CS 294-82 project proposal

Zeyu Liu, Jiaxin Cheng, Qiqin Zhao, Zhekun Luo

Abstract

Based on information theory, deep learning can be viewed as doing data compression. It encodes/decodes input data into some inner structures in some high dimensional feature space, and then cluster them to do classification. Ideally, the neural network should be trained to extract the underlying structure of the input data and use such information to predict. Such inner structures represent the "compressed" version of the input data and can be used to classify unseen data, which is called generalization. The capacity of the network is essential to the model's ability to generalize. If the network has too many parameters, it will tend to lazily remember all the training samples, which leads to zero generalization, aka overfitting. By restricting the capacity of the network, we are effectively forcing the network to generalize. We want to investigate the model's generalization behavior via the control of network capacity. In addition to benchmarking the network's performance on test samples, another critical evidence of overfitting is the existence of adversarial samples. If the model memorizes the samples instead of learning the inner structure from the data, it would be quite vulnerable to adversarial attacks. So we proposed to compress some standard pre-trained models further and further, and then observe their generalization via robustness in terms of adversarial attack. Based on that, we proposed a novel black-box attack method, namely the "OutlierAttack," which aims to pick out the data outliner and use them to make attacks. Intuitively, the training samples which can only be classified correctly by a large network but not a small one is the "data outliner." It sits far from the distribution of the data clusters of its class. As a result, it is highly likely that the model classifies them correctly by memorizing these outline samples. Such samples can be exploited to make attacks on the model.

Generalization and model robustness

The problem we want to deal with is to verify the theory that the compressed model has a better ability to generalize. We want to do the following experiment. We want to generate adversarial examples from the training data to “attack” the model and see what will be the effect is. Secondly, we want to attack the model the same way by training the compressed model with the same amount of adversarial examples. We should expect the effect that the compressed model should be more robust and less prone to adversarial examples attack. More specifically, we will use the CW attack to conduct our experiments. In their paper “Towards Evaluating the Robustness of Neural Networks”, Nicholas Carlini and David Wagner proposed an attack method which frames adversarial attacking as an optimization problem, and to iteratively find the smallest perturbation to the natural samples to generate adversarial samples. When we use the CW attack, the L2/L infinity distance indicates the difficulty of the attack, which further implies the robustness of the model. Therefore, we will calculate the average distance for each attack and expect an increase of L2/L infinity distance as we use a more compressed model.

The OutlierAttack

What we want to do is to simulate a kind of adversarial attack to demonstrate the weakness of the original Neural Networks. We first get our trained model A and a ‘compressed’ version of model A, denoted as model B. The compressed model B should have a slightly worse performance given the training set. However, the difference should not be great. We have already known that a robust neural network would trust generalization other than memorization since memorization would cause overfitting and has no implications of future predicting. The problem is we want to know the proportion of generalization and memorization in both neural networks. Based on the theory we learned from class, the capacity of neural network matters for generalization. The neural networks that have excessive capacity tend to memorize datasets other than generalizing. Therefore, when we try to fit both models with our training dataset, there is a subset of data that model A correctly classifies, but model B does not. This implies model A adapts memorization rather than generalization, which could not be a good example of a robust neural network. This could be the start of our adversarial attack.

The reason why we choose these special samples is that when we perform tiny perturbation on these datasets, they could reveal the flaws of the original model more convincingly, since we consider these datasets as memorization in model A. If model A fails to classify the perturbed sample at most cases (The metric is far worse than with the previous original data), then it definitely applies memorization rather than generalize.

Implementation details

We learn from the class that the neural network tends to memorize the dataset as the learning layers increases. Additionally, as the layers increase the repeated multiplication that happened during gradient back-propagated to the early layers makes the gradient very small. One significant issue of these vanishing gradient is that our neural network performance would start degrade rapidly. It also takes more time to train the model as the layer increases. To tackle these problems, we will use the residual network(ResNet) with 50/100/150 layers which could efficiently test our theory that a model with lower capacity would have better generalization ability. ResNet not only has very powerful representational ability but also train hundreds or thousands of layers in a reasonable time with compelling performance. For the dataset, we use samples from the ImageNet which provide sufficient examples to train in deep neural network. In addition, we would have the flexibility of generating the adversarial examples that could be used to “attack” the models. From the class, we learn that to ensure label balances, we will provide label for each sample from the ImageNet. We will make 1 for odd sample number, and 0 for even sample number.

The adversarial attack implementation can reference the cleverhans toolbox.
https://github.com/tensorflow/cleverhans

Related works

https://arxiv.org/abs/1608.04644
https://arxiv.org/pdf/1706.06083.pdf