International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:11, No:3, 2017

Pose Normalization Network for Object Classification

Bingquan Shen

***Abstract*—**Convolutional Neural Networks (CNN) have demonstrated their effectiveness in synthesizing 3D views of object instances at various viewpoints. Given the problem where one have limited viewpoints of a particular object for classification, we present a pose normalization architecture to transform the object to existing viewpoints in the training dataset before classification to yield better classification performance. We have demonstrated that this Pose Normalization Network (PNN) can capture the style of the target object and is able to re-render it to a desired viewpoint. Moreover, we have shown that the PNN improves the classification result for the 3D chairs dataset and ShapeNet airplanes dataset when given only images at limited viewpoint, as compared to a CNN baseline.

***Keywords*—**Convolutional neural networks, object classification, pose normalization, viewpoint invariant.

I. Introduction

CONVOLUTIONAL Neural Networks (CNN) have been shown to be effective on a variety of computer vision tasks, such as classification [1], [2], detection [3], [4], semantic segmentation [5], and image captioning [6]. Its success could be attributed to the increase in computing power and the availability of big data sets such as the PASCAL VOC [7] and ImageNet [8].

For object recognition, the typical approach is to train the network with multiple images of the object undergoing a combination of variations such as lighting, pose and background [9], and the network is expected to implicitly learn the variations from the data. However, when one have limited viewpoints of a particular object for classification, this method would not be feasible.

Recent works have shown that CNN is capable of generating 2D projections of 3D objects [10] given the desired model parameters, such as viewpoint and color. In [11], they found that one can disentangle the network's latent variables to represent object style and variations, such as out-of-plane rotation.

In this paper, we aim to use these prior knowledge of 3D object rotation to aid in classification task. Given the problem of limited viewpoints of a particular object for classification, we propose the Pose Normalization Network (PNN) to transform the object to an existing viewpoint in the training dataset for before classification.

Bingquan Shen is with the DSO National Laboratories, Singapore (E-mail: sbingqua@dso.org.sg).

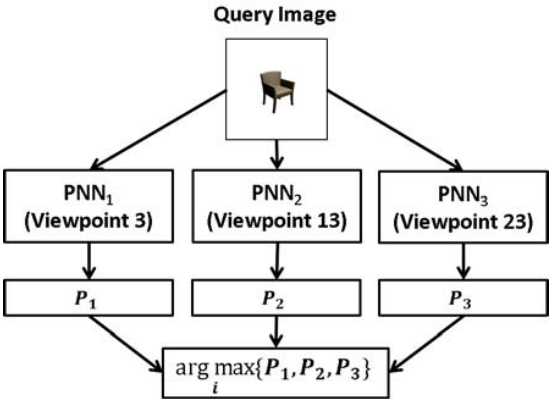

Fig. 1 Classification using PNN

The paper is organised as follows. Firstly, we begin with a review of the related works of using CNN for modeling of out-of-plane rotation and image pre-processing networks. Next, we introduce the PNN architecture, datasets used and the methodology used for training and classification. Then, we compare the classification results of the PNN to a baseline CNN trained with the same training dataset. Finally, we conclude that PNN yield better classification results than the baseline CNN.

II. Related Work

As mentioned, there are a number of works which utilizes CNN for modeling out-of-plane rotations.

In [10], a CNN was trained to generate 2D projections of 3D objects given specific parameters in a supervised setting. Their approach requires the input of the desired object class, and hence cannot generalizes to unseen classes. On the other hand, our method uses an encoder to encode the style of the object, so it is able to generalise to novel objects.

A. Network Architecture

Recently, [12], [13] have shown that variational autoencoder (VAE) can disentangle variations between style and label of MNIST images. Based on the VAE, [11] developed the Inverse Graphics Network (IGN), which is able to disentangle factors of variation, including out-of-plane rotations, from the style of the object within the image in its learnt image representation. By varying the image representations, the IGN is able to re-render an

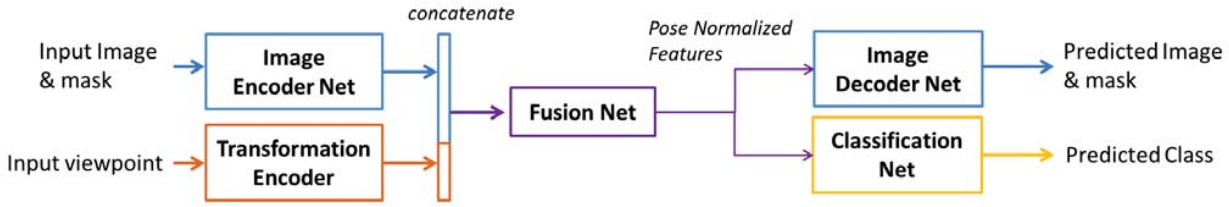

Fig. 2 Overview of Pose Normalization Network (PNN) Architecture

object undergoing various transformations. Nonetheless, the IGN is unable to output a desired viewpoint of the image as variation is encoded in latent space. On the contrary, our model takes in a desired viewpoint signal that allows re-rendering the input object in that viewpoint.

Our work is related to [14] who have developed a recurrent CNN encoder-decoder network which can incrementally applies out-of-plane rotations to the object within the image with control signal provided by the user. However, their method accumulates error at each increment, and reconstruction error builds up as rotation steps increases. Moreover, their work focus on the image reconstruction of the transformed object which differs from ours.

Similarities are seen in [15], [16] whereby input images are transformed to a canonical form to simplify inference in the subsequent layers. In [16], they generalised these network to enable end-to-end training. Still, their methods only considered affine transformations of a 2D plane. On the other hand, our work deals with the normalization of out-of-plane rotation of a 3D object for classification purpose.

III. Method

The overall architecture of the Pose Normalization Network (PNN) is shown in Fig. 2. It consists of several sub-networks, namely image encoder net, transformation encoder, fusion net, image decoder net and classification net.

The image encoder network (Fig. 3) takes the RGB and mask channel of a 64×64 image data as input. It consists of 4 convolutional and max pooling layers followed by a fully connected layer. The first 2 convolutional layers uses 32 and 64 filters of 5×5 respectively, while the next 2 convolutional layers both uses 128 filters of 3×3. The output of the final convolutional layer is flattened and fed through a fully connected layer with 1024 neurons.

The transformation encoder network consists of 3 fully connected layers which takes in the desired viewpoint as input. Following [10], the azimuth transformations θ and elevation transformations ϕ are encoded as $\{cos\theta, sin\theta\}$ and $\{cos\phi, sin\phi\}$ respectively, to ensure continuity. All fully connected layers in this sub-network consists of 64 neurons for each transformation.

The fusion network takes the concatenated outputs of the image encoder network and transformation encoder network as input. It consists of 3 fully connected layers with 1024, 1024 and 2048 neurons respectively. The purpose of the fusion network is to combine the style information and the new viewpoint information, which are provided by the image encoder network and transformation encoder network respectively.

The image decoder net (Fig. 4) reshapes the 2048 output vector of the fusion network into a $128 \times 4 \times 4$ tensor and uses it as input. It is made up of 4 deconvolutional layers. The first 2 deconvolutional layers uses 128 and 64 filters of 2×2 with a stride of 2 respectively, while the next 2 dedconvolutional layers both uses 32 and 4 filters of 4×4 with a stride of 2 respectively. The output is a $4 \times 64 \times 64$ tensor which is used to compared with the target chairs RGB and mask data.

The classification net takes the 2048 output vector of the fusion network as input. It consists of 2 fully connected layers with 1024 and 809 neurons respectively. The number of output neurons correspond to the number of classes of chairs in the dataset.

For all layers, except the classification net output layer, used a leaky rectified linear (ReLU) activation function [17] with a negative slope of 0.2.

A. Dataset

1) Chairs Dataset: The 3D chairs dataset consist of 1393 chairs CAD models made public by Aubry et al. [18], as shown in Fig. 5. A reduced dataset of 809 chairs are selected for our experiments. These are selected by Dosovitskiy et al. [10] to remove near-duplicate models. The preprocessing done are similar to Yang et al. [14]. It consists of 31 viewpoints of each chair rotated from 0^o to 360^o about the azimuth. The elevation is fixed at 20^o. Firstly, the rendered images are cropped to the chair. Next, a small white border is added to the image before they are all resized to 64×64 pixels. A binary mask of each image is obtained by subtracting the white background. The mask and RGB layers are concatenated to form a $64 \times 64 \times 4$ data tensor. The first 500 models are used as training set and the remaining 309 models are used as the test set with limited viewpoint data.

2) ShapeNet Dataset: Airplane 3D models are taken from ShapeNet [19], as shown in Fig. 6. The 1359 airplanes are selected out of the 4045 models, where low-resolution and non-realistic models are removed. Following the chairs dataset, each airplane consists of 31 viewpoints rotated from 0^o to 360^o about the azimuth. In addition, the elevation is varied from 20^o to 50^o with increments of 5^o for 7 elevation variation per azimuth viewpoint, giving rise to 217 images per model. Rendering are done in blender [20]. The preprocessing are similar to the chairs dataset. The first 700

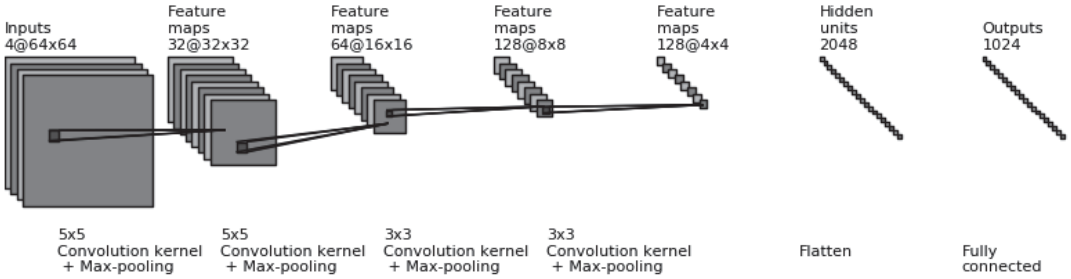

Fig. 3 Image encoder net

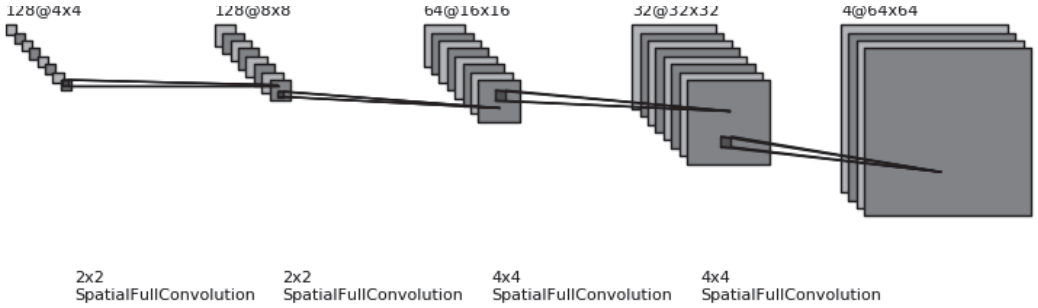

Fig. 4 Image decoder net

Fig. 5 Example images from chair dataset

Fig. 6 Example images from ShapeNet dataset

models are used as training set and the remaining 659 models are used as the test set with limited viewpoint data.

B. Training Details

All networks are trained using Torch [21] and the training process is split into two parts.

For the first part, the concept of out-of-plane rotation is introduced to the network. This is achieved by training the network to predict the target image, given an input image of arbitrary viewpoint and the desired viewpoint. The network was trained using mean square error (MSE) loss function:

$$loss(p,t) = \frac{1}{N}\sum_{i=1}^{N}|p_i - t_i|^2, \quad (1)$$

where p, t and N are the reconstructed image, target image and total number of elements respectively. The reconstructed image is output of the network based on an input image and target viewpoint, while the target image is the actual rendering of the selected class at the target viewpoint.

The network was trained using ADAM [22] with a learning rate of 0.0001, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 1e{-}8$. At each iteration, the chair class and target viewpoint are selected at random to form a batch of 32, before it is fed through the network. The network is trained until the loss stop decreasing significantly (approximately 10 million iterations).

All the viewpoints from the 500 models in the train set are used for this first part of training.

During the second part, the network is made to exploit its learnt rotation for classification. Therefore, only the weights of the classification net are adjusted, while the remaining weights of the network are fixed. Similar optimization parameters were used to trained the classification net, except the learning rate, which was reduced to $1e-5$. At each iteration, a batch of 32 chair classes and target viewpoints are selected at random to train the network. A loss function of the negative log likelihood over all classes was used.

To account for the additional test classes for the classification task, 3 out the 31 azimuth viewpoints from each model in the test set are selected (azimuth viewpoints 3,13 and 23) and merge with the train set for training classification.

C. Classification

The computation of the classification output of the PNN is as shown in Fig. 1. Firstly, each image from the test dataset

TABLE I
CLASSIFICATION RESULTS

Method	Accuracy (%) (Chairs Dataset)	Accuracy (%) (ShapeNet Dataset)
CNN baseline	69.11	45.43
Pose Normalization Network	95.98	85.40

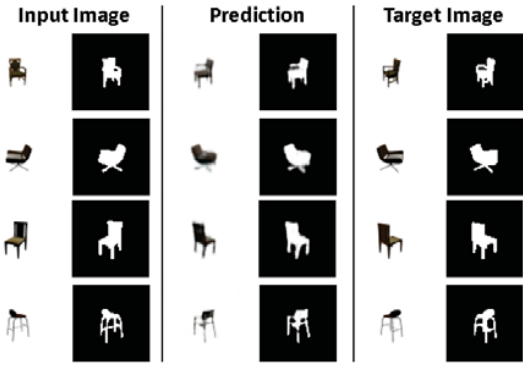

Fig. 7 Reconstructed results of chair dataset

was normalized to the 3 selected viewpoint used during training. This is achieved by passing the input image and desired viewpoint to the PNN. Next, the pose normalized features are passed through the classification network. The PNN output for each selected viewpoint is denoted as,

$$P_n = [p_{n,1}, p_{n,2}, ..., p_{n_i}], \quad (2)$$

where P_n is the output from the PNN at the n-th selected viewpoint and $p_{n,i}$ is the softmax output of the i-th class at the n-th PNN. The maximum activated prediction from the set of results is selected as the classification output.

IV. Results and Discussion

A. Reconstruction Results

For the chairs dataset, the average reconstruction error of 0.0104 is achieved. The qualitative result of the reconstruction is shown in Fig. 7. It is observed that the network can capture the style of the target chair and is able to re-render it to the desired viewpoint. Next, we measure the reconstruction error with respect to the difference in viewpoint between the input and target. Chairs in the test set are uniformly sample and the desired viewpoint is varied with respect to the selected input. The result is as shown in Fig. 8. As expected, the lower reconstruction error corresponds to smaller difference in viewpoint. However, it is interesting to note that reconstruction error is highest when viewpoint differs from between 6 to 10, which approximately correspond to 66 to 116 degrees. Reconstruction error decrease from viewpoint difference of 10 onwards. This could be due to the symmetry of the chair during rotation.

For the shapenet airplane dataset, similar reconstruction results (Fig. 9) is observed even when an additional elevation variation is added. The network is able to capture the style of the target airplane and re-render it to the azimuth and elevation of choice.

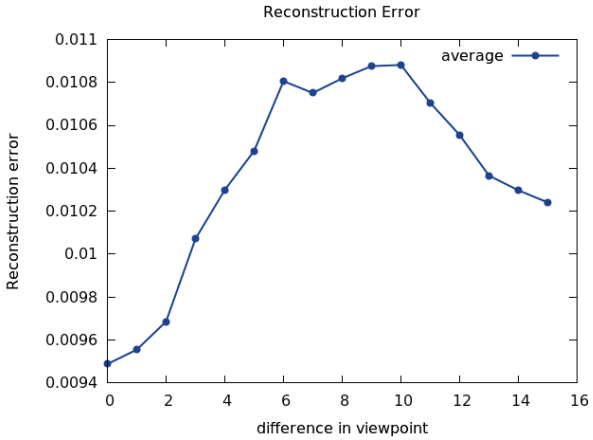

Fig. 8 Reconstruction loss to difference in viewpoint

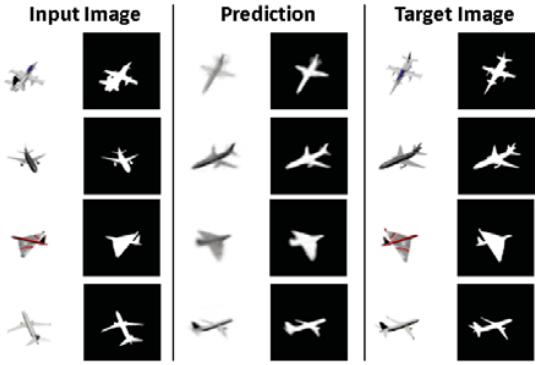

Fig. 9 Reconstructed results of shapenet dataset

B. Classification Results

To test the effectiveness of the PNN, we compare the classification result of the PNN to a baseline convolution neural network (CNN) which was train using similar training dataset. For a fair comparison, the architecture of the baseline CNN is similar to the image encoder net, fusion net and classification layers of the PNN. In addition, the number of training iterations was comparable to the PNN.

The classification results, as shown in Table I, show that the PNN performs significantly better as compared to the CNN baseline. A breakdown of classification accuracy over viewpoint for the chairs dataset and shapenet dataset are shown in Fig. 10 and Fig. 11 respectively. Viewpoints 3, 13 and 23 are in the classification training dataset, hence the corresponding bars depicts the training accuracy results. In general, the classification accuracy decreases as the viewpoint deviates from the viewpoint provided during training. However, the performance of the PNN is more consistent when compare to the CNN baseline. The classification accuracy of the CNN dropped more than the PNN as the viewpoint deviation increase.

For the chairs dataset, it is noticed that there is a major drop in performance for both networks between viewpoints 6

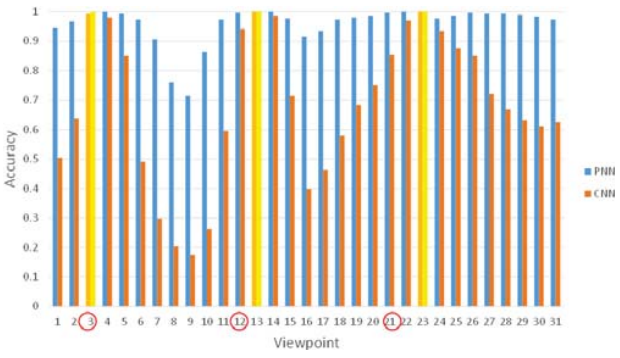

Fig. 10 Accuracy over viewpoints of chairs dataset

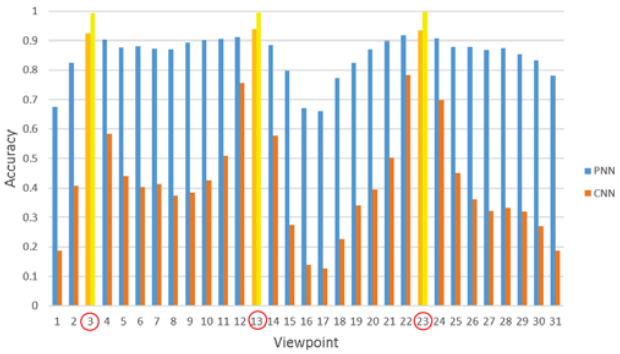

Fig. 11 Accuracy over viewpoints of ShapeNet dataset

to 10. Studying the dataset, it is found that these viewpoints correspond to the back views of the chairs and they appear mostly similar. Hence, these viewpoints are more challenging to classify. For the more challenging shapenet dataset, accuracy of the CNN is consistently lower across all viewpoint deviations.

The results clearly depicts the robustness of the PNN in out-of-plane rotated object classification as compared to the CNN baseline. Similar training dataset are used, however the training methodology enables the PNN to capture the concept of out-of-plane rotation. Hence, the PNN is able to draw relation of similar objects that are rotated significantly from before. This shows that prior specialized training of the network to model variation (out-of-plane rotation) is beneficial to subsequent classification task, both in terms of accuracy and the number of training data required.

However, this study does have some limitations. Firstly, the dataset is rendered from 3D CAD models and hence clean, as there are no background noise and the object is always well-centered without occlusions. Secondly, the transformation learned is specific to the object class, either chairs or airplanes.

Therefore, future works of this study includes extending the PNN to more realistic datasets with the presence of background and the absence of mask. Recent techniques on object detection [3], [4] and object segmentation from background [23] could supplement the PNN on these realistic datasets. In additional, knowledge transfer of the PNN across different object class would be explored.

V. Conclusion

This paper presented a pose normalization network (PNN) for effective object classification under the condition of limit data. By imparting prior knowledge of pose and rotation concepts within the network, we have shown that the PNN is able to improve classification results for objects that are rotated out-of-plane significantly with only limit viewpoints information provided. More importantly, it shows that a network that specializes in modeling a known variation (out-of-plane rotation in this case) can significantly reduce the data required for classification task.

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, pp. 1097–1105, 2012.

[2] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[3] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 580–587, 2014.

[4] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, pp. 91–99, 2015.

[5] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3431–3440, 2015.

[6] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and tell: A neural image caption generator," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3156–3164, 2015.

[7] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International journal of computer vision*, vol. 88, no. 2, pp. 303–338, 2010.

[8] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, *et al.*, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[9] Y. LeCun, F. J. Huang, and L. Bottou, "Learning methods for generic object recognition with invariance to pose and lighting," in *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, vol. 2, pp. II–97, IEEE, 2004.

[10] A. Dosovitskiy, J. Tobias Springenberg, and T. Brox, "Learning to generate chairs with convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1538–1546, 2015.

[11] T. D. Kulkarni, W. F. Whitney, P. Kohli, and J. Tenenbaum, "Deep convolutional inverse graphics network," in *Advances in Neural Information Processing Systems*, pp. 2530–2538, 2015.

[12] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *CoRR*, vol. abs/1312.6114, 2013.

[13] D. P. Kingma, S. Mohamed, D. J. Rezende, and M. Welling, "Semi-supervised learning with deep generative models," in *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pp. 3581–3589, 2014.

[14] J. Yang, S. E. Reed, M.-H. Yang, and H. Lee, "Weakly-supervised disentangling with recurrent transformations for 3d view synthesis," in *Advances in Neural Information Processing Systems*, pp. 1099–1107, 2015.

[15] G. E. Hinton, A. Krizhevsky, and S. D. Wang, "Transforming auto-encoders," in *Artificial Neural Networks and Machine Learning–ICANN 2011*, pp. 44–51, Springer, 2011.

[16] M. Jaderberg, K. Simonyan, A. Zisserman, *et al.*, "Spatial transformer networks," in *Advances in Neural Information Processing Systems*, pp. 2008–2016, 2015.
[17] B. Xu, N. Wang, T. Chen, and M. Li, "Empirical evaluation of rectified activations in convolutional network," *arXiv preprint arXiv:1505.00853*, 2015.
[18] M. Aubry, D. Maturana, A. Efros, B. Russell, and J. Sivic, "Seeing 3d chairs: exemplar part-based 2d-3d alignment using a large dataset of cad models," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3762–3769, 2014.
[19] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, *et al.*, "Shapenet: An information-rich 3d model repository," *arXiv preprint arXiv:1512.03012*, 2015.
[20] Blender Online Community, *Blender - a 3D modelling and rendering package*. Blender Foundation, Blender Institute, Amsterdam, 2016.
[21] R. Collobert, K. Kavukcuoglu, and C. Farabet, "Torch7: A matlab-like environment for machine learning," in *BigLearn, NIPS Workshop*, no. EPFL-CONF-192376, 2011.
[22] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
[23] P. O. Pinheiro, R. Collobert, and P. Dollar, "Learning to segment object candidates," in *Advances in Neural Information Processing Systems*, pp. 1981–1989, 2015.