# The Effect of Data Augmentation on the Performance of Convolutional Neural Networks

Nelson Marcelo Romero Aquino[1], Matheus Gutoski[2]
Leandro Takeshi Hattori[3] and Heitor Silvério Lopes[4]

Federal University of Technology - Paraná
Av. Sete de Setembro, 3165 - Rebouças CEP 80230-901
[1] nmarceloromero@gmail.com
[2] matheusgutoski@gmail.com
[3] lthattori@gmail.com
[4] hslopes@utfpr.edu.br

**Abstract.** Soft biometrics classification has been gaining acceptance during the recent years for critical applications, mainly in the security field. Recognizing individuals by using only behavioral, physical or psychological characteristics is a task that can be helpful for several purposes. Thus, different Deep Learning approaches have been proposed to perform this task. Since these methods require a large amount of data to avoid overfitting, data augmentation is a commonly used method. However, its isolated effect on the performance of the models are usually not evaluated. This work aims at studying the effect of different data augmentation strategies on the performances of two Convolutional Neural Network architectures for classifying soft biometrics attributes from samples of a novel dataset: LABICv1.

**Keywords:** Deep Learning, Convolutional Neural Networks, Data Augmentation, Unbalanced Datasets

## 1 Introduction

Soft biometrics are physiological or behavioral characteristics from humans that provide information useful to differentiate one individual from another [1]. Although they may not be unique for each subject when compared to traditional biometrics, the correct identification of certain soft biometrics can provide helpful prior information to solve some problems. They can also be used to complement other classic biometric identifiers such as fingerprints, voice and face shapes. Some common soft biometrics are: gender, age, height, weight, clothes color and hair length. Unlike traditional biometrics, soft biometrics can be acquired from surveillance cameras without the cooperation of individuals. Over the years, the number of surveillance cameras in malls, streets and offices have increased massively. However, it is impossible to have humans monitoring all the footages with accuracy and efficiency [1]. Thus, methods for automatically identifying soft biometrics traits of individuals within images are necessary.

Recently, several works appeared, aiming at solving this problem through different strategies, usually Deep Learning (DL) [2] approaches such as Convolutional Neural Networks (CNNs) [3]. For instance, Perlin and Lopes [4] presented two CNNs with the same architecture but with different operation modes: one for classifying three soft biometrics (Upper Clothes, Lower Clothes and Gender) at once and the other for classifying a single soft biometric. The first was trained using the negative log-likelihood as loss function and the second with the mean squared error. In the work presented by Levi and Hassncer [5], age and gender of individuals were classified from images of human faces using a deep CNN, whilst Wang et al. [6] presented an approach based on a 6-layer architecture CNN for feature extraction, in order to estimate age from images containing faces. The works by Zhu et al. [7] and Martinho-Corbishley et al. [8] presented architectures based on slicing images of individuals and feeding each sliced window to different input layers of the network. Each input is propagated through separate Convolution and Pooling layers until reaching the Fully Connected layer, where the outputs of each layer are combined to form an unique flattened vector. Both approaches allow to perform multi-label classification. All works used variations of the Stochastic Gradient Descent (SGD) method [9].

Since DL models require a large amount of data to obtain satisfactory results and soft biometrics datasets are usually small, the use of data augmentation is common to reduce overfitting and improve the classification performance [10, 11]. This method is based on generating new samples of the original dataset by applying small random transformations to the original samples, whilst preserving their labels. Works related to soft biometrics classification are often focused on the architecture of the model without evaluating the particular isolated effect of data augmentation.

This work presents a study regarding the effect of data augmentation on the performance of CNN architectures with different complexities for a small dataset (which can also be unbalanced depending on the attribute to classify). For this purpose, we present a new labeled dataset for soft biometric classification: LABICv1. The dataset contains frames of individuals walking in front of a simple white background. It was built specifically aiming at having a small number of instances considering that obtaining labeled images is a very exhaustive task in real world applications. We trained two models varying the augmentation strategy in order to evaluate their effect on their classification results.

Considering what was presented before, this work has three main contributions: (1) the introduction of a novel small labeled soft biometrics dataset, (2) the comparison of the performances of two CNN architectures for the LABICv1 dataset to serve as baseline for future researches, and (3) the evaluation of the influence of data augmentation on the performances of the classifiers.

This paper is organized as follows: Section 2 presents a brief background regarding CNNs. Section 3 presents details about the LABICv1 dataset, the network architectures that were tested and the data augmentation strategies that were applied. Section 4 reports the results. Finally, Section 5 presents the conclusions and future works.

## 2 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) [12] are composed of Convolutional, Pooling and Fully Connected layers. The first two are present at the start of the network and act as the feature extractor, whilst the fully connected layers are used for data classification. With this structure, a CNN learns not only the classifier but also the feature extractor during its training phase.

Fully connected layers apply the dot product between the output of the previous layer and the network weights connected to them, then a bias term is added to the result and finally a nonlinear activation function is applied to generate an output. In a CNN, the first fully connected layer of the network receives as input a flatten output of the previous layer, which can be a convolutional or a pooling layer.

Convolutional layers are composed of a set of filters that slide across the input and calculate dot products between their entries and the input at any position. This operation produces activation maps containing the outputs of the filters for all the spatial positions of the input. Pooling layers are inserted between convolutional layers to progressively downsample the data representation in order to reduce the amount of parameters of the model. This is done to reduce the computation required in the network and to control overfitting, which occurs when the model is over-trained and loses its generalization capability [13].

There are several methods to avoid overfitting, the most common ones are regularization [14] and dropout [15]. The first one is based on adding an extra term to the cost function of the network in order to force it to preferably learn small weights. Dropout, on the other hand, consists in deactivating a percentage of weights of the network randomly chosen, during the training stage by setting their values to zero.

## 3 Methodology

We evaluate the effect of data augmentation on the performances of two CNN architectures with distinct complexity, which were designed to solve problems related to soft biometrics classification in previous works [4, 5]. Figure 1 presents an overview of our approach. We divided the dataset into train and test sets. Details regarding the construction of the dataset are presented in Section 3.1. Next, various on-line (at each epoch of the training step) data augmentation strategies were applied on the train set, which was fed to each model at the train phase. Finally, their classification capability was tested on the test set using evaluation metrics defined in Section 4. Details regarding the dataset, the architectures, the data augmentation strategies and the configuration of the system are presented in this Section.

### 3.1 LABICv1 Dataset

The dataset introduced in this work contains 264 images taken from frames of 33 videos of 11 different individuals walking in front of a simple white background.

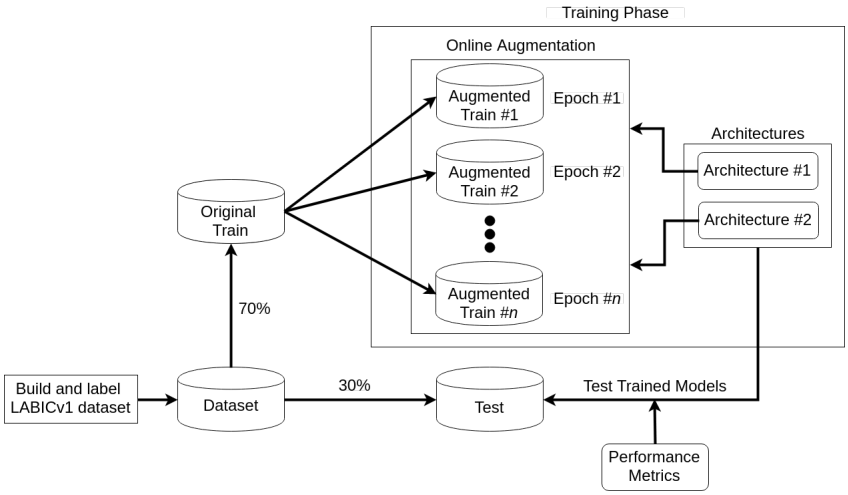



Fig. 1: Overview of the methodology followed in this work. After collecting and labeling the samples of the LABICv1 dataset, we divided it into train (70% of the original dataset) and test (30%) sets. Next, two CNNs with different architectures were trained using augmented versions of the train set, which were created at each epoch of the training phase by applying random transformations on the original set. Once the training phase finished, the classification capability of the models was evaluated using the test set through performance metrics.

Eight frames were selected from each video with a reasonable time gap between them, seeking variety, although the content may be the same. Some samples of the dataset are shown in Figure 2.

Fig. 2: Sample images from the LABICv1 dataset.

The frames of this set are very simple, since they contain only one individual per image, the background is white, there is only one point of view per camera and there is not any type of occlusion. Since it is a simple dataset, it is strongly recommended to apply some type of noise or transformation to add complexity in order to improve the generalization capability of the classifier.

Depending on the label, the dataset may be unbalanced. Figure 3 presents the distribution of the instances for each class according to the attribute. This is an issue tackled through augmentation strategies. In this work, for each label,

the dataset was divided into train (70% of the instances) and test (30%) sets keeping the proportion of classes for each subset. Thus, a new dataset is created for each label, varying the distribution of its instances.

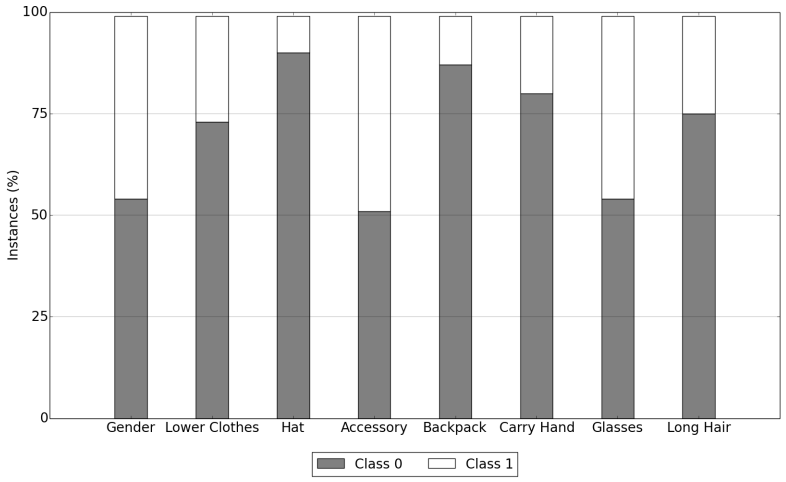



Fig. 3: Distribution of the instances of the LABICv1 dataset for each class according to the label.

### 3.2 Data Augmentation Strategies

We train the models with different strategies based by using data augmentation to improve their classification capability. We also trained and tested the models without any augmentation in order to consider those results as initial baselines. Random transformations were applied on the raw images of the train dataset to augment it (rotation, cropping, swirl, vertical flip, horizontal flip and different types of noises such as salt and pepper and Gaussian noise). The test set remained unchanged in all cases. Two strategies were defined:

**a) Only Augmented** This strategy consisted in augmenting the samples at each epoch of the training phase (on-line data augmentation) without considering if it is balanced or not. We augmented the dataset by 1, 2, 3, 4, 5, 10, 20 and 30 times to evaluate the effect of the size of the train set on the final performances of the models using the test set. Note that the one-time augmentation has the same amount of images as the original dataset, however, with transformations.

**b) Balanced Augmented** This strategy consists in balancing the train set before starting to train the classifier and applying on-line augmentation. The balancing is done by using the same strategies of the data augmentation. Samples from the minority class are selected randomly and augmented until the data is

balanced. From this point on, the strategy is identical to the previous one. The train set was also augmented by 1, 2, 3, 4, 5, 10, 20 and 30 times.

### 3.3 CNN Architectures

All networks are CNNs that receive as input raw images of size $128 \times 128$ with 3 channels, each for a color: Red, Green and Blue (RGB). The final layer outputs a two-dimensional binary vector. All architectures use dropout, with a 50% of keeping probability during the training phase, L2 regularization [16] and Local Response Normalization (LRN) [10] in order to improve the generalization capability of the network. Rectified Linear Unit (ReLU) [17] was used as activation function for all layers except for the output, which is a Softmax function.

**Architecture #1** The first architecture is a modified version of that presented in [4]. It contains three convolutional layers after the input layer, each followed by a max pooling layer. We used zero padding in all cases. One fully connected layer with 768 neurons follows the last pooling layer. Table 1 presents the configuration of each layer of this architecture.

Table 1: Layers of Architecture #1.

| | **Input Maps** | **Input Size** | **Output Maps** | **Output Size** | **Kernel** | **Stride** |
|---|---|---|---|---|---|---|
| **Conv1** | 3 | 128×128 | 32 | 64×64 | 10×10 | 2×2 |
| **MaxPool1** | 32 | 64×64 | 32 | 32×32 | 2×2 | 2×2 |
| **Conv2** | 32 | 32×32 | 64 | 32×32 | 7×7 | 1×1 |
| **MaxPool2** | 64 | 32×32 | 64 | 16×16 | 2×2 | 2×2 |
| **Conv3** | 64 | 16×16 | 128 | 16×16 | 7×7 | 1×1 |
| **MaxPool3** | 128 | 16×16 | 128 | 8×8 | 2×2 | 2×2 |
| **Dense1** | 8192 | - | 768 | - | - | - |
| **Output** | 768 | - | 2 | - | - | - |

The model was trained with SGD and Mean Squared Error (MSE) as cost function, presented in Equation 1, which is the mean of $n$ patterns fed to the model, $\hat{y}$ is the ground truth label, and $y$ is the output of the network for pattern $i$.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{y} - y)_i^2 \tag{1}$$

**Architecture #2** This architecture is based on work presented by [5]. Similarly as Architecture #1, it is composed by three convolutional layers, each followed by a max pooling layer. However, two fully connected layers are placed after the last

pooling layer, both with 512 processing units. Table 2 presents the configuration of each layer of this architecture.

Table 2: Layers of Architecture #2.

| | **Input Maps** | **Input Size** | **Output Maps** | **Output Size** | **Kernel** | **Stride** |
|---|---|---|---|---|---|---|
| **Conv1** | 3 | 128×128 | 96 | 128×128 | 7×7 | 1×1 |
| **MaxPool1** | 96 | 128×128 | 96 | 64×64 | 3×3 | 2×2 |
| **Conv2** | 96 | 64×64 | 256 | 64×64 | 5×5 | 1×1 |
| **MaxPool2** | 256 | 64×64 | 256 | 32×32 | 3×3 | 2×2 |
| **Conv3** | 256 | 32×32 | 384 | 32×32 | 3×3 | 1×1 |
| **MaxPool3** | 384 | 32×32 | 384 | 16×16 | 3×3 | 2×2 |
| **Dense1** | 98304 | - | 512 | - | - | - |
| **Dense2** | 512 | - | 512 | - | - | - |
| **Output** | 512 | - | 2 | - | - | - |

The CNN was trained with SGD and cross-entropy as cost function, presented in Equation 2, where $L_i$ is the loss of the i-th pattern fed to the network, $t_{i,j}$ is the correct label for that pattern and $p_{i,j}$ is the predicted class.

$$L_i = -\sum_j t_{i,j} \log(p_{i,j}) \tag{2}$$

### 3.4 Evaluation Metrics

The results are evaluated with two metrics. The first one is Accuracy, which is calculated by dividing the number of correct classifications $C$ by the number of samples $N$ ($Acc = C/N$). The other metric is the product of Sensibility by Specificity ($Se \times Sp$), presented in Equations 3 and 4 respectively, where $TP$ is the number of True Positives, $FN$ is the number of False Negatives, $TN$ is the number of True Negatives, and $FP$ is the number of False Positives.

$$S_e = \frac{TP}{TP + FN} \tag{3}$$

$$S_p = \frac{TN}{TN + FP} \tag{4}$$

### 3.5 System Setup

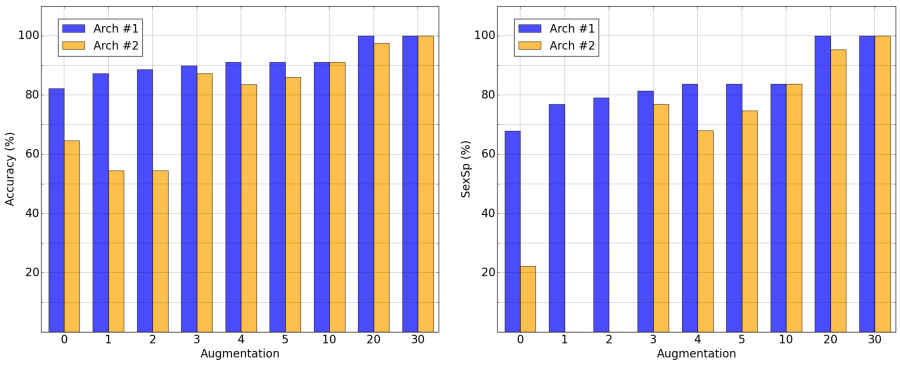
Four GPU boards were used in this work (GTX 1080, Titan X-Pascal, two Titan Xp) for training and testing the models. The GPUs were built within two servers with Intel Core i7-5820K processors and 32Gb RAM memory each. The servers run Ubuntu 14.04.3 LTS. The Python library TensorFlow [18], version 0.11.0, was used to implement, train and visualize the networks.

## 4 Experiments and Results

This section presents the results obtained by the architectures presented in Section 3.3. We present only the study of Gender and Lower Clothes. The categories were selected because they allow to study the effect of data augmentation in two different scenarios related to the distribution of the instances of a dataset: balanced (Gender) and unbalanced (Lower Clothes).

### 4.1 Experiment #1: Gender

In this experiment, the dataset contains 144 samples of the class zero and 120 of the class one, which is quite balanced. Therefore, we consider both *Acc* and $Se \times Sp$ as metrics to evaluate the results for this label. The test set contains 43 samples of the class zero and 36 of the class one after dividing the dataset in train and test sets. Since the dataset is balanced when using this label, the experiments were performed with the Only Augmented strategy (see Section 3.2). Figure 4 presents the results obtained by the two architectures.




a) b)

Fig. 4: Results obtained for the label Gender by the architectures Arch#1 and Arch#2 for each augmentation size. Figure 4a presents the results considering the metric Accuracy, whilst Figure 4b shows the results considering the metric $Se \times Sp$.

Results show that Arch#1 achieved 82.27% of Accuracy, whilst Arch#2 achieved 64.55%, both without any augmentation. Arch#1 tends to improve or at least maintain its performance when the augmentation size rises. Augmenting the training size by 20 and 30 times led to an accuracy of 100% in the test set. Considering the $Se \times Sp$ metric, the tendency is similar, which implies that the model balances its classification capability among both classes without becoming specialized in classifying instances of only one class. Notwithstanding, Arch#2 obtained a baseline result when the train set was doubled and triplied: only 54.43% of accuracy. It correctly classified only instances of the majority class, obtaining 0% of $Se \times Sp$. However, the performance improves as the augmentation size grows: the model achieved 100% of *Acc* and $Se \times Sp$ when the

train set was augmented by 30 times. These results show that Arch#2 needs more data variety than Arch#1 in order to start improving its performance (although it allows to ultimately obtain the same result), which may be due to its higher complexity level.

### 4.2 Experiment #2: Lower Clothes

This dataset is very unbalanced when only the Lower Clothes attribute is considered: class zero contains 195 instances (58 for the test set), whilst class one contains 69 instances (21 for the test set). Therefore, we consider $Se \times Sp$ as the main evaluation metric, since a model capable of correctly classifying instances of both classes even if the overall accuracy decreases is more desirable for unbalanced datasets. Figures 5 and 6 present the results obtained for this experiment. The first one shows those obtained with the Only Augmented strategy, and the second shows those achieved with the Balanced Augmented strategy.

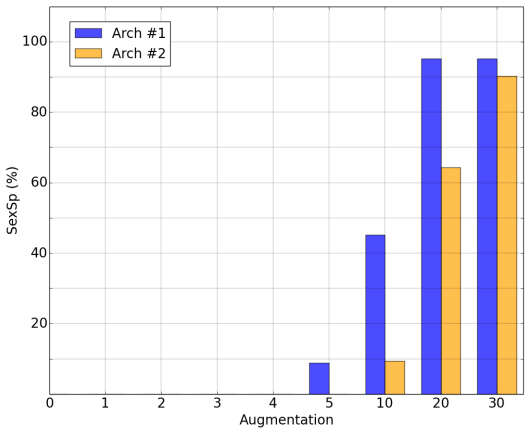



Fig. 5: Results obtained for the label Lower Clothes by the architectures Arch#1 and Arch#2 for each augmentation size. They were obtained without balancing the training dataset before applying the augmentations.

For the Only Augmented strategy, the models tend to classify correctly only instances of the majority class (0% of $Se \times Sp$) even if the train set was augmented by 4 times during the training phase. However, both architectures improve their results as the size of the train set augments through on-line data augmentation. Arch#1 was the fastest to improve its performance, since it achieved 98.73% of $Acc$ and 95.24% of $Se \times Sp$ when the train set was augmented 20 times and maintained that value for an augmentation size equal to 30. The Arch#2 achieved maximum values of 94.94% for $Acc$ and 90.31% for $Se \times Sp$ for a train set augmented 30 times.

In the Balanced Augmented strategy, the performance of the models improve faster, as shown in Figure 6. For instance, the $Acc$ and $Se \times Sp$ values obtained by Arch#1 were 75.95% and 37.68%, respectively, for an augmentation size equal

to 3, whilst their values were 73.41% and 0% (only majority class instances correctly classified) for the same augmentation size.This shows that, for unbalanced datasets, applying augmentation to balance the train set before starting the training phase (combined with on-line data augmentation afterwards) is a valid alternative to improve the generalization capability of the model.

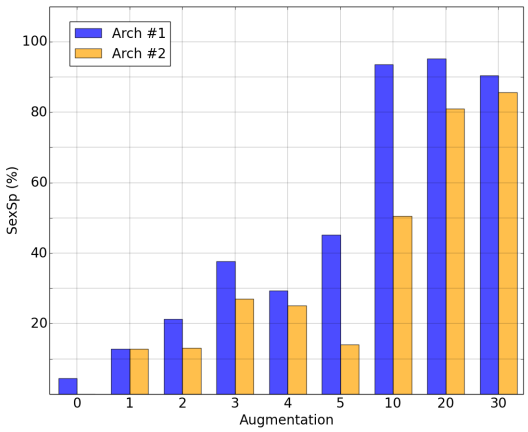



Fig. 6: Results obtained for the label Lower Clothes by the architectures Arch#1 and Arch#2 for each augmentation strategy. They were obtained balancing the training dataset before applying the augmentations.

The results presented in this Section for both experiments, Gender and Lower Clothes, have shown that data augmentation may lead to improved classification results of CNN models for both balanced and unbalanced datasets. For unbalanced datasets, balancing the train set before the training phase may lead to better results with fewer augmented samples compared to the Only Augmented strategy. Results also show that the simpler architecture (Arch#1) needed lower data diversity in order to improve its performance.

## 5 Conclusion

This work aimed at studying the effect of on-line data augmentation on the performance of two Convolutional Neural Networks (CNN) architectures for the classification of soft biometric attributes of a novel dataset. Two augmentation strategies were tested: Only Augmented, which applies on-line data augmentation during the training phase without considering the class balance of the dataset, and Balanced Augmented, which first augments the train set to balance the classes before applying on-line augmentation during the training phase.

This work introduced the LABICv1 dataset, a set of images extracted from videos containing pedestrians with white background. The dataset was manually collected and labeled to be used for supervised learning research projects although unsupervised techniques can be also tested. The dataset provides several soft biometrics attributes. For foster future research using this dataset, it

will be put freely available in the internet. We used the labels Gender and Lower Clothes in this work.

The results obtained show that small augmentation sizes do not have much influence on the performance of the classifiers for both, balanced and unbalanced datasets. Notwithstanding, the Balanced Augmented strategy allows to obtain an improvement of the performance with a smaller augmentation size than the Only Augmented strategy for an unbalanced dataset. We also conclude that a more complex architecture demands a higher augmentation size (more data variety) in order to improve its performance for both approaches tested. Finally, a high augmentation size does not seem to induce overfitting: instead, it allows to improve the generalization capability of the model.

Future works may aim at experimenting with different classification methods or testing different CNN architectures. Other available labels of the dataset may also be used to test the classification capability of the models. Multi-label or multi-class classification approaches can also be explored.

## Acknowledgment

N. M. Romero Aquino thanks the Organization of the American States, the Coimbra Group of Brazilian Universities and the Pan American Health Organization; Author M. Gutoski would like to thank CAPES for the scholarship; Author H.S.Lopes would like to thank to CNPq for the research grant number 440977/2015-0. All authors would like to thank both CAPES and CNPq for the scholarships, as well as NVIDIA for the donation of GPU boards.

## References

[1] Denman, S., Fookes, C., Bialkowski, A., Sridharan, S.: Soft-Biometrics: Unconstrained Authentication in a Surveillance Environment. In: 2009 Digital Image Computing: Techniques and Applications. (2009) 196–203

[2] LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature **521**(7553) (5 2015) 436–444

[3] LeCun, Y., Bengio, Y.: The handbook of brain theory and neural networks. MIT Press, Cambridge, MA, USA (1998) 255–258

[4] Perlin, H.A., Lopes, H.S.: Extracting human attributes using a convolutional neural network approach. Pattern Recognition Letters **68** (2015) 250 – 259

[5] Levi, G., Hassncer, T.: Age and gender classification using convolutional neural networks. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). (June 2015) 34–42

[6] Wang, X., Guo, R., Kambhamettu, C.: Deeply-Learned Feature for Age Estimation. In: 2015 IEEE Winter Conference on Applications of Computer Vision. (2015) 534–541

[7] Zhu, J., Liao, S., Yi, D., Lei, Z., Li, S.Z.: Multi-label CNN based pedestrian attribute learning for soft biometrics. In: 2015 International Conference on Biometrics (ICB). (2015) 535–540
[8] Martinho-Corbishley, D., Nixon, M.S., Carter, J.N.: Retrieving relative soft biometrics for semantic identification. In: 2016 23rd International Conference on Pattern Recognition (ICPR). (2016) 3067–3072
[9] Bottou, L. In: Large-Scale Machine Learning with Stochastic Gradient Descent. Physica-Verlag HD, Heidelberg (2010) 177–186
[10] Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet Classification with Deep Convolutional Neural Networks. In Pereira, F., Burges, C.J.C., Bottou, L., Weinberger, K.Q., eds.: Advances in Neural Information Processing Systems 25. Curran Associates, Inc. (2012) 1097–1105
[11] Chatfield, K., Simonyan, K., Vedaldi, A., Zisserman, A.: Return of the devil in the details: Delving deep into convolutional nets. (2014) 1–11
[12] LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE **86**(11) (1998) 2278–2324
[13] Hung, B.L., Hung, H.A.: Using one-dimensional linear interpolation method to check over-fitting in neural network with multi-dimensional inputs. In: Frontiers of Manufacturing and Design Science IV. Volume 496 of Applied Mechanics and Materials., Trans Tech Publications (2014) 2228–2232
[14] Ng, A.Y.: Feature Selection, L1 vs. L2 Regularization, and Rotational Invariance. In: Proceedings of the Twenty-first International Conference on Machine Learning. ICML '04, New York, NY, USA, ACM (2004) 78–
[15] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: A simple way to prevent neural networks from overfitting. Journal of Machine Learning Research **15**(1) (2014) 1929–1958
[16] Cortes, C., Mohri, M., Rostamizadeh, A.: L2 regularization for learning kernels. In: Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence. UAI '09, AUAI Press (2009) 109–116
[17] Nair, V., Hinton, G.E.: Rectified linear units improve restricted boltzmann machines. In Frnkranz, J., Joachims, T., eds.: Proceedings of the 27th International Conference on Machine Learning (ICML-10), Omnipress (2010) 807–814
[18] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., et al.: TensorFlow: large-scale machine learning on heterogeneous systems. arXiv:1603.04467 (2016) 1–19