Modification of single-purpose CNN for creating multi-purpose CNN

A V Shustanov¹, P Y Yakimov¹,²

¹Samara National Research University, Moskovskoe Shosse 34A, Samara, Russia, 443086
²Image Processing Systems Institute of RAS - Branch of the FSRC "Crystallography and Photonics" RAS, Molodogvardejskaya street 151, Samara, Russia, 443001

e-mail: alexander.shustanov@gmail.com

Abstract. Modern traditional convolutional neural networks (CNN) are built to solve one specific task, which can be: detecting and localizing objects in an image, restoring a human skeletal model, classifying images, etc. However, existing business problems can be a combination of the above subtasks. The simplest way to solve this problem is to use several ready-made CNN models or to train such models, one for each subtask. But this approach leads to a strong increase in the required computing power, which can be economically costly. The approach presented in the article allows reducing computational costs by adding new blocks to the existing CNN and creating a multi-purpose model.

1. Introduction

Deep learning techniques are very effective in solving computer vision problems [1, 2]. In particular, convolutional neural networks (CNN) are successfully used to solve a large number of computer vision problems, such as: self-driving cars, detection of cancers by medical photos, detection and localization of objects in photos. In some tasks, the quality of CNN’s work becomes higher than that of a person. However, the construction of CNN poses some difficulties, such as data preparation and the difficulty of training a neural network. For the first, long manual human labor is needed, for the second, computation power. In case the required data set cannot be obtained, you can use the learning transfer method [3], in which the neural network trained to solve one problem is trained on a new data set, to solve another task. This approach also reduces the learning time of the network.

2. Modern approaches to solving computer vision tasks

Now, many computer vision problems that may arise in front of a business are already solved and distributed as open source, on platforms such as github, bitbucket, etc. Although, a large number of tasks, that are not solved, may be solved rather quickly by modifying existing solutions. As an example, consider the problem of localization and detection of objects in the photo and localization and detection of fur seals from photographs from an aircraft [4]. The existing solution for the first task is rather simply adapted to solve the second one, using the transfer learning techniques.
2.1. Transfer learning
Conventional machine learning and deep learning algorithms, so far, have been traditionally designed to work in isolation. These algorithms are trained to solve specific tasks. The models have to be rebuilt from scratch once the feature-space distribution changes. Transfer learning is the idea of overcoming the isolated learning paradigm and utilizing knowledge acquired for one task to solve related ones. In this article, we will do a comprehensive coverage of the concepts, scope and real-world applications of transfer learning and even showcase some hands-on examples. To be more specific, we will be covering the following.

Transfer learning is a technique that allows to adapt and retrain existing neural network to solve the task, it was not designed to. Most of CNN used to classify images end in several fully connected layers. Number of neurons in last layer equals to number of predictable classes. Transfer learning technique implies replacing last layer (or a bunch of last layers) with one, that have as many neurons as there are classes in new task. This is possible since lots of computer vision tasks can be solved with the same low level features. The approach can dramatically reduce training time, and sometimes, improve prediction quality.

2.2. Objects detection and localization
There are several approaches to detect and localize objects on images. The most common of them are single shot detectors, that is used in such architectures as YOLO [5], SSD[6], RetinaNet[7], and two-stage detectors, that is used in Fast R-CNN[8], Mask R-CNN[9] and others. Two-stage detection approaches may achieve better prediction quality, but low computing performance. Two-stage detection includes proposal of so called regions of interest (ROIs) and classifying predicted regions by extracting CNN features from them. Opposite to it, single shot detection approaches are end-to-end methods that simultaneously predict object bounding boxes and their probabilities of being one of the classes.

2.3. Feature pyramid network
Detecting objects in different scales is challenging in particular for small objects. We can use a pyramid of the same image at different scale to detect objects. However, processing multiple scale images is time consuming and the memory demand is too high to be trained end-to-end simultaneously. Hence, we may only use it in inference to push accuracy as high as possible, in particular for competitions, when speed is not a concern. Alternatively, we create a pyramid of feature and use them for object detection. However, feature maps closer to the image layer composed of low-level structures that are not effective for accurate object detection.

Feature Pyramid Network (FPN) is a feature extractor designed for such pyramid concept with accuracy and speed in mind. It replaces the feature extractor of detectors like Faster R-CNN and generates multiple feature map layers (multi-scale feature maps) with better quality information than the regular feature pyramid for object detection. The visual data flow in feature pyramid network is presented on Figure 1.

![Figure 1. Data flow in feature pyramid network.](image)

2.4. Anchor boxes
Anchor boxes are a set of predefined bounding boxes of a certain height and width. These boxes are defined to capture the scale and aspect ratio of specific object classes you want to detect and are typically chosen based on object sizes in your training datasets. During detection, the predefined
anchor boxes are tiled across the image. The network predicts the probability and other attributes, such as background, intersection over union (IoU) and offsets for every tiled anchor box. The predictions are used to refine each individual anchor box. You can define several anchor boxes, each for a different object size.

The network does not directly predict bounding boxes, but rather predicts the probabilities and refinements that correspond to the tiled anchor boxes. The network returns a unique set of predictions for every anchor box defined. The final feature map represents object detections for each class. The use of anchor boxes enables a network to detect multiple objects, objects of different scales, and overlapping objects.

3. Related work
Consider we need to solve two separate tasks for some business problem. First task is object detection and localization and second task is scene classification. Both of them are already solved with admissible quality. For object localization task we have chosen RetinaNet model trained on Open Images Dataset. For scene classification task we have chosen AlexNet model trained on Places365 Dataset. We aim to build single model to tackle both tasks in one pass.

3.1. Open Images Dataset
Open Images is a dataset of ~9M images annotated with image-level labels, object bounding boxes, object segmentation masks, and visual relationships. It contains a total of 16M bounding boxes for 600 object classes on 1.9M images, making it the largest existing dataset with object location annotations. The boxes have been largely manually drawn by professional annotators to ensure accuracy and consistency. The images are very diverse and often contain complex scenes with several objects (8.3 per image on average). Open Images also offers visual relationship annotations, indicating pairs of objects in particular relations (e.g. "woman playing guitar", "beer on table"). In total it has 329 relationship triplets with 391,073 samples.

3.2. Places365-Standard Dataset
The train set of Places365-Standard has ~1.8 million images from 365 scene categories, where there are at most 5000 images per category.

3.3. RetinaNet model
RetinaNet is a modern one-stage detector that outperforms one of the best two-stage detectors Faster R-CNN. It has ResNet as a backbone with Feature Pyramid Network approach to feature extraction. At the end of the network there is two subnetworks for classification and bounding box regression. Also, RetinaNet uses special focal loss function to tackle class imbalance problem, which is very important due to OID class imbalance. The main idea of focal loss is to make lower penalties for low frequency classes and bigger penalties for high frequency classes. The focal loss is defined as $FL(p_t) = - (1 - p_t)\gamma \log(p_t)$, where $p_t$ is ground truth probability of class $t$.

![Figure 2. Focal Loss.](image-url)
4. Scene classification subnet

We propose novel method of creating multi-purpose network by connecting additional subnets to existing layers of RetinaNet model. Consider the motivation behind the approach. The scene presented on the image tightly correlated with objects, located on it. Naively, we can say, that if we see beer mug it is possibly a photo of a pub. Photo of several men and football ball most possibly represents a football stadium. Last layers of the RetinaNet already have embedded information about image context. The main idea is to convert this implicit information to correct scene classes.

First of all, as we know, RetinaNet converts an image to grid of bounding boxes prediction, and the grid size may vary depending on input image size. So we need to roll up this grid in to single cell, to receive one scene prediction for whole image. The operation, that can do it is a Global Average Pooling.

4.1. Global Average Pooling

Global Average Pooling (GAP) layers are used to reduce the spatial dimensions of a three-dimensional tensor, similarly to Max Pooling layers. However, GAP layers perform a more extreme type of dimensionality reduction, where a tensor with dimensions \( h \times w \times d \) is reduced in size to have dimensions \( 1 \times 1 \times d \). GAP layers reduce each \( h \times w \) feature map to a single number by simply taking the average of all \( hw \) values. The visual example of the layer operation is presented on Figure 3.

![Figure 3. Global Average Pooling.](image)

4.2. Scene classification sub-network

A scene classification sub-network is simple fully-connected network. We tried to use two different architectures to tackle the problem. First architecture takes output of \( \text{res5c} \) RetinaNet layer as an input, and second architecture takes output of \( \text{res5c} \) and \( \text{res4b35} \) RetinaNet layers. The architectures are presented on Table 1. For convenience, we named them one-branch architecture and two-branch architecture.

As seen from the table, the second model much bigger than first, as it uses features from different stages of RetinaNet features pyramid. As we aim to solve two problems simultaneously, only new subnet layers were trained and whole RetinaNet layers were frozen.

5. Experimental results

Proposed models were trained for nine epochs on Places365-Standard dataset. The training was performed on three Tesla-V100 GPUs. Accuracy and computation time measurements were performed on single Tesla-V100 GPU.

The prediction quality of proposed models is presented on Table 2. As we see, our two-branch model outperforms basic AlexNet model in Top-5 accuracy and underperforms in Top-1 accuracy. Let's compare models computation time.

As a result, we see that with slightly reduction in accuracy of scene classification task we achieve lower computation time. Two baseline models process one image for 164 ms, but one combined model processes the same image for 108 ms. However, we should remember, that not all tasks can be combined in such way. Selected problems ideally fits to each other, so we gain in performance a lot.
At Figure 4 there is a result of processing single image from Places365-Standard validation set. Detected objects with confidence more than 0.25 are bounded with boxes.

| Table 1. Proposed models architectures. |
|----------------------------------------|
| **One-branch architecture**            |
| Input  \textit{res5c} \textit{?x?x2048} | 2048 |
| Global Average Pooling                  | 1024 |
| Dense with Relu                         | 512  |
| Dense with Softmax                      | 365  |
| **Two branch architecture**             |
| First branch input  \textit{res4b35} \textit{?x?x1024} | 1024 |
| Global Average Pooling                  | 1024 |
| Dense with Relu                         | 512  |
| Second branch input  \textit{res5c} \textit{?x?x2048} | 2048 |
| Global Average Pooling                  | 1024 |
| Dense with Relu                         | 512  |
| Branches Concatenation 512+512=1024     | 512  |
| Dense with Softmax                      | 365  |

| Table 2. Proposed models accuracy on Place365-Standard Validation Set. |
|-------------------------------------------------|
| Model                            | Top-1 acc. | Top-5 acc. |
| Baseline Places365-AlexNet         | 53.17%     | 82.89%     |
| Proposed one-branch model          | 47.49%     | 79.22%     |
| Proposed two-branch model          | 52.47%     | 83.29%     |

**Figure 4.** Image with scene detected as Aquarium.
Table 3. Proposed models computation time for single image.

| Model                        | Computation time (ms per image) |
|------------------------------|----------------------------------|
| Baseline Places365-AlexNet   | 58                               |
| Baseline RetinaNet           | 106                              |
| Proposed one-branch model    | 106                              |
| Proposed two-branch model    | 108                              |

6. References

[1] Badrinarayanan V, Kendall A and Cipolla R 2017 Segnet: A deep convolutional encoder-decoder architecture for image segmentation *IEEE transactions on pattern analysis and machine intelligence* 39(12) 2481-2495

[2] Shustanov A, Yakimov P 2017 CNN design for real-time traffic sign recognition *Procedia engineering* 201 718-725

[3] Pan S J, Yang Q 2009 A survey on transfer learning *IEEE Transactions on knowledge and data engineering* 22(10) 1345-1359

[4] Kaggle NOAA Fisheries Steller Sea Lion Population Count URL: https://www.kaggle.com/c/noaa-fisheries-steller-sea-lion-population-count (20.05.2019)

[5] Redmon J 2016 You only look once: Unified, real-time object detection *Proceedings of the IEEE conference on computer vision and pattern recognition* 779-788

[6] Liu W 2016 Ssd: Single shot multibox detector *European conference on computer vision* 21-37

[7] Lin T Y 2017 Focal loss for dense object detection *Proceedings of the IEEE international conference on computer vision* 2980-2988

[8] Girshick R 2015 Fast r-cnn *Proceedings of the IEEE international conference on computer vision* 1440-1448

[9] He K 2017 Mask r-cnn *Proceedings of the IEEE international conference on computer vision* 2961-2969

Acknowledgements

This work was partly funded by the Russian Foundation for Basic Research – Project # 17-29-03112 ofi_m and the Russian Federation Ministry of Science and Higher Education within a state contract with the “Crystallography and Photonics” Research Center of the RAS under agreement 007-Г3/43363/26.