Completely Convolutional Neural Networks for semantic image segmentation

Narinder Singh Ghumman
University of Georgia

Spring 2019

Abstract

Convolutional Neural Networks (CNNs) are a variety of neural networks whose design is inspired from the organisation of neurons in the animal visual cortex. These networks of simple information processing units have been successful at recreating a lot of complex visual phenomenon, fueling breakthroughs across many tasks in Computer Vision. Deep CNNs dictate the state of the art on recognition tasks like image classification and object detection, and while these models had been in existence for a while, it is only with the recent advances in hardware performance and data availability that their true potential has come to light. Semantic image segmentation, once again a task in image recognition, is also of great interest because of its applicability to scientific and engineering problems like medical image analysis and autonomous vehicle navigation. This task requires classification at pixel level based upon the contents of the image and this paper looks at two completely convolutional neural network designs for the purpose.

1 Introduction

Convolutional Neural Networks are layered architectures that take an input volume and through a series of transformations, produce an output volume of potentially different dimensions. These transformations are non-linear in nature, organized as a convolutional layer followed by a non-linear activation layer. The convolutional layer is made up of a set of kernels that are learnt as part of the training process and almost always, these convolution-activation layers are interlaced with pooling layers that downsample the volume being transformed. For classification tasks, this downsampling works in favor as it reduces the dimensionality of the volume making it computationally feasibly to connect it with dense layers for generating class prediction vectors. For image segmentation tasks, however, the output volume generated by a network must have the same width and height and this paper looks into two possible designs for achieving that using completely convolutional neural networks. The first design is based off of the simple idea of completely doing away with pooling layers in order to retain the input dimensionality throughout the transformation process while the second design upsamples the downsampled volume through transpose convolution operations so as to match the input dimensions at output. The second design also starts with pre-trained weights for reasons that are brought to light by the failure of the first design.

2 Dataset

The dataset for training and evaluating the models was taken from a Kaggle challenge hosted by Airbus. The dataset consists of satellite images of sea also potentially capturing ships. The task is one of detecting ships in these images. Detecting ships can be seen as an image segmentation task that requires categorizing each pixel as either ship or non-ship.

The dataset was highly clean. The resolution across images was uniform although it would also be worth noting that variable resolution no longer remains to be a concern when we move to completely convolutional designs. The dataset was, however, highly imbalanced with only a quarter of the training instances containing any ships in them. This imbalance was further exacerbated at pixel level for only a ships were only a very small part of the scene. This data imbalance was addressed by only looking at the fraction of instances that had any ships in them (which were about 40,000) and also by choosing an appropriate loss metric. A metric like binary cross-entropy would've been unhelpful since correctly detecting sea pixels should not be equally awarded. With a random pixel being disproportionately likely to be sea, there's very little variability to model here and it could through off our modeling. Dice coefficient, a measure for similarity between two samples, was used instead,

$$Dice = \frac{2 \times TP}{(TP + FP) + (TP + FN)} \tag{1}$$

Dice co-efficient yields a value between 0 and 1 penalizing for misclassifying ships as sea and sea as ship. True negatives, i.e., correct classifications

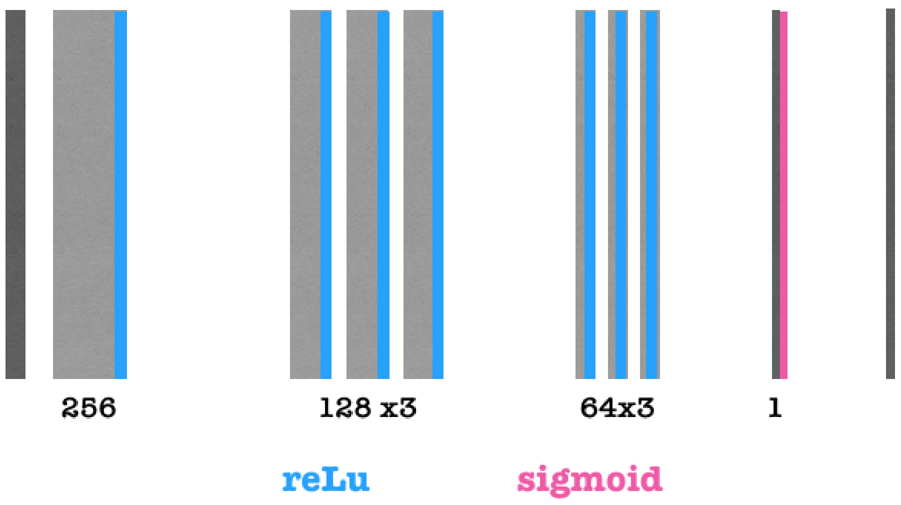

Figure 1: Only Convolutional Neural Network

of sea, do not help the model do better on this measure.

3 CNN Designs

3.1 Only Convolutional Neural Network

As outlined in introduction, the essence of this design is in completely doing away with downsampling layers of a typical convolutional neural network design. This results in the input volume retaining its width and height throughout the transformation process as shown in Figure 1. The kernel size was gradually decreased across layers to finally match the input dimensions with the intent that the network will build upon the primitive features of the earlier layers to develop more nuanced ones eventually leading to a 2-D output mask. The ReLU activation function was used for the early layers (to address the problem of vanishing gradient) and Sigmoid for the end layer to produce binary classifications.

This model when trained, was unable to learn anything of value. The dice coefficient peaked at less than 5% and this inability to learn, when analyzed was attributed immediately to a lack of parameters. None of the capable CNN architectures (refer Table 1.) have less than a few million parameters but this design had only *0.7 M* which greatly inhibited its ability to learn. Large parameter spaces enable modeling of complex decision surfaces; In fact, the primary result of the landmark paper by Krizhevsky et al. [1] from University of Toronto was that the large parameter space and the depth of a Deep Neural Network model were essential to its high performance. At the same time, it should not be construed as an impetus to increase the depth and parameters space of our design. The Imagenet dataset has over 14M images and is capable of feeding such large networks. We do not have this kind of data availability for our task.

Table 1: Error Estimates: REPTree

VGG19	13M
ResNet50	25M
InceptionV3	23M
ResNet152	60M

3.2 Encoder Decoder

The thesis underlying the second design was that given the wideness of the scope of the Imagenet competition, it must be the case that the early layers of the ResNet must be developing some good feature extractors which help help the succeeding layers classify multi milltion images with such high accuracy. These pre-trained layers could be taken out of ResNet to extract features from the images and then a non-linear upsampling transformation can be learnt form these features to generate masks. This was exactly what was done with the second design. All the ending dense layers were chopped off of the ResNet and replaced with an upsampling convolutional layer also known as de-convolutional or transpose convolutional layer. The weights for the ResNet were frozen and the new layers were trainable. Once again, sigmoid was used against the final block to produce binary classification results. The architecture is outlined graphically in Figure 2.

4 Results and Future Work

This network, when trained, showed a frozen loss value around 0.01 (refer Figure 3.). A grid search for the right learning rate was performed but only to get almost identical loss curves.

Training neural networks can be a challenging task and the poor performance can not be directly attributed to a poor architecture. While it might be the case that 1.6 M new parameters are not enough or that a single activation layer is not providing enough non-linear transformational power to learn the outputs, the fault might just as well be with how things are put together.

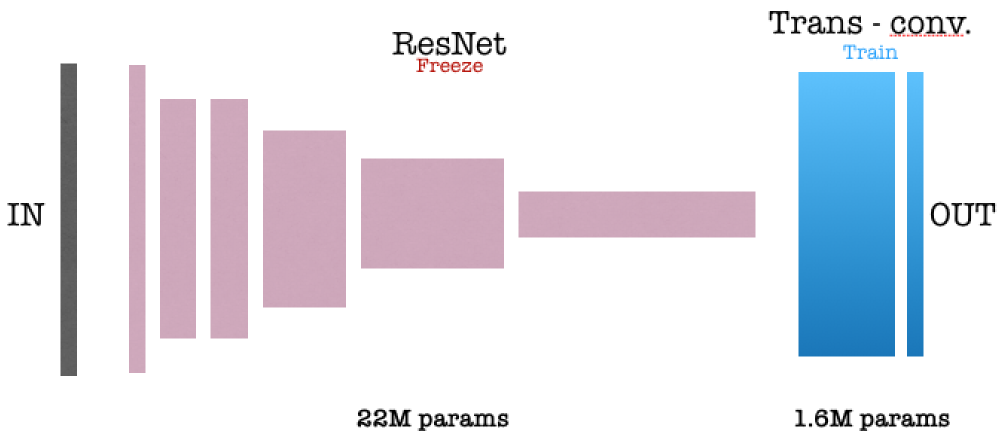

Figure 2: Encoder Decoder Network

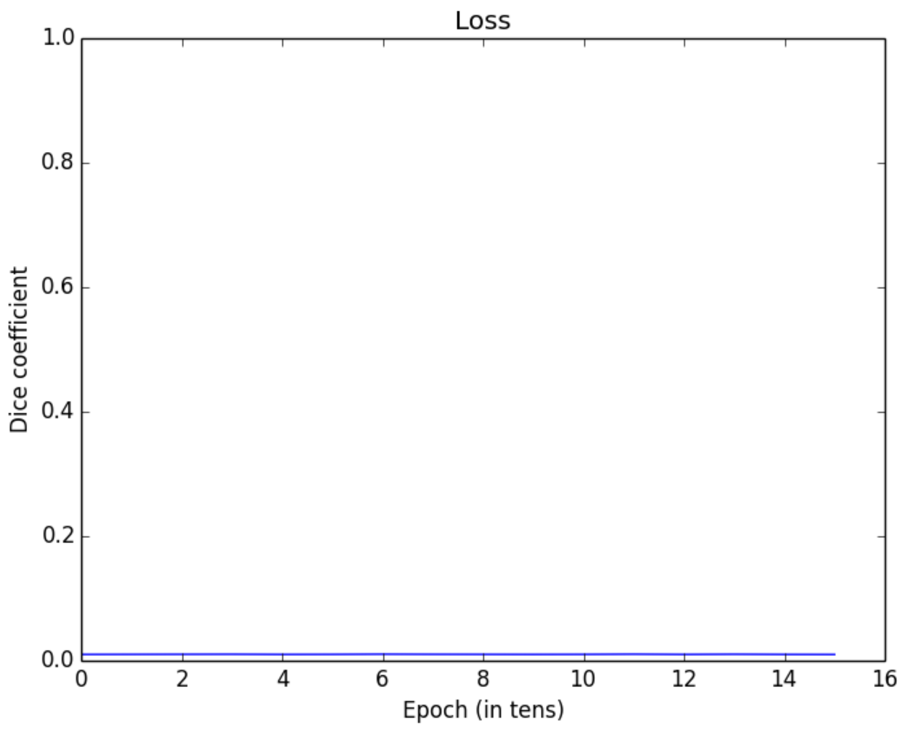

Figure 3: Encoder Decoder Network

ResNets were trained on 3-channel RGBs with mean centring applied to the inputs on a $0 - 255$ scale. So, one of the things that might be happening is that these inputs - even after initial transformations - are too large for the new activation layer and may very well be falling well outside the small range of activation for sigmoid, thus preventing a steepy descent. And this is just one reasonable speculation: other things could be going wrong at the interface. Neural Networks are sensitive artifacts with a lot of variables and perhaps combining multiple nets is not this straightforward but shouldn't be given up on either. In conclusion, the Encoder-Decoder CNN design for semantic segmentation has a strong underpinning thesis and should be subject to more careful study.

References

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

[2] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffer. Gradient Based Learning Applied to Document Recognition. Proceedings of the IEEE, Nov. 1998.

[3] https://www.kaggle.com/c/airbus-ship-detection

[4] I. Goodfellow, Y. Bengio, A. Courville. Deep Learning Book - MIT Press