Comparing Image Representations for Training a Convolutional Neural Network to Classify Gender

Choon-Boon Ng, Yong-Haur Tay, Bok-Min Goi
Faculty of Engineering and Science
Universiti Tunku Abdul Rahman
Kuala Lumpur, Malaysia
{ngcb,tayyh,goibm}@utar.edu.my

***Abstract*—In this work, we evaluated the effect of different image representations on the classification performance of a convolutional neural network. Several different methods for normalization of the input data were also considered. The network was discriminatively trained for the task of gender classification of pedestrians. A publicly available dataset was used for training, containing both frontal and rear views of pedestrians. The best result was obtained using grayscale representation as compared to RGB and YUV, giving cross-validated accuracy of 81.5 % on the dataset. The performance of the convolutional neural network is competitive and comparable to previous works on the same dataset.**

Keywords- convolutional neural network; pedestrian; gender classification; input representation

I. INTRODUCTION

A convolutional neural network is a class of neural network with a multiple-layered architecture and using the idea of shared filter weights to learn task-dependent features from the training data provided to it. Its origin is inspired biologically by the hierarchical nature of the mammalian primary visual cortex. Other such biologically inspired network architectures which have been studied in computer vision system include the Neocognitron [1], which can be considered as its predecessor, and the HMAX model [2].

Neural networks with several hidden layers such as convolutional neural networks are considered to have a deep architecture. Compared to shallow architectures, a deep architecture is believed to be able to more efficiently learn complicated functions to represent the high-level abstractions required for complex tasks in artificial intelligence [3]. Until recently, with the breakthrough of greedy layerwise training introduced in [4], it has been difficult to train such multilayered network architectures. However, the convolutional neural network has been a notable exception, having been successfully trained using back-propagation [5].

Convolutional neural networks have achieved highly competitive results in a variety of pattern recognition problems such as handwritten digit recognition [6], face detection [7], traffic sign classification [8] pedestrian detection [9], object recognition [10] and action recognition [11]. A convolutional neural network integrates feature extraction and classification into a single framework. Discriminative feature detectors are learnt automatically by the network from the data during the supervised training stage, hence the classifier system design does not need to rely on difficult hand-engineered feature extraction.

We are interested in the problem of gender classification of humans using computer vision. The majority of previous research reported in literature relied on using only the face or head region alone to train a gender classifier [12]. In this work, we make use of the whole body of a person to train a gender classifier. This is justifiable in situations where the face is of insufficient resolution or not visible (e.g. rear view of the person)

The problem of inferring gender from whole body information was first investigated in [13], in which Histogram of Oriented Gradients (HOG) features [14] were used to represent patches of the human body image. These features were used to train a boosting type classifier. HOG features have also been used in other prior works [15][16]. Gabor filters were used to extract features in [17]. The features' dimensions were reduced using various manifold learning methods such as Principal Component Analysis and Locality Sensitive Discriminant Analysis. The features are then fed into a linear Support Vector Machine classifier. The best result was obtained in a pipeline containing a view classifier before the gender classifier.

In this work, we trained a convolutional neural network for the task of pedestrian gender classification. When used for such vision tasks, the input data fed into the network are the raw pixel values of the image. An image can be represented in various color models. Various researchers have used different image representations to train the convolutional neural network, such as grayscale, RGB and YUV. However, it is not clear which is the most suitable representation for our task. Referring to the prior works on recognizing gender from whole body, color was used as supplementary features in [15] [16].

The remainder of this paper is organized as follows. In section 2, the methodology of our experiment is presented. In section 3, details of the experiments and training procedures are given. The results are analyzed and compared in section 4. Finally, section 5 presents the concluding remarks for our paper and future work is suggested.

978-1-4799-3251-1/13 $31.00 © 2013 IEEE
DOI 10.1109/AIMS.2013.13

II. METHODOLOGY

A. Convolutional Neural Network

We use a discriminatively-trained convolutional neural network, inspired by the work of LeCun [6] for our experiment. A convolutional neural network typically includes two different kinds of layers corresponding to the simple and complex cells in the visual cortex model of [18], performing local feature detection and output pooling respectively. Each layer is comprised of a stack of so-called feature maps, which are obtained by performing convolution or subsampling on the previous layer's feature maps. The architecture of a convolutional neural network attempts to achieve some degree of scale, translation and deformation invariance in its structure through the use of local receptive fields, shared weights and spatial subsampling.

Fig. 1 shows the architecture of the convolutional neural network that we have used in our experiments. The network contains of a total of 7 layers, including the input and output layers. The hidden layers consist of 2 stages of convolution and subsampling layers, labeled as C and S respectively and a fully-connected layer of perceptron units labeled as F.

The feature maps in the first convolution layer C1 are obtained as a result of the convolution operation of a set of filters with the input units, which are then passed through a squashing activation function. Let $W_{i,j}$ be the filter of size $f \times g$ which connects the i-th feature map from the previous layer I_i to the j-th feature map C_j and b_j the corresponding trainable bias. The feature map is obtained as follows,

$$C_j = \sigma\left(\sum_{i \in S} W_{i,j} \otimes I_i + b_j\right) \tag{1}$$

Here $\otimes$ denotes the convolution operation and S denotes the set of all or selected feature maps from the previous layer. We use the hyperbolic tangent function as the squashing activation function which introduces non-linearities, as given by the following equation,

$$\sigma(x) = \tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{2}$$

Given that the size of a feature map is $h \times w$, convolution with filter of size $f \times g$ will produce an output feature map with a size of $(h - f + 1) \times (w - g + 1)$, i.e. convolution is only performed on valid pixel values as opposed to full convolution which requires zero-padding pixels of the image borders.

The same set of weights for the filter is shared by each unit of a feature map. The values of these weights are parameters which are obtained by supervised training. Hence, weight sharing greatly reduces the number of trainable parameters in order to help reduce overfitting. Overfitting is a problem in which a trained classifier performs well on the training data but does poorly when presented with new data which are not from the training set. Such a classifier is said to have poor generalization ability.

Next, the feature maps of the subsampling layer S2 is obtained by downsampling each feature map in layer C1. We use the maximum-pooling operation [2], in which the largest value in a local region is taken. Specifically, each feature map in layer C1 is partitioned into non-overlapping $p \times p$ sub-regions. The largest value is output from each sub-region to become the units that form the feature map of the S2 layer. Every feature map is thus downsampled in its size by a factor of p to form the next layer S2. Fig. 2 illustrates an example of the maximum-pooling operation. The 6×6 image, after applying 2×2 maximum-pooling, is reduced to 3×3. In the 2×2 sub-region shown, the largest value is taken for the subsampled output.

Similarly, the convolution layer C3 is obtained followed by subsampling layer S4. In our work, each feature map in layer C3 is connected to all the feature maps from layer S2. Layer F5 is a layer of perceptrons similar to the hidden layer of a neural network, each unit fully connected to all the units in the feature maps of layer S4. The units in layer F5 also use hyperbolic tangent activation function. Finally, the output layer units are connected to all the units of layer F5.

Softmax activation units are used in the output layer to form a linear classifier. The unit applies a softmax function on the input $\mathbf{x}$, given the weights $\mathbf{W}_m$ and bias $\mathbf{b}_m$ of the m-th unit. Given n is the number of units, the output of the m-th unit is given by the following equation,

$$y_m(x) = \frac{\exp(\mathbf{W}_m \mathbf{x} + \mathbf{b}_m)}{\sum_{i=1}^{n} \exp(\mathbf{W}_i \mathbf{x} + \mathbf{b}_i)} \tag{3}$$

Each softmax activation unit is associated with a class label and the output value gives the probability of the class given the input, weights and bias. The largest output probability gives the predicted class.

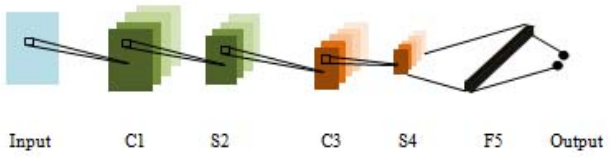

Fig. 1. Architecture of convolutional neural network.

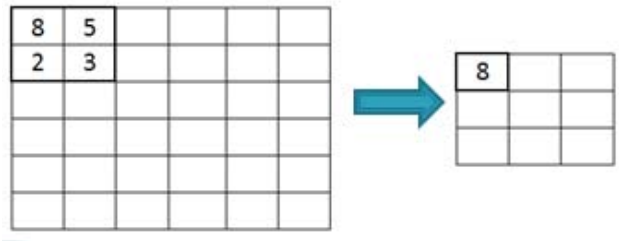

Fig. 2. Example illustrating maximum pooling operation.

B. Image Representatations

The input data which are fed into the network are the raw pixel values of the image. An image can be represented in various different color models. In our experiments we consider three different representations – grayscale, RGB, and YUV.

Grayscale representation contains only intensity or luminance information. A grayscale image, assuming pixel depth of 8 bits, contains pixel values from 0 to 255, presenting a single channel as the input to the neural network. In the RGB model, an image consists of three component images, one for each of the primary colors red, green or blue. Thus, there are three channels to the input of the neural network. The YUV model separates a color image into luminance (Y) and chrominance (U and V) components. YUV is actually a family of color spaces, but generally, in computer video YUV usually refers to the $Y'C_bC_r$ color space [19] where C_b and C_r are the color difference values obtained by subtracting luma Y' from blue and red components respectively In our experiments, we use $Y'C_bC_r$ to represent the YUV model. The relationship between $Y'C_bC_r$ and RGB color models are given by the following equations,

$$Y' = 0.299\ R + 0.587\ G + 0.114\ B \quad (4)$$

$$C_b = B - Y' \quad (5)$$

$$C_r = R - Y' \quad (6)$$

C. Dataset

The MIT Pedestrian dataset [20] was used in the experiments. The dataset consists of 924 color images of people. Each image is 64x128 pixels, containing the whole body of a person aligned to the center of the image. We used the ground truth for gender label as provided by [13], consisting of 600 males and 288 females. There are views of both front (420 images) and rear (468 images) of a person. The images were then cropped to 54x108 by removing equally the border pixels and then resized down to 40x80. Generally, from our initial work, we obtained better results using cropped images compared to the original. Examples of the cropped images from the dataset are shown in Figure 3.

Fig. 3. Example of cropped images from the MIT Pedestrian dataset [20]

Three different image representations as mentioned in the previous section were used in the experiments. The original images in RGB are converted to grayscale and YUV. Additionally, we applied three different normalization methods. The value of the pixels in each channel of an image are scaled down to the range [0,1] or [-1,1], which we refer to as Type I and II respectively. Specifically, for every pixel x in an image,

$$\text{Type I:} \quad x \Leftarrow x / 255 \quad (7)$$

$$\text{Type II:} \quad x \Leftarrow (x - 127.5)/127.5 \quad (8)$$

In the third method, Type III, the pixel values are normalized to zero mean and unit standard deviation. input data which are fed into the network are the raw pixel values of the image. Given an image, the mean μ and standard deviation σ^2 of the pixels are calculated. The following is then applied to every pixel x,

$$\text{Type III:} \quad x \Leftarrow (x - \mu)/\sigma^2 \quad (9)$$

III. EXPERIMENTS

We used the following parameters for the architecture of the convolutional neural network. The size of the input image is 40x80. The number of input channels is one for grayscale image and three for color (RGB and YUV) images.

Filters of size 5x5 are used to obtain 10 feature maps of convolution layer C1, resulting in feature maps of size 36x76. Maximum-pooling operation is applied to 2x2 non-overlapping patches of each feature map, resulting in subsampling layer S2 with feature maps of size 18x38. Layer C3 contains 20 features maps with the size 14x34 produced from 5x5 filters, each connected to all the feature maps in layer S2. Layer S4 contains feature maps of size 7x17 obtained after 2x2 maximum- pooling operation.

Layer F5 contains 25 neuron units which are fully connected to layer S4. The output layer has two softmax activation units for binary classification. All units use hyperbolic tangent activation. The total number of trainable weights including biases is 64,857 when using grayscale image as input and 69,857 when using color images.

For the supervised training of the network, the weights were initialized randomly from a uniform distribution in the range of $[-\sqrt{(6/f)}, \sqrt{(6/f)}]$. Following the suggestion in [21], f equals the number of input connections plus the number of output connections. The weights were updated by back-propagation using mini-batch stochastic gradient descent on the training dataset with randomized order. A small batch size of 4 was chosen, and we tried various learning rates uniformly distributed between 0.01 and 0.2.

Five-fold cross-validation method was applied as follows. The dataset was divided into five folds, with one fold held out as the validation set while the rest formed the training set. After each epoch, the accuracy of the classifier on the validation set was checked. The highest validation set accuracy after 500 epochs was taken as the best result and the mean of the validation results was taken as the overall accuracy.

IV. RESULTS AND DISCUSSIONS

Given that three different image representations were evaluated, each normalized by three different methods, as mentioned in the previous section, this resulted in nine different combinations of datasets to train the convolutional neural network. The results obtained are shown in Table 1, where the lowest average error from five-fold cross validation is shown in each case. Overall, it can be observed that grayscale produces the lowest error, regardless of the type of normalization used. RGB is second best, giving better results compared to YUV.

In Fig. 4, the average error at various learning rates is plotted for the different image representations with Type I normalization. The average error increases by a large amount when the learning rate becomes too large, and slightly when too small.

TABLE I. EXPERIMENTAL RESULTS

Normalization method	Average error (%)		
	grayscale	*RGB*	*YUV*
Unnormalized	23.28	23.73	30.51
Type I	**18.53**	18.76	20.11
Type II	20.00	20.11	20.34
Type III	19.21	20.79	20.90

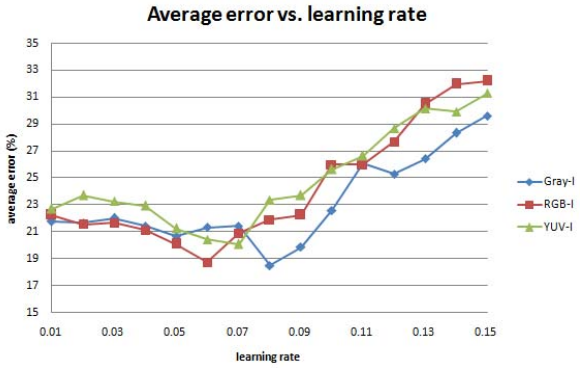

Fig. 4. Plot of average error for Type-I normalization

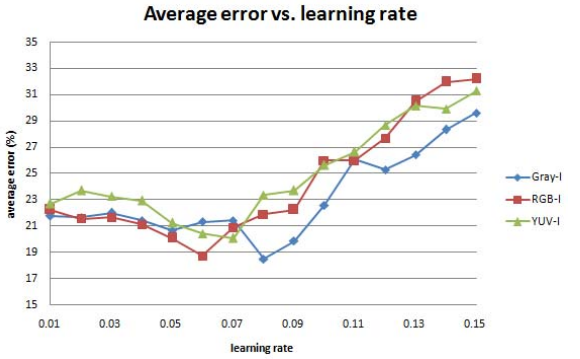

Fig. 5. Plot of average error for grayscale representation

TABLE II. COMPARISON WITH OTHER METHODS

Method	Average accuracy ± std. dev. (%)
Cao et al. [13]	75.0 ± 2.9
Collins et al. [15]	76.0 ± 8.1[a]
Guo et al. [17]	80.6 ± 1.2
Our classifier	**81.5 ± 2.9**

Using frontal view images only

It can also be observed that the best normalization method is Type I, in which the data was scaled to the range [0,1]. It gives the lowest error for all three different image representations. Indeed, some form of normalization seems essential for achieving good performance, as otherwise the average error is significantly higher.

In Fig. 5, the average error at various learning rates is plotted for grayscale representation with different normalization methods. For different learning rates, Type II and III do not have such a large variance of error compared to Type I. This was also observed for the RGB and YUV representations (charts not shown here). In any case, the lowest error is found at different learning rates.

In terms of computation time, grayscale representation, having only one image channel, has the advantage of slightly faster training time compared to RGB or YUV with three image channels. In our implementation using Python with the Theano mathematical computation library [22] running on GPU, it takes an average of 4.8 minutes for a single run of 500 epochs when using grayscale compared to 5.1 minutes using RGB or YUV, approximately 6% faster.

The lowest error overall is 18.53% with normalization to [0,1] and grayscale representation. This corresponds to an accuracy of 81.47 %. As shown in Table 2, the performance of the convolutional neural network is competitive and comparable to other methods applied on the same dataset. To the best of our understanding, these methods also reported the average accuracy from five-fold cross validation.

V. CONCLUSION

In this paper, we have compared three different image representations for training a convolutional neural network to classify the gender of pedestrians. Three different normalization methods were used for preprocessing. From the experiments, grayscale representation produced the best result, followed by RGB and YUV. Using grayscale images normalized to the range of [0,1] resulted in average gender classification accuracy of 81.5% on the MIT pedestrian dataset, which is competitive with previous works on the same dataset. In the future, we plan to attempt more challenging datasets of people in various poses and articulations.

ACKNOWLEDGMENT

C.B.N. gratefully acknowledges the support obtained from UTAR Research Fund and UTAR Staff Scholarship Scheme.

REFERENCES

[1] 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.

[2] M. Riesenhuber and T. Poggio, "Hierarchical models of object recognition in cortex.," *Nature Neuroscience*, vol. 2, no. 11, pp. 1019–25, Nov. 1999.

[3] Y. Bengio, "Learning deep architectures for AI," in *Foundations and Trends in Machine Learning*, vol. 2, no. 1, 2009, pp. 1–127.

[4] G. Hinton, S. Osindero, and Y. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 1554, no. 18, pp. 1527–1554, 2006.

[5] 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.

[6] 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, 1998.

[7] M. Osadchy, Y. Cun, and M. Miller, "Synergistic face detection and pose estimation with energy-based models," *The Journal of Machine Learning Research*, vol. 8, pp. 1197–1215, 2007.

[8] D. Ciresan, U. Meier, J. Masci, and J. Schmidhuber, "A committee of neural networks for traffic sign classification," in *International Joint Conference on Neural Networks*, 2011, pp. 1918–1921.

[9] P. Sermanet and K. Kavukcuoglu, "Pedestrian detection with unsupervised multi-stage feature learning," in *International Conference on Computer Vision and Pattern Recognition*, 2013, in press.

[10] A. Krizhevsky, I. Sutskever, and G. Hinton, "ImageNet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, 2012, vol. 25, pp. 1106–1114.

[11] S. Ji, W. Xu, M. Yang, and K. Yu, "3D convolutional neural networks for human action recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 221–231, 2013.

[12] C. Ng, Y. Tay, and B. Goi, "Recognizing human gender in computer vision: a survey," in *PRICAI 2012: Trends in Artificial Intelligence*, Springer Berlin Heidelberg, 2012, pp. 335–346.

[13] L. Cao, M. Dikmen, Y. Fu, and T. S. Huang, "Gender recognition from body," in *Proceedings of the 16th ACM International Conference on Multimedia*, 2008, pp. 725–728.

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

[15] M. Collins, J. Zhang, and P. Miller, "Full body image feature representations for gender profiling," in *2009 IEEE 12th International Conference on Computer Vision Workshops,* 2009, pp. 1235–1242.

[16] L. Bourdev, S. Maji, and J. Malik, "Describing people: A poselet-based approach to attribute classification," in *2011 IEEE International Conference on Computer Vision*, 2011, pp. 1543–1550.

[17] G. Guo, G. Mu, and Y. Fu, "Gender from body: a biologically-inspired approach with manifold learning," in *Proceedings of the 9th Asian Conference on Computer Vision*, 2009, no. 1, pp. 236–245.

[18] D. Hubel and T. Wiesel, "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex," *The Journal of Physiology*, vol. 160, pp. 106–154, 1962.

[19] "About YUV video," 2013. [Online]. Available: http://msdn.microsoft.com/en-us/library/windows/desktop/ee663260(v=vs.85).aspx.

[20] M. Oren, C. Papageorgiou, P. Sinha, E. Osuna, and T. Poggio, "Pedestrian detection using wavelet templates," in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1997, pp. 193–199.

[21] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2010, vol. 9, pp. 249–256.

[22] J. Bergstra, et al. "Theano: a CPU and GPU math expression compiler," in *Proceedings of the Python for Scientific Computing Conference*, 2010.