HOW MUCH POSITION INFORMATION DO CONVOLUTIONAL NEURAL NETWORKS ENCODE? REPRODUCIBILITY CHALLENGE REPORT

Daji LI, Nian LIU, Tian ZHOU
School of Electronics and Computer Science
University of Southampton
{dl5n19, nl2y19, tz3n19}@soton.ac.uk

ABSTRACT

In this review report, we introduce a different training dataset to reproduce the experiment. In the paper we chose, the authors built up with a simple Position Encoding Network (PosENet) to verify the hypothesis of how much and where the position information while offering clues about the positional information which derived from deep Convolutional Neural Networks (CNNs). Since the authors did not provide the source code, we used Pytorch successfully reproduced the vast majority of the experiment all by ourselves. And even some details did not be provided in the paper, we still restored the experiment through reasonable inference. And the reproduced result in our experiment matches in general with the one from the original paper.

1 INTRODUCTION

The paper we chose(Islam et al., 2020) demonstrate the way of transfer the position information from CNN and how to encode them. As is known, classic CNN model is spatially-agnostic and therefore the CNN model would not be used to encode the absolute spatial information. As detecting saliency on a cropped image by CNN, the most salient region used to be not changed (Jia & Bruce, 2020). In the chose paper, the authors used randomization tests (Edgington, 1969) to examine the role of absolute position information in CNN model and they predict that CNN using encoded position information as one basis to make decisions. And at the same time, their experiments even reveal that position information is learned from zero-padding operation. From this paper, it helps us to better understand the position information from learned features in CNNs which provide an significant observation for the future investigation. In our report, we will show the details of methodology in Section 2, reproduce some parts of experiment and analyse the results in Section 3 and make a conclusion in Section 4 at last. We had uploaded all codes[1] on `github`.

2 METHODOLOGY

One hypothesis we chose in the paper is that position information is implicitly encoded within the extracted feature maps and plays an important role in classifying, detecting or segmenting objects from a visual scene. In order to prove this hypothesis, they build an end-to-end manner (PosENet) to predict position information in different CNN models and VGG16 and ResNet152 have been chose in this paper. The authors built a network called Position Encoding Network (PosENet) which is used to predict a gradient-like position information mask $\hat{f}_p \in \mathbb{R}^{h \times w}$ by given image $\mathcal{I}_m \in \mathbb{R}^{h \times w \times 3}$ and define the absolute coordinates for each pixel from left to right and top to bottom. And they generated five gradient-like mask $\mathcal{G}_{pos} \in \mathbb{R}^{h \times w}$ as supervision, and the weight of this network is fixed.

2.1 POSITION ENCODING NETWORK

The Position encoding network (PosENet) shown in Fig. 1, has two main parts: a feedforward convolutional encoder network f_{enc} and a simple position encoding module, denoted as f_{pem}.

[1] https://github.com/COMP6248-Reproducability-Challenge/HOW-MUCH-POSITION-INFORMATION-DO-CONVOLUTIONAL-NEURAL-NETWORKS-ENCODE-/blob/master/codes/code.ipynb

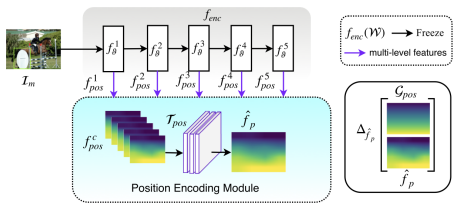

Figure 1: Illustration of PosENet architecture (Islam et al., 2020)

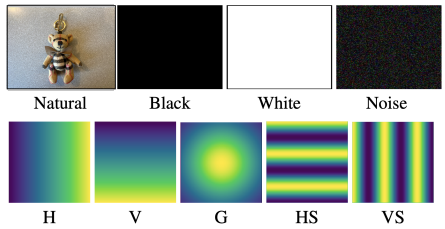

Figure 2: Sample images and five gradient-like ground-truth position maps.

	f_ϑ^1	f_ϑ^2	f_ϑ^3	f_ϑ^4	f_ϑ^5
VGG16	Conv1_2	Conv2_2	Conv3_3	Conv4_3	Conv5_3
Resnet152	Conv1_x	Conv2_x	Conv3_x	Conv4_x	Conv5_x

Table 1: The selection of convolutional layer of VGG16 and ResNet152.

The f_{enc} responses to extract the feature maps from typical CNN models of `torchvision` library with pre-trained parameters, and then input those feature maps to f_{pem}. As in paper authors did not explicitly extract convolutional layers of baselines model (VGG16 and Resnet152). So we choose the convolutional layers in average shown in Table. 1, and extract after following pooling layers of those convolutional layers. The products of f_{enc} are extracting position information which are multi-scale feature maps. We define them as $f_{pos}^1, f_{pos}^2, f_{pos}^3, f_{pos}^4, f_{pos}^5$ extracted by f_ϑ^i as follows:

$$f_{pos}^i = f_\vartheta^i \left(\mathbf{W_a} * \mathcal{I}_m\right) \tag{1}$$

where $\mathbf{W_a}$ and $*$ denotes weights from pre-trained which are frozen and the convolution operation.

The f_{pem} responses to train the extracting position information. As the extracting feature maps have different spatial size, we use tri-linear interpolation transformation function $\mathcal{T}_{pos}$ to transform them at same size $[3, 28, 28]$ in order to reducing the amount of data[2], concatenate all the feature maps from bottom to top, and then transform to a feature map f_{pos}^c. The f_{pos}^c will be trained in the f_{pem} according to equation 2 which just use a convolutional layer to generate $\hat{f}_p$.

$$f_{pos}^c = \left(f_{pos}^1 \oplus \cdots \oplus f_{pos}^5\right) \quad \hat{f}_p = \left(\mathbf{W}_{\text{pos}}^{\text{c}} * f_{pos}^c\right) \tag{2}$$

where the $\oplus$ denotes concatenation operation and the $\mathbf{W}_{\text{pos}}^{\text{c}}$ is the trainable weights.

The main function of the encoding module is to validate whether position information is implicitly learned when trained in typical CNN models of image classification. When the output of model is random, it means there is no position information encoded in the features maps position information encoded in the features maps and vice versa.

2.2 DATASET AND EVALUATION METRICS

Datasets: We also use the DUT-S dataset (Wang et al., 2017) but we use the testing set which contains 5,019 images for training, as reduce the time complexity. We follow the training protocol in the paper we chose and test with a natural image of UoS Bear which is the image sample we provided. And the synthetic images (white, black and Gaussian noise) are also used as described in Fig. 2. In fact, you can choose any image dataset for training, as we only explore how the position encode in CNNs.

Synthetic data and ground-truth generation: According to the paper we chose, we generated five gradient like ground-truth mask (Horizontal (**H**), Vertical (**V**), Gaussian distribution (**G**), horizontal and vertical stripes, (**HS**, **VS**)) as target of PosENet shown in Fig. 2.

Evaluation Metrics: We use the same evaluation method with the paper we chose. We choose Spearmen Correlation (SPC) and Mean Absoute Error (MAE) to measure the position encoding performance.The SPC is defined as the Spearman's correlation between the ground-truth and the predicted position map.

[2]If using whole size multi-scale feature maps, it will lead a huge GPUs consumption. This is simplified.

For ease of interpretation, we keep the SPC score within range [-1, 1]. MAE is the average pixel-wise difference between the predicted position map and the ground-truth gradient position map.

2.3 IMPLEMENTATION DETAILS

We built an extractor with the pre-trained VGG16 and ResNet152 for the ImageNet classification task which collected the outputs(f_{pos}^{c}) as input for training. In f_{pem}, we train the model as mentioned before and the parameter are initialed with uniform *xavier initialization* (Glorot & Bengio, 2010) For training optimizer, we use Adam optimizer for 15 epochs with learning rate of 1e-3 and weight decay of 1e-4. Although authors use stochastic gradient descent, we find Adam optimizer shows a stable performance. We resize each image to a fixed size of 224×224 during training and inference. Since the spatial extent of multi-level features are different, we align all the feature maps to a size of 28×28. One more difference between the paper we chose and our reproduction, the authors defined a pixel-wise mean squared error loss to measure the difference between predicted and ground-truth position maps and we use `nn.MSELoss()` of `torch` library which shows a similar performance.

3 REPRODUCTION

In this section, we shows some parts of experiments which illustrate position information transfers in CNNs model and some influences for this process. We define that VGG indicates PosENet is based on the features extracted from the VGG16 model. Similarly, ResNet represents the combination of ResNet-152 and PosENet. PosENet alone denotes only the PosENet model is applied to learn position information directly from the input image.

	Model	UoS Bear		Black		White		Noise	
		SPC	MAE	SPC	MAE	SPC	MAE	SPC	MAE
H	PosENet	-0.138	0.279	NaN	0.251	NaN	0.323	-0.030	0.274
	VGG	0.394	0.235	0.217	0.241	0.274	0.242	0.214	0.245
	ResNet	0.703	0.186	0.438	0.247	0.109	0.252	0.470	0.234
G	PosENet	-0.242	0.234	NaN	0.196	NaN	0.257	-0.034	0.218
	VGG	0.775	0.141	0.398	0.195	0.487	0.193	0.170	0.188
	ResNet	0.886	0.135	0.505	0.206	0.346	0.196	0.259	0.189
VS	PosENet	-0.017	0.319	NaN	0.308	NaN	0.330	-0.045	0.315
	VGG	0.281	0.292	0.231	0.300	0.224	0.299	0.272	0.290
	ResNet	0.402	0.305	0.515	0.302	0.038	0.305	0.172	0.304

Table 2: Some comparison of different networks in terms of SPC and MAE across different image.

3.1 POSITION INFORMATION IN PRETRAINED MODELS

First, we validate whether there have position information encoded in the typical CNNs for image classification tasks. In this part, we use 3×3 size kernel without any padding in f_{pem} following above training protocol. After training, we input UoS Bear, white, black and noise image to test the model respectively, and the result shows in Table. 2 and some samples' result shows in Fig. 3.

In our reproduction, the result shows a similar performance with the paper we selected but not as well as the results of the paper we chose. The output of f_{pem} is not totally random in VGG and ResNet but shows a similar distribution with ground-truth mask, and the output of PosENet show an irrelevant shape with ground-truth mask. This result illustrates that it is very difficult to extract position information from the input image alone. However, when we input black, white and noise image, the scores of SPC and MAE are lower than inputting natural images. We boldly guess that the texture of the picture will affect the encoding of the position information.

3.2 THE EFFECT OF VARYING KERNEL SIZE AND ZERO-PADDING

In this subsection, the Fig. 4 shows the effect of kernel size and zero-padding in f_{pem} with the ground-truth mask **H**. We can see that the increasing size of kernel shows better performance in randomization

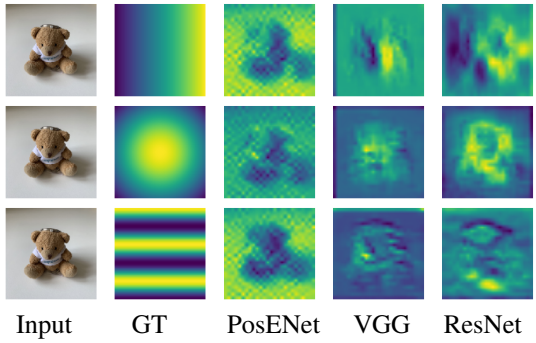

Figure 3: Some results of PosENet based networks corresponding to different ground-truth patterns.

test. Larger receptive fields readout of positional information further augments the readout of absolute position especially in PosENet. For the effect of zero-padding with 3×3 size kernel, we can see that the bigger number of zero-padding shows a slightly better performance in randomization test. And the zero-padding as an anchor from which spatial information is derived and eventually propagated over the whole image as spatial abstraction occurs. Both results match the result of the paper we chose in general.

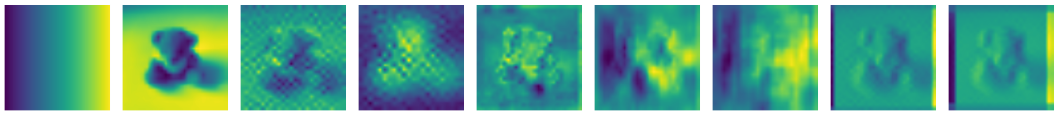

Figure 4: Top: The effect of varying Kernel Size (KS) and zero-padding on Horizon pattern. Left to right: GT (H); PosENet KS=1, PosENet KS=3, PosENet KS=7; ResNet KS=1, ResNet KS=3, ResNet KS=7; PosENet Pad=1, PosENet Pad=2.

4 CONCLUSION

In our reproduction, the result shows a similar performance with the paper we chose. In typical CNNs image classification tasks, the absolute position information can be encoded in models, and the larger kernel size and more zero-padding can lead positive influences to this process which were simply reproduced above. As we simplified some process and some details do not be provided in the paper we chose, which is the reason we did not reproduce the experiment perfectly. And we also found that the choice of test image will affect model performance and the paper selected a special image without too much patterns, which can lead a better performance. So we are slightly doubt for the robustness of the paper. However, we successfully get a similar result with the paper we chose. The authors also use those consequences to build a heat map of semantics, which we will explore in the future.

REFERENCES

Eugene S Edgington. Approximate randomization tests. *The Journal of Psychology*, 72(2):143–149, 1969.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256, 2010.

Md Amirul Islam, Sen Jia, and Neil DB Bruce. How much position information do convolutional neural networks encode? *arXiv preprint arXiv:2001.08248*, 2020.

Sen Jia and Neil DB Bruce. Eml-net: An expandable multi-layer network for saliency prediction. *Image and Vision Computing*, pp. 103887, 2020.

Lijun Wang, Huchuan Lu, Yifan Wang, Mengyang Feng, Dong Wang, Baocai Yin, and Xiang Ruan. Learning to detect salient objects with image-level supervision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 136–145, 2017.