CNN Ensemble Robust to Rotation Using Radon Transform

Václav Košík, Tomáš Karella and Jan Flusser
Czech Academy of Sciences
Institute of Information Theory and Automation
Pod vodárenskou věží 4, 182 00, Prague, Czechia
{kosik, karella, flusser}@utia.cas.cz

***Abstract*—A great deal of attention has been paid to alternative techniques to data augmentation in the literature. Their goal is to make convolutional neural networks (CNNs) invariant or at least robust to various transformations. In this paper, we present an ensemble model combining a classic CNN with an invariant CNN where both were trained without any augmentation. The goal is to preserve the performance of the classic CNN on nondeformed images (where it is supposed to classify more accurately) and the performance of the invariant CNN on deformed images (where it is the other way around). The combination is controlled by another network which outputs a coefficient that determines the fusion rule of the two networks. The auxiliary network is trained to output the coefficient depending on the intensity of the image deformation. In the experiments, we focus on rotation as a simple and most frequently studied case of transformation. In addition, we present a network invariant to rotation that is fed with the Radon transform of the input images. The performance of this network is tested on rotated MNIST and is further used in the ensemble whose performance is demonstrated on the CIFAR-10 dataset.**

***Index Terms*—CNN, rotation invariance, equivariance, Radon transform, network fusion, network ensemble**

I. Introduction

Since the introduction of AlexNet [1] in 2012, convolutional neural networks (CNNs) have started dominating visual recognition and replacing handcrafted features. Although they have achieved tremendous and stunning success in many fields of computer vision, they still face many challenges. One of them is their noninvariance under many basic image transformations frequently occurring in real situations. It is a well-known fact that CNNs perform very poorly on images deformed by scale, rotation, blur, and other transformations if they did not occur in the training set. Yet, a transformed object is still the same object. This problem is most often tackled by augmenting the dataset, which can prolong the training dramatically, worsen the results on non-transformed images, and has also other issues. Hence, an enormous amount of research work has been devoted to designing alternative approaches to augmentation.

In this paper, we present an ensemble model that combines a traditional CNN with a CNN modified to be fully invariant under certain group of transformations. The reasoning behind this ensemble is that the latter CNN was modified specially for the transformation and therefore, it is expected to perform worse when this transformation does not occur, but better when the transformation is met at significant intensity. While totally avoiding augmentation, the goal is to ideally preserve the performance of the traditional CNN on nondeformed images and at the same time to achieve at least the performance of the invariant CNN on deformed images. The ensemble combines the two neural networks at the score level. The core idea of the combination is to estimate the deformation intensity of an image that is passed to the ensemble as input. This requires the inclusion of another model for the estimation that needs to be trained on the same dataset where all images are deformed by different levels of the transformation. Since one of major drawbacks of data augmentation is a much longer training time, this estimation model should preferably be easier and faster to train than the two CNNs for the actual task.

The proposed ensemble idea is general and we formulate it so that it can be applied to any transformation. Since one of the most common (and most investigated in the literature) deformations is rotation, we used rotation as a deformation example in our experiments. For this purpose, we invent a CNN that is fully invariant under rotation and uses the Radon transform of an image as input. The idea is similar to works that use images in polar coordinates [2]–[4] – a rotation is transformed into a circular shift along an axis, which is then preserved between convolutional layers. Unlike the polar transform, the Radon transform does not suffer from unequal sample density in different parts of an image. We show that the performance of the Radon transform is superior to the polar transform on rotated MNIST. To the best of our knowledge, nobody used the Radon transform to tackle the problem of rotational noninvariance of CNNs before.

A special kind of research work dedicated to this subject seeks to achieve the so-called equivariance of feature mapping to a group of transformations (which can be, for instance, a group of rotations). We say that a feature mapping f is equivariant to G if

$$f(gI) = g' f(I) \text{ for all } g \in G$$

where I is either an image or a feature map and g' belongs to a group G' of transformations and can be determined if we know g. This means that if an image is transformed, the feature maps behave in a predictable way. If we slightly adjust the padding in convolutional layers, then a CNN based on the Radon transform has this property since any rotation results in a cyclic shift of the transform, which is then preserved between

979-8-3503-2541-6/23/$31.00 ©2023 IEEE

convolutional layers, and the final invariance can be achieved by eliminating the spatial dimension.

In the following section, we refer to the literature devoted to CNN modifications that make the models robust to a transformation. We focus especially on rotation. In Section III-A, our proposed ensemble is defined and explained for a general transformation. A continuous definition of the Radon transform, its discretization and properties that we need are covered in Section III-B. A CNN using the Radon transform of an image requires two modifications to be invariant under rotation. These are explained in Section III-C. Finally, we perform experiments in Section IV. First, we show the effectiveness of our invariant CNN on rotated MNIST and also its superiority to invariant CNNs using polar coordinates. Then, we train a classic CNN and an invariant CNN with the same architectures on the MNIST dataset without any augmentation and examine the impact of rotations on both networks. MNIST is an example of a dataset which is too simple for applying the proposed ensemble, as the invariant CNN is similarly good on nonrotated images and much better on rotated images. That is not the case of CIFAR-10 where we apply the proposed ensemble and show that it almost achieves the maximum accuracy of both models for any rotation of the test set.

II. Related Work

There are many publications dealing with the impact of transformations on convolutional neural networks such as scale [5], [6], blur [7], [8] or even affine transformation. For instance, the popular Spatial Transformer Network [9] by Jaderberg et al. brings inputs to a standard position before sending them to a classification network, and it can deal with a general affine transformation.

Since our experiments are focused on rotations, we cover the literature with the same aim in the remainder of the section.

In 2016, Cohen and Welling introduced a popular concept of G-CNN [10], a neural network equivariant to a discrete group of transformations. The idea is general, but is tested on the group of mirror reflections and 90° rotations, where the equivariance is achieved by rotating and mirroring convolutional filters. The major drawback of this approach is the discreetness – equivariance is designed only for rotations by multiples of an angle, and increasing the group size increases the computational cost.

A special approach to constructing equivariant networks uses steerable filters introduced in [11]. Steerability puts constraints on filters so that they can be written as a linear combination of a finite set of filters. The first paper that presented the theory of steerable filters in CNN was written by Cohen and Welling [12]. The publication [13] proposed steerable filters based on complex circular harmonics. The E(2) - Equivariant Steerable CNN designed in [14] achieved state of the art on rotated MNIST [15], a dataset used as a popular benchmark for rotationally robust classification models. Recently, the result was slightly surpassed in [16].

Transforming an image to polar coordinates is another way how to enable rotational equivariance in CNNs between layers [2]–[4]. In polar coordinates, a rotation manifests as a circular shift of the image. To preserve rotational equivariance between layers, the papers suggest using an alternative kind of padding instead of the traditional zero-padding. However, [3], [4] do not get rid of the spatial dimension in the last convolutional layer before passing it to a fully-connected layer making it only robust to rotations, not invariant, as stated. A drawback of this polar transformation is the different sampling density in different parts of an image. Another drawback is the loss of translational equivariance, which is a natural benefit of classic CNNs. The Polar Transformer Network [2] tries to mitigate this problem by adding a trainable polar origin predictor before transforming the images into polar coordinates.

Summarizing, geometric invariance of CNNs is a hard and widely studied problem. For more details, we refer to the survey paper [17] and to the references thereof.

III. Methodology

A. Ensemble

Let us assume, we have two trained models – a classic CNN and a CNN that is modified to be invariant under a transformation. Let us denote them by CNN and I-CNN. Generally, we suppose that I-CNN is significantly worse than CNN on nondeformed images (this can be due to many reasons; for example, I-CNN often works with a lossy image representation), but also significantly better on deformed images. If any of these assumptions were not true, we could simply use one model which would be better in all cases. Both assumptions justify using an ensemble whose prediction, denoted by pred, is of the form

$$\text{pred}(I) = \beta(I) \cdot \text{pred}_{\text{CNN}}(I) + (1 - \beta(I)) \cdot \text{pred}_{\text{I-CNN}}(I) \quad (1)$$

where pred is a softmax output, i.e. a vector of n numbers, where n is the number of classes.

The parameter β is learnable and dependent on the image I and expresses how much I is deformed. When an image is not deformed at all and CNN is much better than I-CNN, then β should tend to 1. If it is the other way around, it should tend to 0. In the cases where I is only slightly deformed, so that the performance of CNN is similar to I-CNN, β is supposed to be somewhere between 0 and 1.

Learning β is therefore a question of the performances of CNN and I-CNN. We can examine how the performance of CNN depends on the intensity of the deformation, which can be, for example, an angle of rotation. Let us say that at intensity μ, the accuracies of both models coincide. Then let us choose l_1, l_2 so that β goes continuously from 1 to 0 (by a chosen function) in the interval $[\mu - l_1, \mu + l_2]$. That means that the ideal β is 1 for deformation intensities less than $\mu - l_1$ and 0 for those greater than $\mu + l_2$. Given a dataset, we can artificially deform each image with various intensities and assign corresponding β using the described procedure. Then we train a model with image as input and β as output. An important assumption that we put on this model

is that the training is easier and shorter than training CNN or I-CNN because one of major drawbacks of augmentation is its extreme extension of training time.

The proposed formula (1) is only a simple example of a combination at the score level that we choose. However, one could possibly find more sophisticated methods of doing so.

In the case of rotation, estimating β (which depends on the rotation angle) can work only for objects/images with a native orientation. It would not make sense for aerial/satellite/microscope images, for instance.

The scheme of the ensemble, where I-CNN is our rotation-invariant network using the Radon Transform as input, is shown in Figure 1.

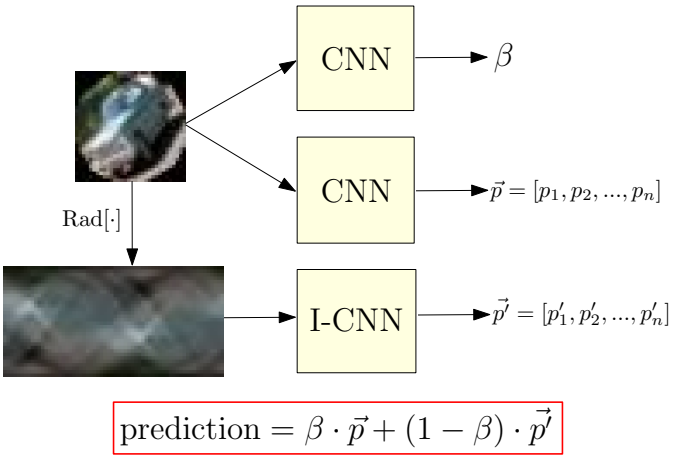

Fig. 1. The scheme of the proposed ensemble combining a classic and an invariant network. The combination is based on another model that estimates the intensity of the image deformation. It outputs $\beta \in [0, 1]$ and the final prediction is a convex combination of softmax outputs. In this case, the Radon transform is used as an input of I-CNN.

B. Radon transform

The Radon transform [18] captures directional information in an image, i.e. information along straight lines. A straight line with ρ as the perpendicular distance of the line from the origin and θ as the angle between the line and the y-axis can be described as

$$\rho = x \cos\theta + y \sin\theta.$$

Let f be a continuous function defined on a disc $\Omega \subset \mathbb{R}^2$ with radius r. Then, the Radon transform of f is defined as

$$\mathrm{Rad}[f](\rho, \theta) = \int_\Omega f(x, y)\delta(\rho - x\cos\theta - y\sin\theta)\,\mathrm{d}x\mathrm{d}y$$

where $\theta \in [0, 2\pi)$ and $\rho \in [-r, r]$. Let R_α denote the rotation of f by $\alpha \in [0, 2\pi)$. Then obviously

$$\mathrm{Rad}[R_\alpha f](\rho, \theta) = \mathrm{Rad}[f](\rho, \theta + \alpha)$$

Hence, a rotation of an image in Cartesian coordinates is merely a circular shift along the second axis of the Radon transform (see Figure 2 for an example). That is a key property that we require.

Traditionally, the coordinate θ is taken only from $[0, \pi)$ because $\mathrm{Rad}[f](\cdot, \theta)$ for $\theta \in [\pi, 2\pi)$ is only a mirror image of $\mathrm{Rad}[f](\cdot, \theta)$ for $\theta \in [0, \pi)$. However, since convolutional layers are not equivariant to mirroring, we need to include both intervals to get a circular shift when a rotation appears.

When working with digital images, discretization has to be done. For simplicity, let us assume I is an $S \times S$ image with C channels. First, the image has to be masked by the inscribed circle, as in Figure 2. Then, apparently, the angle θ must be sampled. We do so equidistantly using two settings $\alpha = \frac{2\pi}{S}$ and $\alpha = \frac{\pi}{S}$, so that the image after the transform has size $S \times 2S$ and $S \times S$, respectively. The choice of the sampling step is a question of preserving information about the image as much as possible, and at the same time not increasing the computational cost of the consequent network more than necessary. And one has to bear in mind that one half of the transformed image does not contain any information, since it is only the mirror image of the other half.

After the discretization, $\mathrm{Rad}(I)(\cdot, n\alpha)$ can be computed by rotating I by $n\alpha$ and summing each column individually.

C. Invariant CNN using Radon transform

The invariant CNN uses the Radon transform of an image as input. Then, any rotation manifests as a circular shift along the second axis of the input. It is possible to use an existing CNN like ResNet, but two modifications have to be done. The first one enforces equivariance of convolutional layers and the second makes the network invariant before classification.

- Although most CNNs use zero-padding for convolutions, this is not a suitable choice here because the convolutional feature mapping would not be equivariant to a circular shift. Let us assume that a convolutional layer uses a kernel of size $k \times k$ where k is odd. Then we pad the input along the second axis so that we copy the first and the last $\frac{k-1}{2}$ columns and append them as the new last and first columns, respectively. The rows might be zero-padded if needed. A similar idea appears in [2], [4]. With this modification, a circular shift of the input is transformed to a circular shift in the last convolutional layer.
- Let us assume that the CNN has a traditional structure of convolutional layers with max poolings (other frequently used techniques like batch normalization do not have an impact), possibly followed by fully-connected layers, and with a softmax output at the end. To make the final prediction invariant under rotation, it is necessary to eliminate the spatial dimension (or at least the angular dimension) in the last convolutional layer. We do so by applying Global Average Pooling [19].

However, while achieving rotation invariance, translation invariance (a natural benefit of classic CNNs) is lost, as the Radon transform depends on the origin around which an image is rotated. In the transform of a translated image, all columns remain the same except for a shift that is different for each column. Therefore, a translation invariance can be achieved for images without a background simply by shifting all columns so that their first nonzero intensities match. For images with

Fig. 2. Radon transform calculation. The images (examples taken from MNIST and CIFAR-10 datasets) are masked by the inscribed circle. Then, their Radon transform is computed. Image rotation results in a cyclic shift of the Radon transform along the horizontal axis.

a background, an origin predictor would have to be applied if one requires both invariance properties.

IV. Experiments

We perform experiments on three datasets – MNIST, rotated MNIST, and much more complex CIFAR10.

A. rotated MNIST

Rotated MNIST [15] is a standard dataset frequently used as a benchmark for networks adjusted to handle rotations better. Unlike regular MNIST, the training set is reduced to 10000 samples. The validation set and the test set have 2000 and 50000 images, respectively. Each image in each set is randomly (all angles equally probable) rotated by $\alpha \in [0, 2\pi)$. The random rotations and the reduced training set are the reasons why traditional CNNs perform worse on this dataset than the modified CNNs and even data augmentation is not able to change this.

We use the dataset to show the effectiveness of incorporating the Radon transform into traditional CNNs and also its clear superiority to the polar coordinates used in [2]–[4] for the same goal.

Our experiments are performed on a CNN consisting of 4 convolutional layers with 3×3 kernels, ReLU activation function, and $30, 30, 35$ and 40 channels, respectively. The invariant network uses the padding described in Section III-C. The first three layers are followed by a batch normalization and a 0.2 dropout, only the first two layers are followed by a 2×2 max pooling. The spatial dimension in the fourth layer is eliminated by Global Average Pooling and the last 0.2 dropout is applied before the softmax output. This architecture has 31,155 learnable parameters. For optimization, the Adam algorithm is used and the model with the highest accuracy on the validation set in 200 epochs is chosen.

The results are summarized in Table I. All models were trained 5 times and we show the mean results on the test set along with the standard deviation, maximum and minimum.

Although the results of Radon $S \times 2S$ are better than, for instance, the popular H-Net in [13] or others ([10], [20], [21]), it is not our purpose to possibly compete with state of the art models [14] and [16] by optimizing all the network settings and possible architectures.

model	mean	std	max	min
Radon $S \times 2S$	**98,54 %**	0,08 %	98,62 %	98,41 %
Radon $S \times S$	**98,35 %**	0,09 %	98,48 %	98,24 %
Polar $\frac{1}{2}S \times 2S$	**97,79 %**	0,08 %	97,88 %	97,67 %
Polar $\frac{1}{2}S \times 4S$	**97,11 %**	0,07 %	97,22 %	97,01 %
Cartesian $S \times S$	**94,60 %**	0,11 %	94,76 %	94,43 %

TABLE I
Rotated MNIST accuracies of CNNs based on Radon transform, polar transform and Cartesian coordinates. The accuracies are reported on the test set and the model with the highest accuracy on the validation set is taken.

B. MNIST

Classic MNIST has 60000 training samples and 10000 test samples and all of them have a standard orientation. We took 10000 images from the training test for validation.

The network architecture and the optimization procedure were the same as in rotated MNIST, but we ran only 100 epochs since the training set is much larger. The classic CNN trained on Cartesian coordinates achieved an accuracy of 99.39 % and the invariant CNN an accuracy of 99.21 % on the test set. These results are not surprising; MNIST has a large training set with rather simple visual content. However, we show in Figure 3 how rotating the test set influences the accuracy of the models. While the invariant CNN is almost not influenced at all by rotations (with a minimum of 98.93 % due to interpolation errors), the classic CNN deteriorates to almost a random classifier with 15.90 % for a rotation of 99 degrees. For simplicity, we show only the angles from $[0, 180]$ in Figure 3 since the other half is almost symmetrical.

We do not use the ensemble for MNIST, as the basic assumption is not met – the invariant CNN performs similarly to the classic one on nontransformed images. Therefore, there is no point in involving the ensemble and one can just use the invariant network when rotations might occur.

C. CIFAR-10

CIFAR-10 contains RGB images of size $32 \times 32 \times 3$, it has 50000 images for training and 10000 for testing. We took 10000 images from the training test for validation. The image

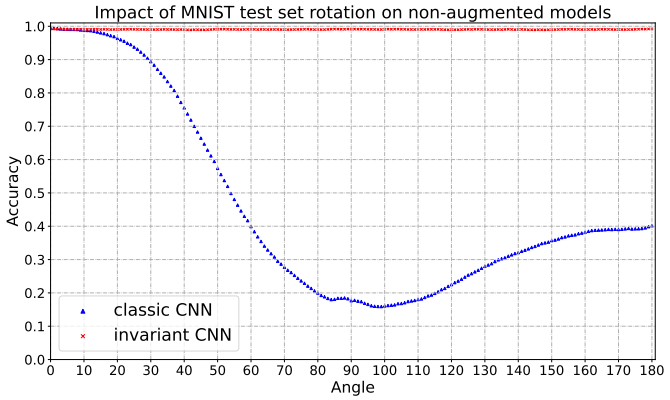

Fig. 3. Both networks with the same architecture (except for padding) were trained on standard MNIST without any augmentation. The graph shows their accuracy on the test set where all images were rotated by the same angle. The invariant network based on the Radon transform keeps almost the same accuracy along all angles.

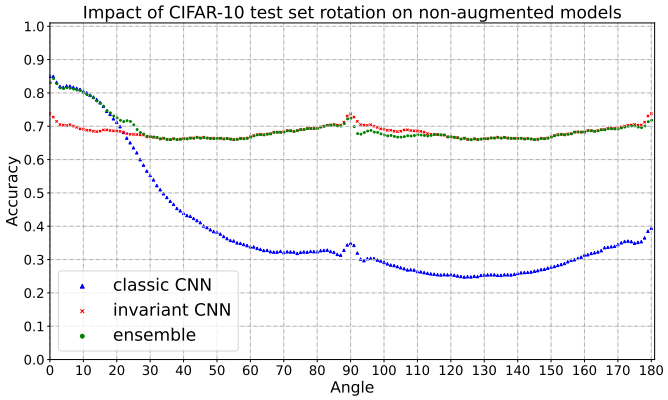

Fig. 4. A classic CNN and an invariant CNN with the same architecture (except for padding) were trained on CIFAR-10 without any augmentation. Moreover, an ensemble combining both models was constructed by procedure from Section III-A. The graph shows accuracy of all three models on the test set where all images were rotated by the same angle. The invariant CNN is not that stable anymore, probably due to bigger interpolation errors. The ensemble, however, almost copies the maximum of both models.

content is much more complex compared to MNIST. And since all objects have a background, it was necessary to mask all images by the inscribed circle for all training, validating and testing. Otherwise, the artificial rotations would be easily recognizable.

We used a slightly more powerful CNN consisting of 6 convolutional layers (32, 32, 64, 64, 128, 128 channels) with the same kernel size and activation function. Batch normalization follows after each layer. A 2×2 max pooling and dropout (0.2 and 0.3) is applied after the second and the fourth layer. The important part is Global Average Pooling after the last layer followed by a 0.4 dropout and the softmax output. This network has 289,194 trainable parameters. The optimization is the same as before with 100 epochs.

The dependence of the classic and invariant CNNs performance on the angle of rotation is depicted in Figure 4. Since, similarly as in Figure 3, the graph would be almost symmetric along $180°$, we focus only on the interval $[0, 180]$ for simplicity. The accuracy of the classic CNN goes from 85.00 % for no rotation down to 24.77 % for the 124 degrees rotation. The invariant CNN is not as stable as in the previous subsection and it decreases from 73.80 % down to a minimum of 65.98% for the 126 degrees rotation. However, the results are almost identical for 0, 90 and 180 degrees rotation. Therefore, we assume this is caused by interpolation errors that are stronger than those in the MNIST dataset. This explanation is also supported by a small irregular performance gain of the classic CNN around 90 degrees.

Nevertheless, the results suggest to apply the proposed ensemble because the classic CNN performs significantly better on nonrotated images (by circa 11 %), but also significantly worse on images rotated by higher angles.

First, we need to define the ground truth β for combining the models. By the procedure described in Section III-A, we need to find an angle μ where the accuracies of both models are approximately equal. This will be $\mu = 22°$ with accuracies of 68.16 % and 68.21 %. Then we choose $l_1 = l_2 = 6$ where the accuracies are still not that far away from each other. Hence, the ground truth β will be 1 for angles in $[0, 16]$ and 0 for $[28, 180]$. Let a_α be the accuracy of the classic CNN at angle α. Then for $\alpha \in [16, 28]$, we let β be

$$\frac{a_\alpha - a_{28}}{a_{16} - a_{28}}$$

to copy the decrease of the CNN performance on this interval.

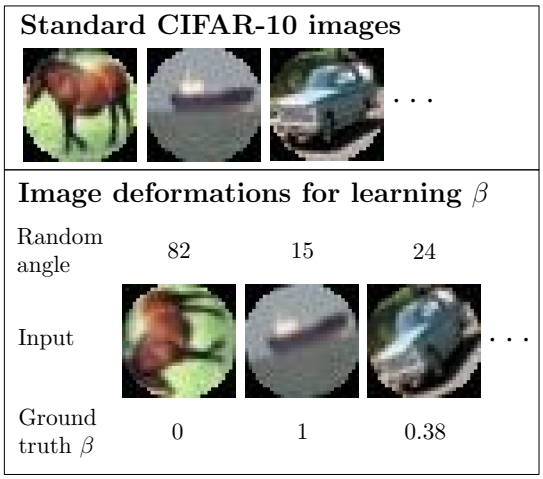

Fig. 5. For training the CNN that outputs β, all images are randomly rotated and the corresponding label is assigned by the described procedure.

Second, we have to train a model that predicts the corresponding β for a rotated CIFAR-10 image. We used the same CNN as for the classification, but with a single sigmoid node in the output. The optimization was set to 40 epochs, i.e. less than for the classification. The training set was the same, but with each image randomly rotated (see Figure 5) by an integer angle so that the intervals $[0, 3], [4, 28], [88, 112]$ and $[178, 180]$ are chosen with probabilities 0.2, 0.3, 0.33 and 0.1, respectively. This distribution was chosen heuristically because

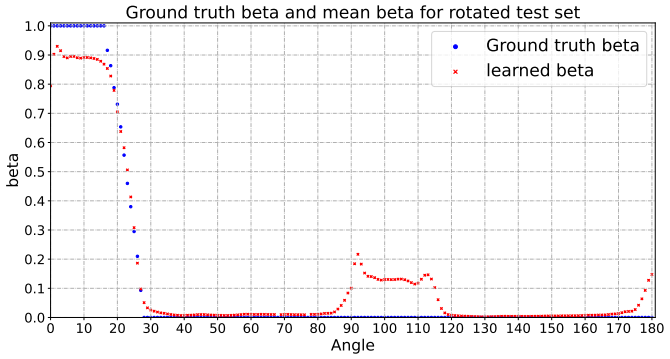

Fig. 6. A network computing $\beta(I)$ for the ensemble was trained on CIFAR-10 images. The graph shows the mean of β on the test (red crosses) where all images were rotated by a certain angle. The blue dots refer to ground truth β as we defined it for each angle.

with a uniform distribution, the network seemingly tends to learn interpolation errors too much (they are strong in 32×32 images). For instance, the angle 0 is then treated as angles 90 or 180 and the learned β is too small. Therefore, we put a higher weight on intervals with distinct β and the same intensity of interpolation errors to force the network to focus more on the object orientation.

In Figure 6, we show the mean values of β on the rotated test set for all integer angles. The learned β in the intervals $[0, 16]$ and $[90, 116]$ is further from the ground truth β than in the rest of $[0, 180]$. We assume that this is a realistic scenario (since these two intervals have different β and similar interpolation errors) and the rest would probably behave similarly on datasets of higher resolution. Nevertheless, even when taking this into account, the ensemble is still able to achieve almost the maximum of the two combined models, as can be seen in Figure 4. It is very close to the performance of the classic CNN on nonrotated images, but it is much more robust to rotations, as it mostly relies on the invariant network when rotations by higher angles are met. Let us also remark that in the interval $[16, 28]$ where the ground truth β decreases from 1 to 0, the ensemble is even a bit better than the individual CNNs (the maximum gain is 3.88% for 23°).

V. Conclusion

We proposed an ensemble of two networks to achieve robustness to rotation. Both of them are trained without any data augmentation, which is the most important benefit of the method. One of the networks is a rotation-invariant CNN that we proposed and it uses the Radon transform of an image as input. The main idea of the ensemble is the design of the auxiliary network, which estimates the rotation angle of an input image and yields the parameter for the network fusion. We demonstrated that the ensemble works significantly better than each individual network.

The presented idea is general and can be used for other deformations than just a rotation if a proper invariant model is used in the combination.

Acknowledgment

This work has been supported by the Czech Science Foundation under the grant no. GA21-03921S and by the Praemium Academiae.

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, vol. 25, Curran Associates, Inc., 2012.

[2] C. Esteves, C. Allen-Blanchette, X. Zhou, and K. Daniilidis, "Polar transformer networks," in *International Conference on Learning Representations*, 2018.

[3] R. Jiang and S. Mei, "Polar coordinate convolutional neural network: From rotation-invariance to translation-invariance," in *2019 IEEE International Conference on Image Processing (ICIP)*, pp. 355–359, 2019.

[4] J. Kim, W. Jung, H. Kim, and J. Lee, "CyCNN: A rotation invariant CNN using polar mapping and cylindrical convolution layers," 2020.

[5] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–9, 2015.

[6] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[7] I. Vasiljevic, A. Chakrabarti, and G. Shakhnarovich, "Examining the impact of blur on recognition by convolutional networks," 2017.

[8] M. Lébl, F. Šroubek, and J. Flusser, "Impact of image blur on classification and augmentation of deep convolutional networks," in *Image Analysis*, (Cham), pp. 108–117, Springer Nature Switzerland, 2023.

[9] M. Jaderberg, K. Simonyan, A. Zisserman, and k. kavukcuoglu, "Spatial transformer networks," in *Advances in Neural Information Processing Systems*, vol. 28, Curran Associates, Inc., 2015.

[10] T. Cohen and M. Welling, "Group equivariant convolutional networks," in *Proceedings of The 33rd International Conference on Machine Learning*, (New York, New York, USA), pp. 2990–2999, PMLR, 6 2016.

[11] W. Freeman and E. Adelson, "The design and use of steerable filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 9, pp. 891–906, 1991.

[12] T. S. Cohen and M. Welling, "Steerable CNNs," in *International Conference on Learning Representations*, 2017.

[13] D. E. Worrall, S. J. Garbin, D. Turmukhambetov, and G. J. Brostow, "Harmonic networks: Deep translation and rotation equivariance," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (Los Alamitos, CA, USA), pp. 7168–7177, IEEE Computer Society, 7 2017.

[14] M. Weiler and G. Cesa, "General E(2)-equivariant steerable CNNs," in *Advances in Neural Information Processing Systems*, vol. 32, Curran Associates, Inc., 2019.

[15] H. Larochelle, D. Erhan, A. Courville, J. Bergstra, and Y. Bengio, "An empirical evaluation of deep architectures on problems with many factors of variation," vol. 227, pp. 473–480, 06 2007.

[16] D. M. Knigge, D. W. Romero, and E. J. Bekkers, "Exploiting redundancy: Separable group convolutional networks on lie groups," in *Proceedings of the 39th International Conference on Machine Learning*, pp. 11359–11386, PMLR, 7 2022.

[17] A. Mumuni and F. Mumuni, "Cnn architectures for geometric transformation-invariant feature representation in computer vision: A review," *SN Comput. Sci.*, vol. 2, 6 2021.

[18] J. Radon, "On the determination of functions from their integral values along certain manifolds," *IEEE Transactions on Medical Imaging*, vol. 5, no. 4, pp. 170–176, 1986.

[19] M. Lin, Q. Chen, and S. Yan, "Network in network," 2014.

[20] Z. Shen, L. He, Z. Lin, and J. Ma, "PDO-eConvs: Partial differential operator based equivariant convolutions," in *International Conference on Machine Learning*, pp. 8697–8706, PMLR, 2020.

[21] J. Singh and C. Singh, "Learning invariant representations for equivariant neural networks using orthogonal moments," 2022.