A Convolutional Neural Network that Self-Contained Counts

Oliver Urbann and Jonas Stenzel
Fraunhofer IML, Dortmund, Germany
Email: {oliver.urbann, Jonas.stenzel}@iml.fraunhofer.de

***Abstract*—We propose a Convolutional Neural Network for counting objects and persons in images. Utilizing a sequence of images, it is possible to count with respect to a movement direction (e.g. UCSD pedestrian dataset). The proposed Number Convolutional Neural Network (NCNN) directly outputs the desired count and thus does not require additional counting steps or dense maps as intermediate step. It cannot only be superior with respect to the mean absolute error evaluated on already known datasets. It also requires only the count as ground truth data and is thus easily and quickly applied to a variety of new problem statements. Additionally, it is able to count with respect to a movement direction by integrating time-dependent information.**

***Index Terms*—deep learning, CNN, crowd counting**

I. Introduction

In vision research and application Convolutional Neural Networks (CNN) are a famous approach for various kinds of problems. The most common utilization is for classification of images, e.g. the widely known implementation called AlexNet [1]. However, in this paper we are interested in counting people or objects in single static images or with respect to a movement.

A. Related Work

A popular approach for counting people is the method proposed by Chan *et al.* [2]. It is not limited to a static count, i.e. counting without involving time on single images. The solution is able to count people walking in different directions, which we call here a dynamic count. However, they show that the accuracy depends on various features selected and composed for that particular problem and modelled as a trained Gaussian process.

Lempitsky *et al.* [3] propose to learn a linear mapping from local features in images to their density maps of objects. A density map for counting persons generated by a regression random forest is proposed by Fiaschi *et al.* [4]. Unfortunately, density maps cannot be used for dynamic counting and thus these approaches are limited to static counts.

All three approaches mentioned above are presented for comparison with our approach in the evaluation.

A more general approach for counting would be to detect all persons in the image as the first step. This could be done by applying the proposed method by Li *et al.* [5] which is a fast approach applying CNN for face detection. However, such approaches would require visible faces in a sufficient resolution.

Thus, counting utilizing CNN is usually done by detecting crowds in density maps as e.g. proposed by Zhang *et al.* [6], [7]. In the former mentioned paper, they propose to switch the learning process depending on the input and to count utilizing a ridge regressor for density estimation first. In their latter paper, they propose to switch between different CNNs instead of different trainings. The density maps are then combined which is afterwards used for counting in a subsequent step.

This is a more general approach as it could be adapted to various kinds of objects but is limited to a static count as well. The result can be improved by applying an additional CNN to select a net that best suits the crowd density in the given image [8] instead of combining the results. It is also possible to match small patches to object densities by utilizing a CNN as proposed by [9]. An overview about related approaches is given in [10].

B. Contribution

To the best of our knowledge, all approaches utilizing a CNN are focused on density maps and thus cannot count dynamically. Zhang at el. mentioned two natural approaches for a CNN to count [7]. First is the density map, second is the number as direct output. They argue for following the first idea and so to neglect the second. We see reasonable advantages of the latter idea:

- As ground truth data only the count is required. Thus, labeling an image takes only a short time.
- Counting is done by the system without requiring object specific methods, e.g. features especially for detecting persons.
- It is not limited to a static count (see above).

The goal of this paper is not only to further improve the already excellent accuracies of the approaches mentioned above. We also propose a CNN that directly counts static and dynamic persons and objects. It is thus easily and quickly applied to new problem definitions and we call it the Number Convolutional Neural Network (NCNN).

For a dynamic count informations about the movement must be visible in the images.

We thus propose a method to enable a dynamic count using a single image with three channels for three different time instances.

Manuscript received July 23, 2019; revised November 7, 2019.

doi: 10.18178/joig.7.4.112-116

Manually generating ground truth data is fast as mentioned above. However, a large training set is required to cover all possible situations in all image areas. We therefore also show how to generate additional training data by shuffling parts of an image and recombine these to a new image.

In the evaluation, we compare this approach with other methods utilizing already known datasets, namely the UCSD pedestrian dataset [11] and the TRANCOS v3 data set [12]. We conclude under which circumstances the proposed NCNN is superior to other approaches regarding Mean Absolute Error (MAE) and deduce the requirement for a broad training data set for this approach.

C. Overview

The paper is organized as follows. In the next section II we present the NCNN for static (Sec. II-A) and dynamic counting (Sec. II-B) where we also propose the concept for including the required time-dependent information in movies like walking directions. Afterwards in Sec. II-C we present our algorithm to generate additional training data.

In the evaluation in Sec. III we apply the methods to the UCSD pedestrian dataset (Sec. III-B) for static and dynamic counting and afterwards in Sec. III-C to the TRANCOS dataset for static counting. The conclusion (Sec. IV) utilizes the outcome of the evaluation and summarizes the methods.

II. Number Convolutional Neural Network

The Number Convolutional Neural Network (NCNN) proposed in this paper is based on the idea that a CNN is composed of multiple well-known mathematical operations. The counting step by detecting objects or counting in crowd/density maps can be integrated into the net due to the aforementioned general nature of a CNN. The input of the NCNN is thus the image with people or objects to count. The output is the count as a scalar.

Fig. 1 depicts the NCNN. It is simple and small and thus very fast compared to the CNN mentioned in Sec. I-A. One convolutional layer is sufficient, where $c_x \times c_y \times c_d$ denotes the size and depth of the convolutional filters and c_N the number of filters. The filter size should be chosen according to the resolution of the input images. As the evaluation in Sec. III shows, smaller resolutions benefit from smaller filter sizes (e.g. $c_x \times c_y = 3 \times 3$) both in MAE and speed while the size should be higher (e.g. $c_x \times c_y = 15 \times 15$) for a resolution above 640×480. The depth of each filter c_d is 1 for static counting and 3 for dynamic counting, see Sec. II-B. Additionally, the number of output scalars is variable as well depending on the number of different classes to be counted.

The convolutional layer is followed by a layer with rectifying linear units and a max-pooling layer with a 2x2 kernel and stride 2. The next layer is a dense layer with 10 neurons and again rectifying linear units. The last layer is the counting layer with variable number of neurons, depending on the number of desired scalars.

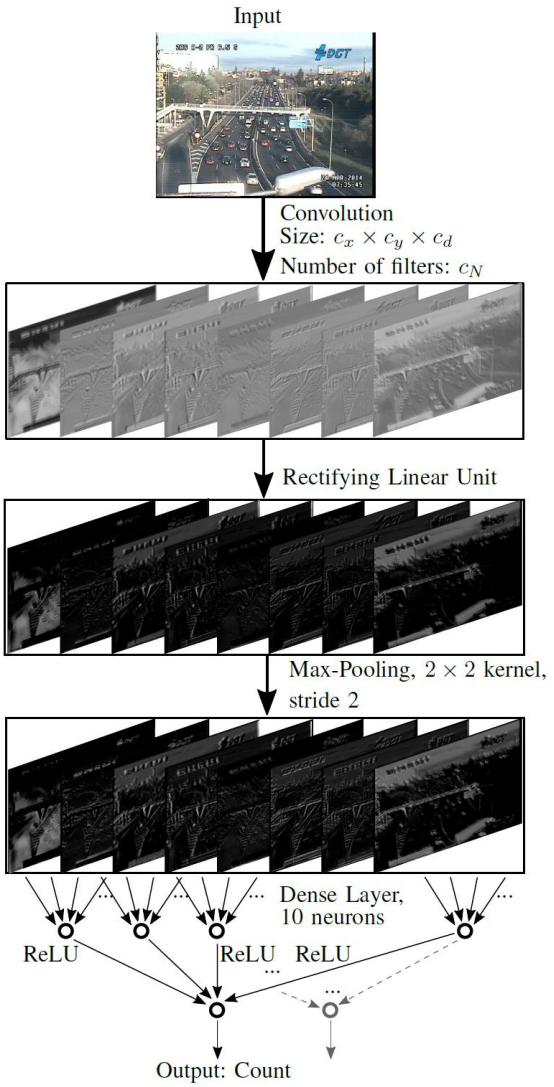

Figure 1. The size $c_x \times c_y$ of the convolutional filter should be adapted to the resolution of the input images. The depth c_d should be 1 for a static count and 3 for dynamic counting, see Sec. II-B. The number of filter c_N depends on the training data.

A. Static Count

Static count denotes counting objects or persons in a single image without respect to a possible movement. Even though walking people have a leg and arm position that indicates their walking direction, we propose a different approach to include this in the next section.

To this end we apply the NCNN structure depicted in 1 with a filter depth of $c_d = 1$ in the convolutional layer for grayscale images as input. The number of output neurons is one: the scalar estimating the count.

B. Dynamic Count

A dynamic count of people as explained in Sec. I-A could be done by training to detect typical arm and leg movements or the direction of the face indicating the walking direction. However, in the given datasets the objects and persons are rather small, such that the limbs and faces are not always clearly visible.

Therefore, we propose a slightly different structure of the NCNN compared to the static count. The filter depth of the first layer is now three, which is commonly the case for colored images. Here the three input channels have a different meaning. Assuming we want to count at time t, the grayscale image at t is the second channel of the input image. The first channel is then the grayscale image at time $t - \Delta t$ and the third channel is the grayscale image at time $t + \Delta t$. Interpreting this image as a colored BGR image leads to Fig. 2 where we set $\Delta t = 1\text{s}$. In a grayscale image people are dark compared to the walkway which leads to a lower value in the corresponding channel. In an area without movement the image is thus gray as all channels have the same value. If for example at time $t - \Delta t$ a person is in that area the blue channel has a lower value resulting in yellow. Thus, the direction of a walking person is identifiable by a set of color transitions.

Figure 2. Image of walking people in the UCSD pedestrian dataset [11] using the image channels for different time instances.

The output of the NCNN are two scalars for the dynamic count, one for each direction comparably to the approach in [2].

C. Image Shuffling

CNN are typically applicable to images that are comparable to images in the training data. While convolutional layer work with a sliding window applying the same operation in all areas of the image, the matrix of a dense layer has different gains for the pixels in the image. To train all entries of the matrix correctly we thus need all possible situations in all areas of the training images.

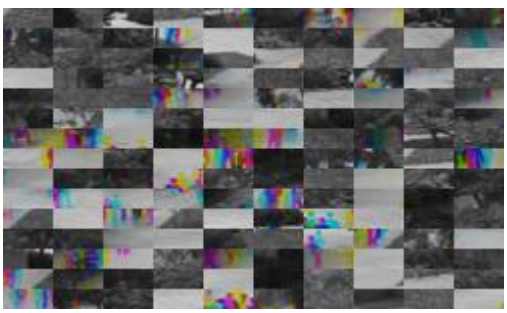

Figure 3. Image composed as a mixture of a given source image to generate additional training data.

We therefore apply the following method to generate additional training data. We cut an image in smaller parts (e.g. patches with size 25x10 pixel) and reorganize them randomly to generate a new image, see Fig. 3. As one of the goals of this paper is to require only simple ground truth data (i.e. the count), we do not know where persons or objects are in the source image. Thus, it is possible that persons/objects are cut and placed in different locations in the target image, which can reduce the quality for training.

III. Evaluation

The goal of this section is to evaluate the objectives described in Sec. I-B. We are especially interested in the ability to count persons and objects statically and dynamically without changing the setup.

It is known that CNN are impractical on datasets that require extrapolation, i.e. if the training data does not cover all possible situations. As our approach is based on a CNN only, it is prone to this and we are interested in how it performs under these circumstances and if the generation of additional training data as explained in Sec. II-C can improve the results.

We utilize two challenging datasets for which already precise solutions exist. First one is the UCSD pedestrian dataset [11] that consists of 2000 images of a walkway at the University of California in San Diego. The dataset provides ground truth data and contains the total person count useful for the static count evaluation. Furthermore, the number of people walking towards the camera und away from the camera is also contained which we utilize for the evaluation of the dynamic count. Additionally, we use the TRANCOS dataset with 1244 images of traffic scenes recorded by surveillance cameras.

A. Setup

Our CNN training and inference is based on Tensorflow [13]. All source images are grayscale with the mean value per image subtracted. For the UCSD pedestrian dataset a Region of Interest (ROI) is usually defined. All pixel outside this region are set black. As cost function we use the mean squared error between the ground truth and the estimated count. The number of scalars in this count depends on the desired type of count, i.e. static or dynamic count, here one and two respectively. The Adam optimizer proposed by Kingma *et al.* [14] leads to the best results compared to e.g. Adagrad algorithm [15] or Gradient Descent. Learning rates and batch sizes are adapted to image size and shape of the cost function. All variables are initialized by random numbers, also the permutation used by the image shuffling is random. However, to allow to reproduce all results, we set the random number generator seed to the random but fixed number 42.

B. UCSD Pedestrians

The resolution of the images is 238x158. However, we resize all images to 250x150 to have a larger variety of possible patch sizes for shuffling (Sec. II-C). At this resolution a convolutional filter size of $c_x \times c_y = 3 \times 3$ is sufficient.

We compare training sets with additional shuffled images in different patch resolutions with no additional

images. The comparison is done using the MAE after 10 training epochs on the evaluation images. We finally selected a patch resolution of 25x10 and add 40 shuffled images per source image to the training set.

In [2] frames 600 to 1399 are utilized for training, the remaining images for evaluation. As the authors correctly state, this split allows to test the extrapolation ability of the approach.

TABLE I. MAE OF DIFFERENT APPROACHES COUNTING ON THE UCSD PEDESTRIAN DATASET AS DEFINED IN [2]. $NCNN_1$ DENOTES THE NCNN WITH TRAINING IMAGES AS PROVIDED IN [2] AND $NCNN_2$ WITH ADDITIONAL SHUFFLED IMAGES AS EXPLAINED IN SEC. II-C

Appr.	Static Count	Dynamic Count "Away"	Dynamic Count "Towards"
[2]	N/A	**1.621**	**0.869**
[16]	N/A	1.808	1.343
[17]	N/A	1.995	1.108
[6]	1.6	N/A	N/A
[7]	**1.07**	N/A	N/A
$NCNN_1$	3.493	3.272	3.041
$NCNN_2$	2.585	2.653	2.889

Fig. 4 depicts the MAE using the evaluation images during the NCNN training. As can be seen ("Normal"), shortly after 20 training epochs no further improvements can be observed. Table I shows the MAE of different known approaches to statically counted persons. The performance of the proposed NCNN trained as described above leads to a low performance after 100 training epochs. Extending the training dataset by images shuffled as explained in Sec. II-C improves this result as Fig. 4 ("Shuffled") and Table I shows. However, it is still below state of the art methods.

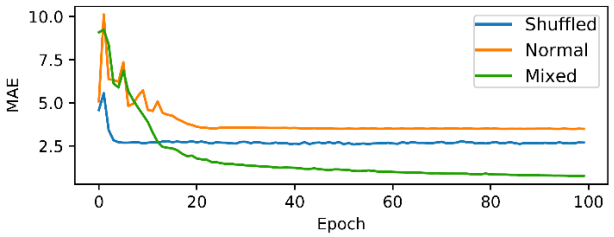

Figure 4. MAE using the evaluation image set of the UCSD pedestrian dataset of three different training setups. "Normal" denotes an experimental setup as in [2], "Shuffled" denotes our training set extension as explained in Sec. II-C and "Mix" denotes a random mix of all images before splitting into training and evaluation images.

To confirm that a missing extrapolation capability is causal, we conduct an experiment where all 2000 images are mixed randomly before 40% are chosen for training. Fig. 4 shows this training ("Mixed"). As can be seen, the MAE is vastly improved to 0.7525 for static count and 0.402/0.472 for the dynamic count. However, we do not add this result to Table I as this experimental setup is not comparable to the one above.

An interesting observation during these trainings is that the number of filters in the convolutional layer should depend on the generality of the training set. Generally speaking, a CNN should be small with a low number of filters to be fast. Thus, we use two filters in the convolutional layer in case of a training as described in [2]. However, our "mixed" setup leads to a more general training set and a larger number of filters can benefit from this. Therefore, we use 8 convolutional filters in this case.

C. TRANCOS

The TRANCOS dataset contains 1244 images in various resolutions of different traffic scenes for counting cars, see Fig. 5 for an example. It is split into a training set with 403 images, an evaluation set with 420 images and a test set with 421 images. We trained our NCNN with the training images only and utilized the evaluation set to improve the shape of the convolutional layer and to choose training parameters. The test set is used only for reporting the results.

Figure 5. Example picture of the TRANCOS dataset for counting vehicles, here 27 according to ground truth data.

The training set is well distributed and thus, as described above, we use 8 filters in the first layer and do not use shuffling. We resize all images to a unified resolution of 640x480 and set the filter size to 15x15, suitably for the higher resolution of the images compared to the UCSD pedestrian dataset.

TABLE II. MAE OF DIFFERENT APPROACHES COUNTING CARS ON THE TRANCOS DATASET [12]

Approach	MAE
[4]	17.77
[3]	13.76
CCNN [9]	12.49
Hydra 2s [9]	11.41
Hydra 3s [9]	10.99
Hydra 4s [9]	12.92
NCNN	**10.79**

After training 100 epochs we found the lowest MAE using the evaluation images at epoch 18. Later epochs seem to be overfitted. We decided to use this model (trained for 18 epochs) and achieve an MAE on the test images of 10.791 which outperforms all known methods applied to this dataset, see Table II. Note that due to our approach we can only give the GAME(0) metric [9] which is the MAE.

IV. CONCLUSION

This paper proposes a possibly evident solution for counting objects and persons utilizing a CNN, the

Number CNN (NCNN). It's distinction to other methods involving a CNN is the self-contained approach by directly giving the desired number as an output of the net. Thus, it is a very simple approach that can be quickly adapted to new problem statements and requires only training images with counts as ground truth data. While being simple it is able to count with respect to a moving direction which is uncommon for a CNN utilized for counting.

As the evaluation points out, a self-contained counting solution based mostly on machine learning reveals poor extrapolation capabilities. Using well-formed training data the evaluation shows that the proposed NCNN outperforms state of the art methods comparing the mean absolute error.

Future work should not specialize the NCNN for particular problem statements like person counting to improve performance. Rather research must be conducted to further generalize the training data beyond the capabilities of the image shuffling utilized here. This should not require other or new labels for training data to keep the generality of NCNN. Thus, Generative Adversarial Networks [18] are a promising approach.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems*, pp. 1097-1105, 2012.

[2] A. B. Chan, Z. S. J. Liang, and N. Vasconcelos, "Privacy preserving crowd monitoring: Counting people without people models or tracking," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2008, pp. 1-7.

[3] V. Lempitsky and A. Zisserman, "Learning to count objects in images," *Advances in Neural Information Processing Systems*, pp. 1324-1332, 2010.

[4] L. Fiaschi, U. Köthe, R. Nair, and F. A. Hamprecht, "Learning to count with regression forest and structured labels," in *Proc. 21st International Conference on Pattern Recognition*, 2012, pp. 2685-2688.

[5] H. Li, Z. Lin, X. Shen, J. Brandt, and G. Hua, "A convolutional neural network cascade for face detection," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 5325-5334.

[6] C. Zhang, H. Li, X. Wang, and X. Yang, "Cross-scene crowd counting via deep convolutional neural networks," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 833-841.

[7] Y. Zhang, D. Zhou, S. Chen, S. Gao, and Y. Ma, "Single-image crowd counting via multi-column convolutional neural network," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 589-597.

[8] D. B. Sam, S. Surya, and R. V. Babu, "Switching convolutional neural network for crowd counting," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2017, vol. 1, p. 6.

[9] D. Onoro-Rubio and R. J. López-Sastre, "Towards perspective-free object counting with deep learning," in *Proc. European Conference on Computer Vision*, 2016, pp. 615-629.

[10] V. A. Sindagi and V. M. Patel, "A survey of recent advances in cnnbased single image crowd counting and density estimation," *Pattern Recognition Letters*, vol. 107, pp. 3-16, 2018.

[11] A. B. Chan and N. Vasconcelos, "Modeling, clustering, and segmenting video with mixtures of dynamic textures," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 5, pp. 909-926, 2008.

[12] R. Guerrero-Gómez-Olmedo, R. Beatriz, L. S. Torre-Jiménez, S. M. Bascón, and D. Onoro-Rubio, "Extremely overlapping vehicle counting," in *Proc. Iberian Conference on Pattern Recognition and Image Analysis*, 2015.

[13] M. Abadi, *et al.*, "Tensorflow: Largescale machine learning on heterogeneous distributed systems," arXiv preprint arXiv:1603.04467, 2016.

[14] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," arXiv preprint arXiv:1412.6980, 2014.

[15] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, pp. 2121-2159, 2011.

[16] D. Kong, D. Gray, and H. Tao, "Counting pedestrians in crowds using viewpoint invariant training," in *Proc. BMVC*, 2005, pp. 1-6.

[17] A. C. Davies, J. H. Yin, and S. A. Velastin, "Crowd monitoring using image processing," *Electronics & Communication Engineering Journal*, vol. 7, no. 1, pp. 37-47, 1995.

[18] I. Goodfellow, *et al.*, "Generative adversarial nets," *Advances in Neural Information Processing Systems*, pp. 2672-2680, 2014.

Oliver Urbann received his PhD in computer science in 2017 at the Robotics Research Institute for his research on biped walking robots by applying model based control and observer for sensor feedback. He won several awards at world championships for robotic soccer, e.g. the world championship on RoboCup 2016 for outdoor soccer in the Standard Platform League. Afterwards, he co-founded a startup for intelligent observer, specialized on machine learning and embedded real-time vision systems. Currently, he is employed at Fraunhofer IML in Dortmund, Germany, as Senior Research Scientist for AI and responsible for machine learning under resource constraints.

Jonas Stenzel is team leader and PhD student at Fraunhofer IML, Dortmund, Germany. He is interested in computer vision and automation in the industrial domain.