CSCI 8960 Term Project Proposal

1 Team Members

Josh Messitte, Eli Burstiner

2 Methodology

ResNet is a convolutional neural network which became popular for tasks like object recognition when it made it possible to construct networks with up to thousands of layers, overcoming the ”vanishing gradient” problem. The main component of ResNets are called residual blocks or building blocks. Residual blocks help make deep convolutional networks possible by facilitating skip connections, where the ResNet adds intermediate inputs (often the output from some previous layers) to the output of some group of blocks (He, 2015). We are considering this model due to to the many prior implementations and the fact that it is very competitive with state-of-the-art accuracy. A 110-layer ResNet was used for classification error on the CIFAR-10 dataset and acheived a 6.43 percent error which, at the time, rivaled state-of-the-art models (He, 2015).

EfficientNet is a convolutional neural network which is unique in how it scales the three dimensions of the network: depth (number of layers), width (channels in a single layer), and resolution (of input image). The main component of the EfficientNet architecture is the compound coefficient, which is used to uniformly scale all three dimensions of the model. This model has achieved state-of-the-art accuracy on CIFAR-100 and other image classification tasks in the past. For example, EfficientNet-B7 achieved a top 1 percent accuracy of 84.3 percent (Tan, 2020) on ImageNet, a new state-of-the-art level.

Our training objective is to implement ϵ-DP Root Means Squared Propagation (RMSprop), an extension of Gradient Descent-based algorithms. PyTorch's implementation takes the square root of the gradient average prior to adding ϵ.

3 Experimental Setup

For this project, we plan to use the Google Colab environment to train and test our models and display our results. We also will make use of their free access to GPUs. EDIT: Google Colab has a GPU usage limit, so we also are training our model using SaturnCloud, which also supports Jupyter-lab functionality with GPUs.

We also plan to to use PyTorch as our deep learning framework. PyTorch will allow us to use its many built-in models (including the two discussed in section 2 above) in the torchvision package as well as Opacus, a PyTorch library that will facilitate training different models with epsilon differential privacy.

There are a couple key performance metrics that we will be measuring in our project.

- Accuracy:
$$\frac{\text{Correct Predictions}}{\text{Total Predictions}} * 100$$
- Precision:
$$\frac{\text{Airplanes identified correctly}}{\text{Airplanes identified correctly} + \text{Airplanes identified incorrectly}}$$

4 References

- He, K., Zhang, X., Ren, S., Sun, J. (2015). Deep Residual Learning for Image Recognition. arXiv [cs.CV]. Opgehaal van http://arxiv.org/abs/1512.03385
- Tan, M., Le, Q. V. (2020). EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. arXiv [cs.LG]. Opgehaal van http://arxiv.org/abs/1905.11946