Spiking Neural Networks

Arslan Salikhov Erik Caceros Brandon Lam

April 8, 2022

Abstract

Use of Deep Neural Network, commonly referred to as *deep learning* spiked in recent years and has been used as a tool for impressive advancements in the field of *Artificial Intelligence (AI)* Spiking Neural Networks draw inspiration from the Purpose of this project is to demonstrate the capabilities of a Spiking Neural Network and compare it to a more conventional Object Recognition Deep Neural Network

Contents

1 Architectures Of Deep Neural Networks

When it comes to building a neural network, one has to establish an architecture or an arrangement of layers and how they connect with each other. It is particularly important, when the network increases in complexity. Object classification + localization being a demanding task for a network to solve, we have adopted a published and well-tested architecture for the task – ResNet. The particular architecture we have tried to imitate titled *DECOLLE*, was first developed by Kaiser, Mostafa, and Neftci (2018) and later enhanced by Barchid, Mennesson, and Djéraba (2021). *DECOLLE* Neural Network adopts properties of classical ResNet model architecture as well as a pyramidal structure that was proposed by Lin et al. (2016).

1.1 ResNets

To begin with the cornerstone of our architecture – ResNet, it was named for its property, of convolutional layers of the network that are residually connected with each other using shortcut connections (He, Zhang, Ren, and Sun (2015)). According to the authors of the original paper on ResNets, addition of residual connections improves model's convergence, or ability for model's loss to move towards a minimum with a decreasing trend. Meanwhile, ResNets converge faster than their plain counterparts, the complexity of ResNets is much lower than complexity of well established Visual Geometry Group (VGG) Networks given the same depth (3.6 billion FLOPs (Float Point Operations) vs. 19.6 billion FLOPs) (He et al. (2015)).

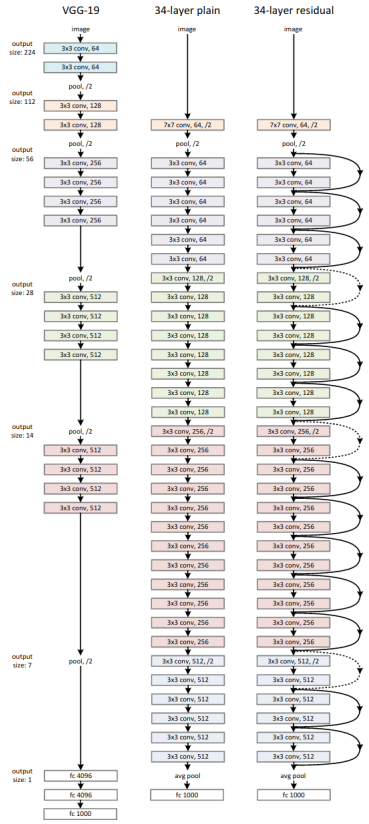

Figure 1: Examples of 16-layer VGG, Plain 34-layer, and 34-layer ResNet Network Architectures

1.2 Comparing Model Performance

To compare networks' performance one has to understand how it is measured and what constitutes an accurate model. For the task of object detection/localization accuracy of the model at predicting class of the object in the image is not enough. Thus, two additional performance metrics have been adapted : Mean Average Precision (mAP) and Intersection of Union (IoU). Firstly, Mean Average Precision for the pur-

poses of Object Detection is well-defined in the paper by Everingham, Gool, Williams, Winn, and Zisserman (2010). According to the authors, mAP is mean of Average Precision of the model for all the classes in the dataset. It is calculated using the following equation.

$$\mathbf{mAP} = \frac{1}{N} * \sum_{i=1}^{N} \mathbf{AP}_i \tag{1}$$

AP = Average Precision,

N = number of classes.

Next, IoU is the area of overlap of ground-truth boxes and the model's detected box output, the higher the IoU the better performance of the model. Intersection of Union can be represented as follows:

$$\mathbf{IoU} = |A \cap B| \div |A \cup B| = |I| \div |U| \tag{2}$$

I = Intersection area,

U = Union area.

1.3 Pyramidal Architecture of Neural Network

Pyramidal structure of a Neural Network refers to the shape of the layers as the model gets deeper. With each layer dimensions of convolutional layers shrink (usually in half). For example, the first convolutional layer would have input dimension equal to number of channels in the image (3 channels in case of RGB images vs. 1 for grayscale) and the output dimension of 512, with next layer input equal to previous layer's output. Second layer would have the dimension of 256 (half of previous layer's output) and so on (Lin et al. (2016)).

For the purposes of our network, pyramidal structure of neural network is leveraged using Encoder-Decoder model. When image is passed through the shrinking layers of the Encoder Pyramid, it loses resolution as well as contextual information, however when the image leaves the encoder and is located in the hidden state it is fully comprised of the semantic information about the image (What object is on the picture and where it is located). Furthermore, in order to localize an object on the image that model has never encountered before, it has to be able to place the semantic information in context. This is where Decoder comes into play, being the reverse of an Encoder, the convolutional layers of decoder double in size until desired dimensions. The decoder is residually connected to encoder, and thus it gains contextual information about the image through every pass of a layer.

At the end, in order to extract meaningful, last decoder layer is connected to fully connected linear layers that output predicted class as well as detected box coordinates.

1.4 Architecture of the Model Used

Architecture developed by Barchid et al. (2021) was the target to replicate. The model is based on Encoder-Decoder architecture with residually connected layers akin to ResNets. Encoder and Decoder follow the pyramidal architecture that was discussed earlier.

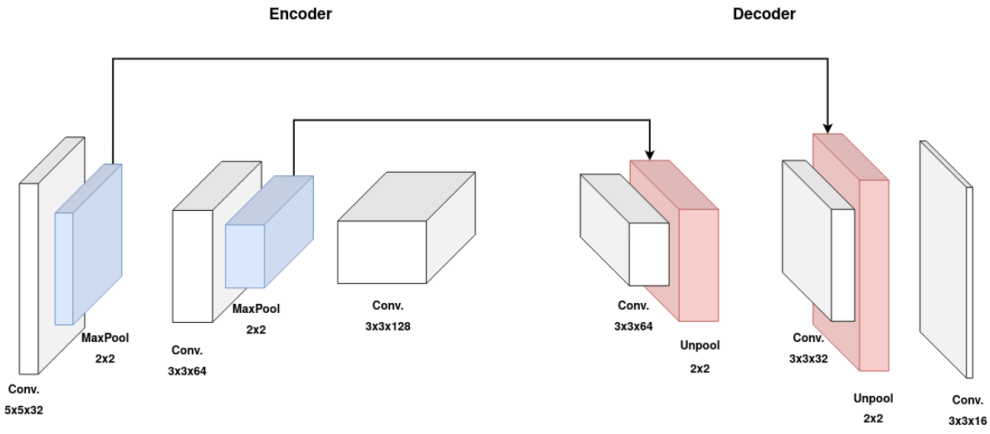

Figure 2: Sketch of the model's architecture. White layers represent convolutional layers, blue represent pooling layers, and red unpooling layers

2 Implementation

2.1 PyTorch

The main package that allowed us to develop such a complex model was PyTorch. PyTorch is an open-source a machine learning/deep learning library and framework for Python that allows user to develop and ship high performance models (Paszke et al. (2019)). This library eliminates the need from scratch development of the most important methods in machine learning, such as activation functions, model architecture assembly,

2.2 Spiking Neurons

Given the complexity of Spiking Neural Networks and the integration of Leaky-Integrate-and-Fire into the network, we used a package that allows to use LIF within a PyTorch model – *snnTorch*. snnTorch also allowed us to integrate backpropogation for spiking neural networks with one line. Indeed, to use LIF in your model only need to use one line in model definition.

```
import snntorch as snn


lif1 = snn.Leaky(beta=beta, spike_grad=spike_grad)
```

Listing 1: Leaky-Integrate-and-Fire using snnTorch

3 Challenges

With the number of operations required to train the model to complete the task of object localization, it was clear since the beginning that a lot of compute power was required. A number of steps was required in order to make this project plausible the scope of this course.

First challenge in implementation of the model was difficulty to replicate the Barchid et al. (2021) paper given how the lack of description on the implimintation in the paper as well non-existence of public repository with the code for their model.

References

Barchid, S., Mennesson, J., & Djéraba, C. (2021). Deep spiking convolutional neural network for single object localization based on deep continuous local learning. In *2021 international conference on content-based multimedia indexing (cbmi)* (pp. 1–5).

Everingham, M., Gool, L., Williams, C. K., Winn, J., & Zisserman, A. (2010, jun). The pascal visual object classes (voc) challenge. *Int. J. Comput. Vision*, *88*(2), 303–338. Retrieved from `https://doi.org/10.1007/s11263-009-0275-4` doi: 10.1007/s11263-009-0275-4

He, K., Zhang, X., Ren, S., & Sun, J. (2015). *Deep residual learning for image recognition.* arXiv. Retrieved from `https://arxiv.org/abs/1512.03385` doi: 10.48550/ARXIV.1512.03385

Kaiser, J., Mostafa, H., & Neftci, E. (2018). Synaptic plasticity dynamics for deep continuous local learning (decolle). *arXiv preprint arXiv:1811.10766*.

Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2016). *Feature pyramid networks for object detection.* arXiv. Retrieved from `https://arxiv.org/abs/1612.03144` doi: 10.48550/ARXIV.1612.03144

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., ... Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems 32* (pp. 8024–8035). Curran Associates, Inc. Retrieved from `http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf`