Multi-channel Convolutional Neural Network Ensemble for Pedestrian Detection

David Ribeiro[1] **, Gustavo Carneiro[2], Jacinto C. Nascimento[1], and Alexandre Bernardino[1]

[1] Instituto de Sistemas e Robótica, Instituto Superior Técnico, Lisboa, Portugal
david.ribeiro@ist.utl.pt,jan@isr.ist.utl.pt, alex@isr.ist.utl.pt,
[2] Australian Centre for Visual Technologies, The University of Adelaide, Adelaide, Australia
carneiro.gustavo@gmail.com

Abstract. In this paper, we propose an ensemble classification approach to the Pedestrian Detection (PD) problem, resorting to distinct input channels and Convolutional Neural Networks (CNN). This methodology comprises two stages: (*i*) the proposals extraction, and (*ii*) the ensemble classification. In order to obtain the proposals, we apply several detectors specifically developed for the PD task. Afterwards, these proposals are converted into different input channels (e.g. gradient magnitude, LUV or RGB), and classified by each CNN. Finally, several ensemble methods are used to combine the output probabilities of each CNN model. By correctly selecting the best combination strategy, we achieve improvements, comparatively to the single CNN models predictions.

Keywords: Pedestrian Detection,Convolutional Neural Networks, inputs channels, ensemble classification

1 Introduction

Driver assistance systems, autonomous vehicles, robots that interact with humans, and surveillance systems, all of them need to robustly and accurately detect people in order to perform their task correctly. Therefore, Pedestrian Detection (PD) emerges as a relevant and demanding problem, which already has more than ten years of study. The main factors that increase the difficulty of this task are: the illumination settings, the pedestrian's articulations and pose variations, the different types of clothes and accessories, and the occlusions.

As a result of this research, various image channels and feature representations [1] have been proposed (surveys are presented in [2] and [3]). Although the handcrafted features based detectors were popular in the past, the current state-of-the-art detectors rely on Deep Learning architectures, namely Convolutional Neural Networks (CNN). Some of these later models, result from the adaptation

** This work was partially supported by FCT[UID/EEA/50009/2013], and by the FCT project AHACMUP-ERI/HCI/0046/2013.

and extension of object detection frameworks, such as R-CNN [4], Fast R-CNN [5], and Faster R-CNN [6].

Regarding the CNN inputs, the most recent approaches use the RGB images (for example, [7, 8]), but other color spaces (e.g. LUV) and representations (e.g. HOG) have been explored in other recent works [9].

In this paper, we propose an improvement to the pipeline of the R-CNN, and to the PD works of [9, 10]. Two stages can be identified in these methods: 1)the proposal's extraction from the original image, and 2) the CNN post-processing. In our methodology (depicted in Figure 1), we maintain the first stage, by obtaining proposals with conventional and already developed pedestrian detectors, based on handcrafted features. Then, we introduce improvements in the second stage, by using an ensemble of CNN classifiers, instead of a single CNN model. Each CNN is trained with a different input channel, generated from the original RGB proposals (e.g. LUV or gradient magnitude), and applied to the test proposals. Then, the outputs of each of the input channel's CNN models are combined. We observe gains, when comparing the combination's performance with the one from each individual input channel CNN model.

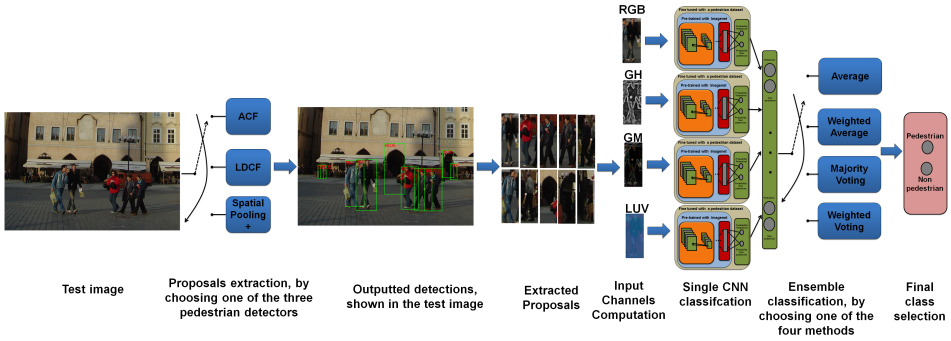

Fig. 1. Pipeline of our method's overall architecture.

2 Proposed Method

As mentioned in Section 1, our methodology (shown in Figure 1) consists of the following two stages: 1) proposals extraction, and 2) multi-channel CNN ensemble classification.

In the first stage, we apply conventional PD detectors to the images, in order to obtain proposals, i.e. regions of interest that might contain pedestrians. More specifically, we resort to the Aggregated Channel Features (`ACF`) [11], the Locally Decorrelated Channel Features (`LDCF`) [12] and the Spatial Pooling + (`SP+`) [13,

14] detectors. This stage performs the multi-scale sliding window task and Non-Maximal-Suppression, which reduces the computational demands imposed to the CNNs, used in the next stage.

In the second stage, we convert the proposals, originally in RGB, to distinct input channels, namely: gradient magnitude, normalized sum of gradient histograms for six orientations[3] and LUV. Each CNN is trained with a different input channel, and applied to the test proposals. Afterwards, the probabilities (of a certain proposal containing a pedestrian), resulting from each of the input channel's CNN models, are combined. This combination is performed according to four different ensemble methodologies [16]: (*i*) averaging, (*ii*) weighted averaging, (*iii*) majority voting, and (*iv*) weighted voting.

2.1 Datasets and Proposals extraction

To reduce the computational demands required by the CNN, we generate proposals with conventional pedestrian detectors, which do not use Deep learning architectures. These detectors are applied to a pedestrian dataset, represented by: $\widetilde{\mathcal{D}} = \{(\widetilde{\mathbf{x}}, \mathcal{B}^{gt})_i\}_{i=1}^{|\widetilde{\mathcal{D}}|}$, with $\widetilde{\mathbf{x}} \in \mathbb{R}^{\tilde{H}\times\tilde{W}\times\tilde{D}}$ denoting the input RGB images (height, width and depth, respectively), and $\mathcal{B}^{gt} = \{\tilde{\mathbf{b}}_k\}_{k=1}^{|\mathcal{B}^{gt}|}$ denoting the set of ground truth bounding boxes containing the pedestrians, with $\tilde{\mathbf{b}}_k = [x_k, y_k, w_k, h_k] \in \mathbb{R}^4$ corresponding to the top-left point and width and height, respectively.

The set of detections provided by the detectors mentioned above, are denoted by $\mathcal{O} = \{(\widetilde{\mathbf{x}}(\mathcal{B}^{dt}), \mathcal{S})_i\}_{i=1}^{|\mathcal{O}|}$, where $\mathcal{B}^{dt} = \{\mathbf{b}_k\}_{k=1}^{|\mathcal{B}^{dt}|}$ are the bounding boxes corresponding to the detections, $\widetilde{\mathbf{x}}(\mathcal{B}^{dt})_i$ constitutes the proposals for image $\widetilde{\mathbf{x}}_i$, i.e. represents the regions of the image $\widetilde{\mathbf{x}}_i$ delimited by the bounding boxes in $\mathcal{B}_i^{dt}$, having $\mathcal{S}_i = \{s_k\}_{k=1}^{|\mathcal{B}^{dt}|}$, with $s_k \in \mathbb{R}$ as the corresponding scores, expressing the confidence in the existence of a pedestrian.

From the output of the detectors, we build a new dataset, where the CNNs are used. We denote this dataset by: $\mathcal{D} = \{(\mathbf{x}, \mathbf{y})_i\}_{i=1}^{|\mathcal{D}|}$, where $\mathbf{x}_i = \widetilde{\mathbf{x}}(\mathcal{B}^{dt})_i \in \mathbb{R}^{H\times W\times D}$ represents the proposals, and $\mathbf{y}_i \in \mathcal{Y} = \{0,1\}^C$ denotes the classes for the proposal $\mathbf{x}_i$: non-pedestrian (label zero) and pedestrian (label one), in our case the number of classes is $C = 2$.

2.2 Input channels

In order to generate several inputs for the CNNs ensemble, we compute three input channels from the original RGB proposals. In total there are four input channels (as illustrated in Figure 2), denoted by: $\mathbf{x}_t \in \mathbb{R}^{H\times W\times D}$, for $t \in T$ ={RGB,GH,GM,LUV}, which correspond to RGB, normalized sum of gradient histograms for six orientations, gradient magnitude (computed from each channel in RGB) and LUV [1]. Since, originally, the normalized sum of gradient histograms for six orientations only has one channel in the third dimension, we replicate this channel to obtain the remaining depth dimensions.

[3] The six orientations are obtained in equally spaced intervals in the range $[0, \pi[$, see details in [15].

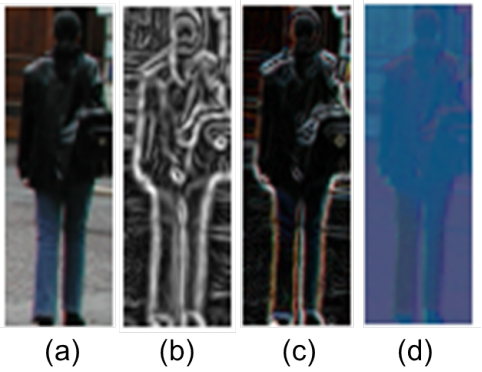

Fig. 2. Depiction of the input channels applied in each CNN model: (a) RGB, (b) normalized sum of gradient histograms for six orientations, (c) gradient magnitude, and (d) LUV.

2.3 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are architectures based on the composition of multiple layers, i.e. the output at layer l is obtained from the input at layer $l-1$ and from the parameters at layer l, according to: $\mathbf{x}^l = f^l(\mathbf{x}^{l-1}, \theta^l)$, where $\theta^l = [\mathbf{w}^l, \mathbf{b}^l]$ denotes the weights and biases (respectively) for the l-th layer out of the total L layers [17].

The initial input at layer $l = 0$, is an image with three channels (height, width and depth, e.g. RGB), i.e. $\mathbf{x}_0 \in \mathbb{R}^{H_0 \times W_0 \times D_0}$, with $D_0 = 3$.

The main operations assumed by these networks at a certain layer l, are represented by the function f^l, and can be of several types: (i) convolutional (followed by an activation function), (ii) pooling, (iii) fully connected (a particular instance of the convolutional one) and (iv) multinomial logistic regression [18].

Therefore, the CNN model $f(.)$ maps each image $\mathbf{x} = \mathbf{x}_0$ to classification probabilities for each class: $p_c = softmax(\mathbf{x}^{L-1}, \theta^L) \in [0, 1]$, where $c = \{0, 1\}$ denotes the non-pedestrian and pedestrian classes (respectively), and the last function is a softmax, defined by: $softmax(\mathbf{x}_c^{L-1}) = \frac{\exp^{\mathbf{x}_c^{L-1}}}{\sum_j \exp^{\mathbf{x}_j^{L-1}}}$, where $c = \{0, \ldots, C-1\}$ denotes the classes (in our case, $C = 2$).

The convolution is defined by:

$$\mathbf{x}^l_{m',n',d'} = \sum_{mnd} \mathbf{w}^l_{m,n,d,d'} \cdot \mathbf{x}^{l-1}_{m+m',n+n',d} + \mathbf{b}^l_{d'}, \tag{1}$$

where m, n, d, index the image height, width and depth, respectively, and d' is the number of filters (see [17]). The convolution is followed by the Rectified Linear Unit (ReLU) activation function [18] defined by $a(\mathbf{x}^l_{m',n',d'}) = max(0, \mathbf{x}^l_{m',n',d'})$

(see [17]). The max pooling is defined by $\mathbf{x}^l_{m,n,d} = max(\mathbf{x}^{l-1}_{m',n',d})$ (see [17]), with $m', n' \in \mathcal{I}(m,n)$, where $\mathcal{I}(m,n)$ are the input locations.

2.4 CNN architecture

Regarding the choice of the CNN architecture and initialization, we have considered the work of [19]. The authors mention the advantages of transfer learning, i.e. using a model pre-trained with an auxiliary dataset, and re-training (i.e. fine-tuning) it with the target dataset, instead of randomly initializing the network.

Consequently, we adopt the configuration D of the VGG Very Deep 16 CNN model [20], which was subject to pre-training with the Imagenet dataset [21] (which contains generic objects). Then, we adapt this architecture to the PD task, by introducing the following changes. First, the original input size is downscaled from $224 \times 224 \times 3$ to $64 \times 64 \times 3$, in order to ease the computational expense associated with the CNN. As a result, we adjust the fully connected layers to this modification, by randomly initializing and resizing them. To obtain the probability of the non-pedestrian and pedestrian classes, the softmax was adjusted to this new number of outputs. Afterwards, we re-train (i.e. fine-tune) this model with the pedestrian dataset, as described in Section 2.1.

The main pre-processing steps required to use the CNN are: the mean subtraction (computed from the training images) and the resize to the defined CNN input dimensions.

2.5 Ensemble classification

The ensemble is composed of four single CNN models, trained with each of the four input channels, described in Section 2.2. Each CNN model constitutes an individual learner $f_t(\mathbf{x}_t)$, $t \in T = \{\text{RGB,GH,GM,LUV}\})$, where $\mathbf{x}_t$ corresponds to the proposal for the t-th input channel, and $f_t(.)$ denotes the t-th CNN model. As mentioned in Section 2.3, each of these CNNs provides a classification score (i.e., a probability) for each class, denoted by $p^c_t = f^c_t(\mathbf{x}_t) \in [0,1]$, where $c = 0$ represents the non-pedestrian class, and $c = 1$ represents the pedestrian class.

To combine the probabilities of the four CNN models, we consider a set G that contains all possible probabilities combinations[4].

For the i-th probability configuration, denoted as $G^i \in G$, four different ensemble methods are used, that is: (i) averaging, (ii) weighted averaging, (iii) majority voting, and (iv) weighted voting [16].

For the average computation, we consider the probabilities p^c_t, and for the voting computation, we transform the probabilities in votes v^c_t, according to: $v^c_t = 0$, if $p^c_t < 0.5$, and $v^c_t = 1$, if $p^c_t \geqslant 0.5$.

For the simple and weighted average, for the class c, we consider the following expression:

$$P^c = \sum_{t \in G^i} w_t \cdot f^c_t(\mathbf{x}_t) = \sum_{t \in G^i} w_t \cdot p^c_t, \tag{2}$$

[4] Considering four input channel's CNN models, the cardinality of G is $|G| = 15$.

where $\sum_{t \in G^i} w_t = 1$. In the simple average case, we have $w_t = \frac{1}{|G^i|}$, and in the weighted average case, we use the CNN model's log average miss rate metric [2] to build the weights, i.e. $w_t = \frac{(1/\mathrm{MR}_t)}{\sum_{t \in G^i} (1/\mathrm{MR}_t)}$, where MR_t denotes the log average miss rate for the input channel's CNN model t.

For the majority voting, the class c is selected, if the number of votes for that class is greater than half the total number of votes. If there is a tie, the pedestrian class ($c = 1$) is selected. The weighted voting is similar to the majority voting, but the votes v_t^c are weighted by w_t, similarly to the weighted average case.

To find which combination of the probabilities (resulting from different CNN models, associated with distinct input channels) achieves the best performance, we try all the possible combination's subsets $G^i \subset G$. Then, for $c = 1$, if $P^c \geqslant 0.5$, or if c has the most votes, we consider that the proposal $\mathbf{x}$ contains a pedestrian, and, consequently, remains unchanged (including the score provided by the detector). Otherwise, the proposal is regarded as a negative, i.e. not enclosing a pedestrian, and is discarded. In fact, we use this final probability to reduce the number of false positives provided by the detector in the first stage, and, therefore, improve the accuracy of the method.

3 Experiments

We perform experiments in the Caltech [2] and INRIA [22] pedestrian datasets, using three detectors for the proposals extraction, namely: LDCF, ACF (ACF-Caltech+ for Caltech as in [12, 15], and mentioned as ACF+) and SP+. All the combination methods described in Section 2.5 are used, for all the possible input channels probabilities combination sets. We adopt the log average miss rate as the performance metric, as described in [2]. We use the following toolboxes: for the CNN [17], for the performance assessment, ACF and LDCF [15], and for SP+, the code provided at [5]. The CNN train settings are: ten epochs with a batch size of 100, a learning rate of 0.001 and a momentum of 0.9.

3.1 Caltech dataset

In order to build the dataset to train the CNN (as described in Section 2.1), we extract positive proposals (i.e. containing pedestrians) using the ground truth annotations from the Caltech dataset with third frame sampling (for further details, see [2]). Afterwards, we augment this set by horizontally flipping each of the proposals. In total, we obtain 32752 positive train proposals.

The negative proposals (i.e. not containing pedestrians) result from applying a non-completely trained version of the LDCF detector to the Caltech dataset with thirtieth frame sampling (for additional information, see [2]). A maximum of five negative proposals are extracted in each image, and the Intersection over Union with the ground truth is restricted to be less than 0.1. By not training

[5] https://github.com/chhshen/pedestrian-detection

this specific LDCF detector with the same amount of data as in [15], we are able to obtain more negative proposals, which correspond to false positive detections (i.e. detector errors). In total, we obtain 17420 negative train proposals.

Finally, the positive and negative proposals collected previously (50172, in total), are split into train (90% of the total amount, i.e. 45155) and validation (10% of the total amount, i.e. 5017) sets.

We assess the performance in the Caltech test set according to the reasonable setting, i.e. the pedestrians must have 50 pixels of height or more, and the occlusion must be partial or inexistent (additional details can be found in [2]).

3.2 INRIA dataset

For the construction of the CNN training set, we resort to the ground truth annotations of the INRIA positive images, in order to obtain positive proposals (i.e. containing pedestrians). Subsequently, this set is expanded by using horizontal flipping. Then, the overall set is further augmented by applying random deformations. In total, we extract 4948 positive proposals.

The negative proposals (i.e. not containing pedestrians) are acquired with a non-completely trained version of the LDCF detector (similarly to the Caltech case), by establishing that only a maximum of 18 negative proposals could be obtained from each of the INRIA negative images. In total, we extract 12552 negative proposals.

Finally, the acquired positive and negative proposals (17500, in total) are split into train (90% of the total amount, i.e. 15751) and validation (10% of the total amount, i.e. 1749) sets.

3.3 Results

Table 1 presents an overview of the results for the INRIA and Caltech datasets by using the proposals extracted with various detectors (column "Proposals"), adding the CNN classification for different input channels (columns from "RGB" to "LUV") and finally, selecting the best combination of the probabilities for these input channels (the column "Inputs" contains the input channels used for the method in the column "Method", to achieve the performance in the column "Comb.").

From Table 1, it can be seen that adding the CNN classification generally improves the baseline detector performance. Although there are some exceptions where the CNN performance is worst than the baseline, we can see that, by correctly choosing the best combination strategy, the proposed ensemble classification is always able to boost the results, reaching the top classification score. This means that the proposed method is effective in the exploration of synergies among individual learners.

Table 1. Log average miss rate % of the detector's proposals, the proposals with CNN classification for distinct input channels, and the best combination of probabilities for several input channels, evaluated using the INRIA and Caltech datasets.

Dataset	Detector	Proposals	RGB	GH	GM	LUV	Comb.	Inputs	Method
INRIA	ACF	16.83	15.24	15.84	16.52	15.55	15.03	{RGB,LUV}	weighted average
	LDCF	13.89	12.43	14.75	13.53	13.29	12.15	{RGB,GM}	weighted average
Caltech	ACF+	29.54	23.23	28.93	28.19	25.09	22.55	{RGB,LUV}	weighted average
	LDCF	25.19	21.49	27.19	26.14	22.42	21.10	{RGB,GH,LUV}	weighted average
	SP+	21.48	16.85	22.94	21.48	19.00	16.60	{RGB,LUV}	simple average

4 Discussion and Conclusions

A novel approach to the PD problem, based on ensemble classification with several input channel's CNN models, is proposed. Comparatively to the individual CNN models performance, we achieve gains by combining the classification results of various CNNs, where each one of them was trained with different input channels (namely, RGB, normalized sum of the gradient histograms, gradient magnitude, and LUV). We show that synergies can be found resorting to ensemble methods, and our approach is easily extensible to more channels and features of different types. As further work, the weights used during the combination could be learned jointly by all CNN models during training, or the CNN features of the top layers could be extracted to train a classification model.

References

1. Dollar, P., Tu, Z., Perona, P., Belongie, S.: Integral channel features. In: BMVC. (2009)
2. Dollár, P., Wojek, C., Schiele, B., Perona, P.: Pedestrian detection: An evaluation of the state of the art. PAMI **34** (2012)
3. Benenson, R., Omran, M., Hosang, J., Schiele, B.: Ten years of pedestrian detection, what have we learned? In: ECCV, CVRSUAD workshop. (2014)
4. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2014)
5. Girshick, R.: Fast R-CNN. In: Proceedings of the International Conference on Computer Vision (ICCV). (2015)
6. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: Towards real-time object detection with region proposal networks. In: Neural Information Processing Systems (NIPS). (2015)
7. Cai, Z., Fan, Q., Feris, R.S., Vasconcelos, N.: A unified multi-scale deep convolutional neural network for fast object detection. In: European Conference on Computer Vision (ECCV). (2016) 354–370
8. Zhang, L., Lin, L., Liang, X., He, K.: Is Faster R-CNN Doing Well for Pedestrian Detection? In: European Conference on Computer Vision (ECCV). (2016)

9. Hosang, J., Omran, M., Benenson, R., Schiele, B.: Taking a deeper look at pedestrians. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2015) 4073–4082
10. Ribeiro, D., Nascimento, J.C., Bernardino, A., Carneiro, G.: Improving the performance of pedestrian detectors using convolutional learning. Pattern Recognition **61** (2017) 641 – 649
11. Dollár, P., Appel, R., Belongie, S., Perona, P.: Fast Feature Pyramids for Object Detection. PAMI (2014)
12. Nam, W., Dollár, P., Han, J.H.: Local decorrelation for improved pedestrian detection. In: NIPS. (2014)
13. Paisitkriangkrai, S., Shen, C., van den Hengel, A.: Strengthening the effectiveness of pedestrian detection with spatially pooled features. In: ECCV. (2014)
14. Paisitkriangkrai, S., Shen, C., van den Hengel, A.: Pedestrian detection with spatially pooled features and structured ensemble learning. IEEE Transactions on Pattern Analysis and Machine Intelligence **38**(6) (2016) 1243–1257
15. Dollár, P.: Piotr's Computer Vision Matlab Toolbox (PMT). http://vision.ucsd.edu/ pdollar/toolbox/doc/index.html
16. Zhou, Z.H.: Ensemble methods: Foundations and algorithms (2012)
17. Vedaldi, A., Lenc, K.: MatConvNet – Convolutional Neural Networks for MATLAB. In: Proceeding of the ACM Int. Conf. on Multimedia. (2015)
18. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NIPS. (2012)
19. Yosinski, J., Clune, J., Bengio, Y., Lipson, H.: How transferable are features in deep neural networks? In: NIPS. (2014) 3320–3328
20. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: ICLR. (2015)
21. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.: ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision (IJCV) **115**(3) (2015) 211–252
22. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: CVPR. (2005)