test (2720 images). This partition is utilized by all experiments on ICPR2014 dataset. Data augmentation is not used when tuning hyper-parameters. Following [41], the parameters are tuned until the error rate of not only the training set but also the validation set become sufficiently small and stabilized. The hyper-parameters obtained by this tuning process are summarized in Tables I and II.

We highlight that training-relevant hyper-parameters can significantly affect the convergence of cost function, the learning speed and the generalization capability of the network. Their impacts are demonstrated via the learning curves of MCA on training, validation and test sets shown from Fig.

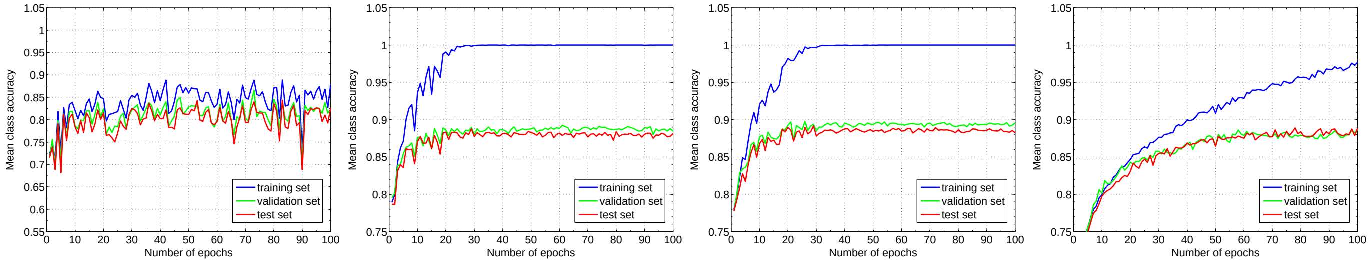


(a) Learning rate = 0.001

(b) Learning rate = 0.01

(c) Learning rate = 0.1

Fig. 4. Demonstration of the impact of learning rate. It shows that an over-small learning rate, e.g., 0.001, slows down the learning process, whereas an over-large learning rate, e.g., 0.1, destabilizes the learning process and degrades the classification performance. A better classification result can be obtained by properly tuning the learning rate, as shown in (b).



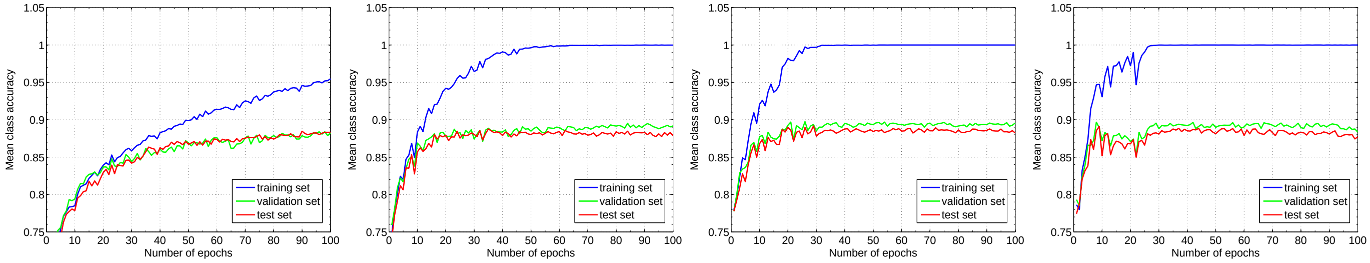
(a) Mini-batch size = 11

(b) Mini-batch size = 77

(c) Mini-batch size = 113

(d) Mini-batch size = 791

Fig. 5. Demonstration of the impact of mini-batch size. It shows that when mini-batch size is unnecessarily small, the learning process becomes bumpy and does not lead to the best result. On the other hand, when the mini-batch size is too large, the learning process becomes less responsive and the learning efficiency is decreased.



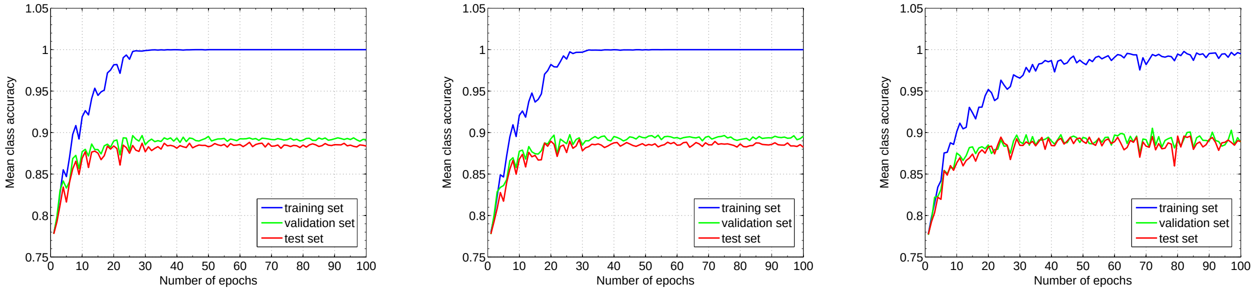
(a) Momentum coefficient = 0

(b) Momentum coefficient = 0.8

(c) Momentum coefficient = 0.9

(d) Momentum coefficient = 0.97

Fig. 6. Demonstration of the impact of momentum. It shows that using momentum can well accelerate the learning process. Meanwhile, a large momentum coefficient, e.g., 0.97, makes the descent direction dominated by the previous ones and causes oscillation at the initial stage. Also, it decreases the classification performance at the later stage.



(a) Weight decay coefficient = 0.00005

(b) Weight decay coefficient = 0.0005

(c) Weight decay coefficient = 0.005

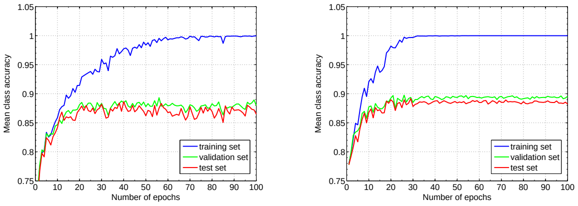
Fig. 7. Demonstration of the impact of weight decay. It shows that a smaller weight decay coefficient seems to be a safer choice, while a larger coefficient, e.g., 0.005, could destabilize the learning process.

TABLE I
MODEL-RELEVANT HYPER-PARAMETERS OBTAINED

Layer Number	**Layer Type**	**Hyper-parameter**
Layer 1	Convolution	Filter size: 7×7 Feature map number: 6 Activation function: hyperbolic tangent $\phi(x) = 1.7159 \tanh(\frac{2}{3}x)$
Layer 2	Pooling	Pooling region size: 2×2 Pooling method: max-pooling
Layer 3	Convolution	Filter size: 4×4 Feature map number: 16 Activation function: hyperbolic tangent $\phi(x) = 1.7159 \tanh(\frac{2}{3}x)$
Layer 4	Pooling	Pooling region size: 3×3 Pooling method: max-pooling
Layer 5	Convolution	Filter size: 3×3 Feature map number: 32 Activation function: hyperbolic tangent $\phi(x) = 1.7159 \tanh(\frac{2}{3}x)$
Layer 6	Pooling	Pooling region size: 3×3 Pooling method: max-pooling
Layer 7	Full connection	Neurons number: 150 Activation function: hyperbolic tangent $\phi(x) = 1.7159 \tanh(\frac{2}{3}x)$

TABLE II
TRAINING-RELEVANT HYPER-PARAMETERS OBTAINED

Hyper-parameter	Initial learning rate	Mini-batch size	Momentum coefficient	Weight decay coefficient	Dropout ratio
Value	0.01	113	0.9	0.0005	0



(a) Dropout ratio = 0.5

(b) Dropout ratio = 0

Fig. 8. Demonstration of the impact of dropout. It shows that the dropout strategy shall be used cautiously. As seen in (a), the learning process becomes slow and fluctuated on ICPR2014 cell dataset, when dropout is applied. A better learning process is obtained in (b) after removing dropout.

4 to Fig. 8. In each figure, we focus on one hyper-parameter while the others are set to their optimal values in Table II.

Fig. 4 (a) indicates that when learning rate is small, e.g., 0.001, the learning process is so slow that the MCA of the three sets have not become stable in 100 epochs. Properly increasing the learning rate effectively improves learning efficiency and the MCA becomes stable in 35 epochs, as shown in Fig. 4 (b). At the same time, an over-large learning rate, e.g., 0.1, will destabilize the learning process and degrade the classification performance. Also, Fig. 5, 6 and 7 demonstrate the impacts of mini-batch size, momentum and weight decay, respectively.

The comparison in Fig. 8 shows that the dropout strategy [39] shall be used cautiously. When dropout with ratio of 0.5 is applied to the first fully-connected layer of our CNN system, the learning process becomes slow and fluctuated on ICPR2014 cell dataset. Removing dropout leads to a stabler and faster learning process, as well as better classification performance. In light of this, we decide not to employ dropout when training our network on ICPR2014 dataset.

In sum, among the hyper-parameters of a CNN, the learning rate, mini-batch size, momentum coefficient, and weight decay coefficient can significantly impact the network training process. They have to be carefully tuned before satisfactory classification performance is obtained. For our deep CNN system, with the hyper-parameters set in Table II, we can achieve the MCA of 89.17% on the test set of ICPR2014 dataset without using data augmentation.

C. Experiments on Data Augmentation

This experiment demonstrates the two points presented in Section III-C, which are recapped as follows: i) generating new training images by rotating existing ones can effectively boost the classification performance of the CNNs; ii) instead of merely increasing the robustness of the CNNs against the global orientation of a cell, the extra samples generated via such rotation-based augmentation help to show the staining patterns within a cell image in different ways, which is a more important factor contributing to the improvement of the classification performance.

Effectiveness of data augmentation. We augment the training set by rotating each cell image for $360°$, with the step of $36°$, $18°$ and $9°$, respectively. In this way, the training set is expanded by 10, 20 and 40 times, and they are used to train the CNNs, respectively. To improve the robustness of our system, we select four CNNs corresponding to the 75th, 85th, 95th and 100th epochs after the network learning becomes stable. A test image will go through the same rotation process as the training images and be jointly classified by the four CNNs as in Eq.(7). We call this network "4CNNs". As shown in the first row of Table III, the MCA is significantly improved (by more than 7 percentage points) from "No data augmentation" to "Augmentation by a rotation angle step of $36°$". Furthermore, applying a smaller angle step to generate more training data pushes the MCA even higher, reaching 96.76%. Similar results can be observed on the ACA values. These consistent and continuous improvements well demonstrate the effectiveness and efficiency of data augmentation on cell image classification.

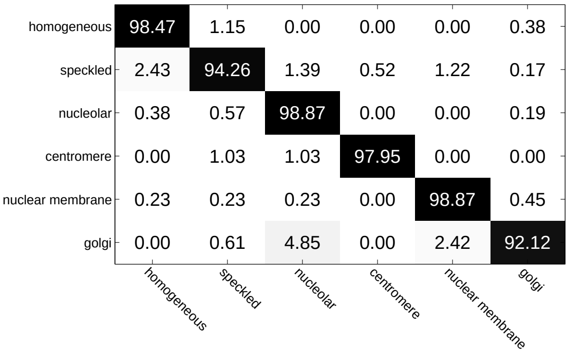
Data augmentation vs. pre-alignment. To gain more insight on the rotation-based data augmentation, we pre-align all the cell images with PCA as described in Section III-C to train the CNNs. We call this method "4CNNs-Align". Two experiments are conducted: i) only using these aligned images to train the CNNs without performing data augmentation; and ii) as a comparison, we further rotate each aligned training image by $360°$, with an angle step of $36°$. The augmented training set is used for training. As previous, augmentation (or no augmentation) is equally applied to test images.

As shown in Table III, when no augmentation is performed, 4CNNs-Align does not achieve any improvement over 4CNNs. This indicates that pre-alignment does not help here. In contrast, when training data are augmented by rotation (even with the largest angle step of $36°$), 4CNNs-Align improves significantly. This sharp change clearly demonstrates that through the rotation-based augmentation, the CNNs can access

TABLE III
CLASSIFICATION ACCURACY OF OUR DEEP CNNS ON ICPR2014 DATASET

Method	**Accuracy** (on test set)	No data augmentation	Augmentation by a rotation angle step of 36°	Augmentation by a rotation angle step of 18°	Augmentation by a rotation angle step of 9°
4CNNs	MCA	88.58%	95.99%	96.71%	**96.76**%
	ACA	89.04%	96.51%	97.10%	**97.24**%
Method	**Accuracy** (on test set)	No data augmentation	Augmentation by a rotation angle step of 36°	-	-
4CNNs-Align	MCA	88.86%	**95.13**%	-	-
	ACA	88.71%	**95.33**%	-	-

more examples showing the diverse staining patterns within cell images. Compared with pre-alignment, this is the more important factor contributing to the performance improvement.



	homogeneous	speckled	nucleolar	centromere	nuclear membrane	golgi
homogeneous	98.47	1.15	0.00	0.00	0.00	0.38
speckled	2.43	94.26	1.39	0.52	1.22	0.17
nucleolar	0.38	0.57	98.87	0.00	0.00	0.19
centromere	0.00	1.03	1.03	97.95	0.00	0.00
nuclear membrane	0.23	0.23	0.23	0.00	98.87	0.45
golgi	0.00	0.61	4.85	0.00	2.42	92.12

Fig. 9. Confusion matrix of our 4CNNs (9° rotation) (%).

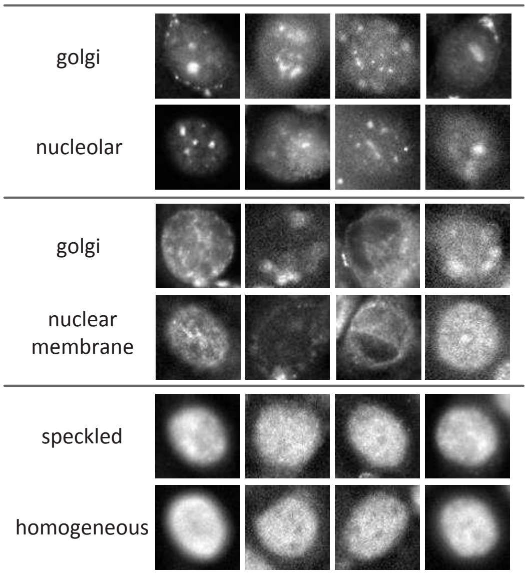


Fig. 10. Misclassification examples of the three highest misclassification rates in the confusion matrix of Fig. 9. Every two rows form a group, and the first row shows cells that are misclassified to the cell type of the second row.

In addition, the confusion matrix of the best 4CNNs (with the rotation angle step of $9°$) is shown in Fig. 9. The overall classification performance is very promising. The staining patterns *nucleolar* and *nuclear membrane* obtain the highest classification accuracy (both 98.87%), which means that they are well separated from the others. The maximum misclassification rate (4.85%) happens to *golgi* cells. They are easy to be misclassified as *nucleolar* cells, because both patterns consist of a few large dots within the cells (see misclassification examples in Fig. 10). Also, *golgi* can be confused with *nuclear membrane*. This may be because when the large dots within *golgi* cells are at the edge, they will look like the *nuclear membrane* cells having ring-like edges. In addition, the *speckled* cells are easy to be misclassified as *homogeneous* cells, probably because the densely distributed speckles are the main signatures for both patterns. Misclassification examples of these staining patterns are shown in Fig. 10.

D. Comparison with the BoF and Fisher Vector Models

Experimental setting. To ensure a fair comparison, the same image preprocessing in our CNN model is equally used in both models. For each cell image, SIFT descriptors are extracted from densely sampled patches with a stride of two pixels. The visual dictionary is generated by applying the k-means clustering to the descriptors extracted from training images. Local soft-assignment coding (LSC) [42], [20] is employed to encode the SIFT descriptors. SPM is used to partition each image into 1×1, 2×2 and 1×3 regions, and max-pooling is applied to extract representations from each region.

A similar setting is applied to the FV model. In addition, the 128-dimensional SIFT descriptors are decorrelated and reduced to dimensions of 64 by PCA as in [30]. A GMM is then estimated to represent the visual dictionary. Afterwards, each PCA-reduced SIFT descriptor is encoded with the improved Fisher encoding [29], where the signed square-root and l^2-normalization are applied to the coding vector. SPM with four regions (1×1 and 1×3) are adopted [30]. Following the literature, a multi-class linear SVM classifier is used in the BoF and FV models. In our implementation of BoF and FV, the publicly available VLFeat toolbox [43] is used.

Parameter setting. There are two primary parameters in the BoF and FV models: patch size and dictionary size (or equally, the number of components of the GMM in the FV model). We tune these parameters by five-fold cross-validation on the union of training and validation sets, with the criterion of MCA. The candidate patch sizes are 9×9, 11×11, 13×13, 15×15 and 20×20, while the candidate dictionary sizes are $1,000$, $2,000$, $3,000$, $4,000$, $5,000$ and $10,000$. Also, the number of Gaussian components will be chosen from 64, 128, 256, 512 and 1024 for FV. Through the cross-validation,

TABLE IV
COMPARISON OF CLASSIFICATION ACCURACY AMONG THE METHODS OF BOF, FV AND OUR DEEP CNNS ON ICPR2014 DATATSET

Accuracy	**Methods**	No data augmentation	Augmentation by a rotation angle step of 36°	Augmentation by a rotation angle step of 18°	Augmentation by a rotation angle step of 9°
MCA (%)	BoF	89.83	94.23	93.98	94.14
	FV	**91.60**	95.41	95.73	95.53
	4CNNs	88.58	**95.99**	**96.71**	**96.76**
ACA (%)	BoF	90.70	94.30	94.19	94.38
	FV	**92.65**	95.78	96.07	95.81
	4CNNs	89.04	**96.51**	**97.10**	**97.24**

the patch size and the dictionary size in the BoF model are selected as 15×15 and $10,000$. With the use of SPM, this results in a $80,000$-dimensional representation for each cell image. For the FV model, the patch size is chosen as 20×20 and the number of GMM components is 512. With the use of SPM, this leads to a $262,144$-dimensional representation for each image.

Comparison results. The BoF, FV and CNNs models are compared on the same training and test sets. Also, both of the cases, i.e., with and without data augmentation, are investigated. To be fair, when data augmentation is used, the visual dictionary in the BoF and FV models will be built with the augmented training set. Also, to keep consistent with the setting of our deep CNNs system, each test image in this case will be equally augmented and its label is predicted in the way similar to Eq.(7), except that the probabilities are replaced by the decision values of the linear SVM classifier.

As shown in Table IV, FV is consistently better than BoF, regardless of whether data augmentation is applied or not. This agrees well with the literature. Furthermore, both BoF and FV can well benefit from data augmentation, with an average performance increase of about 4 percentage points. Compared with BoF and FV, 4CNNs show slightly lower performance (88.85% vs. 89.83% for BoF and 91.60% for FV), when there is no augmentation. However, 4CNNs outperform both BoF and FV once data augmentation is applied. In specific, the highest MCA, 96.76%, is obtained by our 4CNNs, while BoF and FV achieve only 94.23% and 95.73% respectively. Similar situation can be observed from the ACA values. These results suggest that i) when training samples are not sufficient, the high-capacity CNNs are more difficult to train than the shallower, hand-designed models such as BoF and FV; and ii) by properly using data augmentation to generate more training data, the CNNs can be better trained and are able to achieve better performance than the BoF and FV models.

E. Experiments on the Generalization across Datasets

As previously mentioned, HEp-2 cell image classification varies with laboratory settings, the types of staining patterns involved, and the size of dataset. Such differences can be well seen from the ICPR2014 and ICPR2012 datasets. As a result, it is highly desired that a cell classification system trained with one dataset can be conveniently adapted to another one. Owning this feature not only improves the efficiency of system building, but also can take full advantages of the image data in different datasets. To demonstrate this feature for our CNN-based system, we compare the CNN purely trained on ICPR2012 dataset (called CNN-Standard in short) with the other CNN which is an adapted version of the CNN pre-trained on ICPR2014 dataset to ICPR2012 dataset (called CNN-Finetuning).

Following previous experimental settings, CNN-Standard is trained with the 721 training images predefined in ICPR2012 dataset. The dropout strategy (of ratio 0.5) is used, because it can benefit network training on this small dataset. CNN-Standard is trained by 100 epochs and then used to classify the predefined test images by following Eq.(7).

To train CNN-Finetuning, we first select a basic CNN system learnt with the ICPR2014 dataset. It is the one obtained at the 100th epoch when the system is trained with an augmented (rotation with an angle step of $9°$) training set of ICPR2014. Afterwards, this basic system is fine-tuned with the training set of ICPR2012 dataset, with or without data augmentation. To demonstrate the efficiency, we only fine-tune this basic system by 10 epochs, which takes significantly less time than the 100 epochs spent in training CNN-Standard.

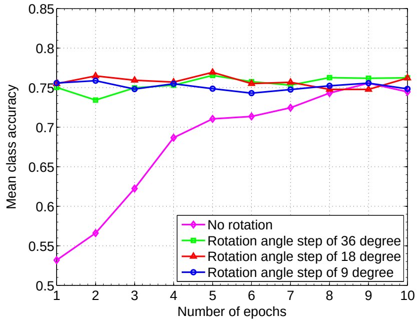


Fig. 11. The MCA of test set obtained by CNN-Finetuning at each of the 10 epochs. Data augmentation with various angle steps is investigated.

The evolution of the MCA on test set with the 10 epochs is plotted in Fig. 11. As shown by the line of “No rotation”, CNN-Finetuning does not work well at the beginning. Nevertheless, it catches up quickly in a couple of epochs and reaches a satisfying performance in 10 epochs. Furthermore, the adaption stage is significantly shortened, by applying data augmentation to the small training set of ICPR2012 to increase training samples. These results demonstrate the high efficiency of the adaptation of our CNN-based system, especially considering that there are two different classes of staining patterns across these datasets. Comparison of CNN-Standard and CNN-Finetuning is shown in Table V. It is interesting to note

TABLE V
CLASSIFICATION ACCURACY OF OUR CNN-BASED SYSTEM ON ICPR2012 DATASET

Accuracy (on test set)	**Methods**	No data augmentation	Augmentation by a rotation angle step of 36°	Augmentation by a rotation angle step of 18°	Augmentation by a rotation angle step of 9°
MCA (%)	CNN-Standard	63.1	72.4	72.4	73.2
	CNN-Finetuning	74.5	76.3	76.2	74.9
ACA (%)	CNN-Standard	64.3	70.2	70.0	70.1
	CNN-Finetuning	72.9	74.8	74.7	73.3

that CNN-Finetuning consistently outperforms CNN-Standard, even though it is only fine-tuned for a few epochs. We attribute its superiority to the good initialization of the network obtained from the training process on ICPR2014 dataset. Based on the above results, we believe that our CNN-based system will be a better option for practical applications.

At last, we compare our CNN-Finetuning (rotation with an angle step of $36°$) with other methods reported in the literature in Table VI. As seen, it outperforms the CNN at the ICPR2012 contest and the best-performing method of that contest. Our CNN-Finetuning is just slightly inferior to the method in [8]. That method combines two kinds of hand-crafted features: the distribution of SIFT and gradient-oriented co-occurrence LBP, and a dissimilarity representation of an image is created with them.

TABLE VI
COMPARISON WITH OTHER METHODS ON THE ICPR2012 DATASET

Method	**Average classification accuracy (ACA)**
2012 contest best-performing method [3]	68.7%
2012 contest CNN [3]	59.8%
Nosaka et al. [7]	68.5%
Shen et al. [26]	74.4%
Faraki et al. [31]	70.2%
Larsen et al. [44]	71.5%
Theodorakopoulos et al. [8]	**75.1**%
Our CNN-Finetuning	**74.8**%

In addition, it is worth mentioning that in the ICPR2014 contest [5], the three methods that perform better than or comparable to our deep CNNs system (87.10%, 83.64% and 83.33% vs. 83.23% with the MCA criterion) are built on the extension of the traditional BoF model. The first method utilizes multi-scale and multiple types of local descriptors; the second method adopts the hand-crafted rotation invariant dense scale local descriptor; and the third method combines morphological features and different local texture features. In contrast, our CNNs system generates discriminative features from raw pixels directly by utilizing class label information and jointly learns the classifier in a single architecture.

F. Discussion on Computational Issues

For the CNN-based classification system, training the network is the most time-consuming step in the whole pipeline. However, this process can be well accelerated by utilizing GPU programming. Also, as previously shown, an existing CNN-based system can be efficiently transferred to a new but related task via a short training process. Once the network is trained, a test cell image only needs to go through the network and can be classified in 1.2 seconds on a computer with 3.30GHz Intel CPU and 16GB RAM.

For the BoF and FV models, building visual dictionary or the GMM is computationally intensive, especially when there are a large number of training images, e.g., due to the use of data augmentation. For example, building a dictionary of $10,000$ visual words and the GMM of 512 components takes more than 4 days and 2 days in our implementation, when the training set of ICPR2014 dataset is augmented by rotation with an angle step of $9°$. Also, a large dictionary in the BoF model could slow down the encoding process, e.g., around 78 seconds per image in our experiment. Although this process can be well shortened in the FV model, it still takes about three seconds per image. In addition, SPM is usually needed to attain better classification performance. In this case, the dimensions of the resulting image representation are much higher than that in the CNN-based system ($80,000$ or $262,144$ vs. 150 only) .

V. CONCLUSION

This paper proposes an automatic HEp-2 cell staining patterns classification framework with deep convolutional neural networks. We give a detailed description on various aspects of this framework and carefully discuss a number of key issues that could affect its classification performance. Extensive experimental study on two benchmark datasets demonstrates i) the advantages of our framework over the well-established image classification models on cell image classification; ii) the importance and effectiveness of data augmentation, especially when training images are not sufficient; iii) the desirable generalization capability of our CNN-based system across different datasets, which makes our system attractive for practical tasks.

Much future work can be done to further improve the performance of the proposed system. In particular, a super-CNN trained with a large-scale generic image benchmark, ImageNet [45], has recently prevailed on many generic visual recognition tasks. We would like to explore the effectiveness of the features generated by this CNN for HEp-2 cell image and the adaption of this CNN to cell image classification. These issues will be of significance considering the substantial differences between generic images and HEp-2 cell images.

REFERENCES

[1] A. Rigon, P. Soda, D. Zennaro, G. Iannello, and A. Afeltra, “Indirect immunofluorescence in autoimmune diseases: assessment of digital images for diagnostic purpose,” *Cytometry Part B: Clinical Cytometry*, vol. 72, no. 6, pp. 472–477, 2007.

[2] P. L. Meroni and P. H. Schur, “Ana screening: an old test with new recommendations,” *Annals of the rheumatic diseases*, vol. 69, no. 8, pp. 1420–1422, 2010.

[3] P. G. S. P. Foggia, P. and M. Vento, "Benchmarking hep-2 cells classification methods," *Medical Imaging, IEEE Transactions on*, vol. 32, no. 10, pp. 1878–1889, Oct 2013.

[4] P. Foggia, G. Percannella, A. Saggese, and M. Vento, "Pattern recognition in stained hep-2 cells: Where are we now?" *Pattern Recognition*, vol. 47, no. 7, pp. 2305–2314, 2014.

[5] M. V. Brian C. Lovell, Gennaro Percannella and A. Wiliem, "Performance evaluation of indirect immunofluorescence image analysis systems," *ICPR 2014*. [Online]. Available: http://i3a2014.unisa.it/

[6] D.-C. He and L. Wang, "Texture unit, texture spectrum, and texture analysis," *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 28, no. 4, pp. 509–512, Jul 1990.

[7] R. Nosaka and K. Fukui, "Hep-2 cell classification using rotation invariant co-occurrence among local binary patterns," *Pattern Recognition*, vol. 47, no. 7, pp. 2428–2436, 2014.

[8] I. Theodorakopoulos, D. Kastaniotis, G. Economou, and S. Fotopoulos, "Hep-2 cells classification via sparse representation of textural features fused into dissimilarity space," *Pattern Recognition*, vol. 47, no. 7, pp. 2367–2378, 2014.

[9] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[10] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1, June 2005, pp. 886–893 vol. 1.

[11] R. Haralick, K. Shanmugam, and I. Dinstein, "Textural features for image classification," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. SMC-3, no. 6, pp. 610–621, Nov 1973.

[12] G. Thibault, J. Angulo, and F. Meyer, "Advanced statistical matrices for texture characterization: Application to cell classification," *Biomedical Engineering, IEEE Transactions on*, vol. 61, no. 3, pp. 630–637, March 2014.

[13] A. Wiliem, C. Sanderson, Y. Wong, P. Hobson, R. F. Minchin, and B. C. Lovell, "Automatic classification of human epithelial type 2 cell indirect immunofluorescence images using cell pyramid matching," *Pattern Recognition*, vol. 47, no. 7, pp. 2315–2324, 2014.

[14] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks." in *NIPS*, vol. 1, no. 2, 2012, p. 4.

[15] A. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "Cnn features off-the-shelf: An astounding baseline for recognition," in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2014 IEEE Conference on*, June 2014, pp. 512–519.

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

[17] Y. A. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller, "Efficient backprop," in *Neural networks: Tricks of the trade*. Springer, 2012, pp. 9–48.

[18] Z. Gao, J. Zhang, L. Zhou, and L. Wang, "Hep-2 cell image classification with convolutional neural networks," in *Pattern Recognition Techniques for Indirect Immunofluorescence Images (I3A), 2014 1st Workshop on*, Aug 2014, pp. 24–28.

[19] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray, "Visual categorization with bags of keypoints," in *Workshop on statistical learning in computer vision, ECCV*, vol. 1, no. 1-22, 2004, pp. 1–2.

[20] L. Liu, L. Wang, and X. Liu, "In defense of soft-assignment coding," in *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 2011, pp. 2486–2493.

[21] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong, "Locality-constrained linear coding for image classification," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, 2010, pp. 3360–3367.

[22] H. Jegou, M. Douze, C. Schmid, and P. Perez, "Aggregating local descriptors into a compact image representation," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, June 2010, pp. 3304–3311.

[23] O. Boiman, E. Shechtman, and M. Irani, "In defense of nearest-neighbor based image classification," in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. IEEE, 2008, pp. 1–8.

[24] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," in *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, vol. 2. IEEE, 2006, pp. 2169–2178.

[25] X. Kong, K. Li, J. Cao, Q. Yang, and L. Wenyin, "Hep-2 cell pattern classification with discriminative dictionary learning," *Pattern Recognition*, vol. 47, no. 7, pp. 2379–2388, 2014.

[26] L. Shen, J. Lin, S. Wu, and S. Yu, "Hep-2 image classification using intensity order pooling based features and bag of words," *Pattern Recognition*, vol. 47, no. 7, pp. 2419–2427, 2014.

[27] R. Stoklasa, T. Majtner, and D. Svoboda, "Efficient k-nn based hep-2 cells classifier," *Pattern Recognition*, vol. 47, no. 7, pp. 2409–2418, 2014.

[28] F. Perronnin and C. Dance, "Fisher kernels on visual vocabularies for image categorization," in *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*. IEEE, 2007, pp. 1–8.

[29] F. Perronnin, J. Sánchez, and T. Mensink, "Improving the fisher kernel for large-scale image classification," in *Computer Vision–ECCV 2010*. Springer, 2010, pp. 143–156.

[30] J. Sánchez, F. Perronnin, T. Mensink, and J. Verbeek, "Image classification with the fisher vector: Theory and practice," *International journal of computer vision*, vol. 105, no. 3, pp. 222–245, 2013.

[31] M. Faraki, M. T. Harandi, A. Wiliem, and B. C. Lovell, "Fisher tensors for classifying human epithelial cells," *Pattern Recognition*, vol. 47, no. 7, pp. 2348–2359, 2014.

[32] X.-H. Han, J. Wang, G. Xu, and Y.-W. Chen, "High-order statistics of microtexton for hep-2 staining pattern classification," *Biomedical Engineering, IEEE Transactions on*, vol. 61, no. 8, pp. 2223–2234, Aug 2014.

[33] D. H. Hubel and T. N. Wiesel, "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex," *The Journal of physiology*, vol. 160, no. 1, p. 106, 1962.

[34] K. Fukushima, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biological Cybernetics*, vol. 36, no. 4, pp. 193–202, 1980.

[35] D. H. Hubel and T. N. Wiesel, "Receptive fields of single neurones in the cat's striate cortex," *The Journal of physiology*, vol. 148, no. 3, pp. 574–591, 1959.

[36] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification," in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, June 2014, pp. 1701–1708.

[37] P. Buyssens, A. Elmoataz, and O. Lézoray, "Multiscale convolutional neural networks for vision–based classification of cells," in *Computer Vision–ACCV 2012*. Springer, 2013, pp. 342–352.

[38] Y.-L. Boureau, J. Ponce, and Y. LeCun, "A theoretical analysis of feature pooling in visual recognition," in *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 2010, pp. 111–118.

[39] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.

[40] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural computation*, vol. 1, no. 4, pp. 541–551, 1989.

[41] Y. Bengio, "Practical recommendations for gradient-based training of deep architectures," in *Neural Networks: Tricks of the Trade*. Springer, 2012, pp. 437–478.

[42] J. C. van Gemert, J.-M. Geusebroek, C. J. Veenman, and A. W. Smeulders, "Kernel codebooks for scene categorization," in *Computer Vision–ECCV 2008*. Springer, 2008, pp. 696–709.

[43] A. Vedaldi and B. Fulkerson, "Vlfeat: An open and portable library of computer vision algorithms," in *Proceedings of the international conference on Multimedia*. ACM, 2010, pp. 1469–1472.

[44] A. Larsen, J. Vestergaard, and R. Larsen, "Hep-2 cell classification using shape index histograms with donut-shaped spatial pooling," *Medical Imaging, IEEE Transactions on*, vol. 33, no. 7, pp. 1573–1580, July 2014.

[45] J. Deng, A. C. Berg, K. Li, and L. Fei-Fei, "What does classifying more than 10,000 image categories tell us?" in *Computer Vision–ECCV 2010*. Springer, 2010, pp. 71–84.