Project #8
Adversarial Robustness in Machine Learning Models

Modeling Report

Members:
Sai Coumar - 25%
Patrick Florendo - 25%
Nischay Uppal - 25%
Supriya Dixit - 25%

1. SELECT MODELING TECHNIQUE

As the first step in modeling, select the actual initial modeling technique. If multiple techniques are to be applied, perform this task separately for each technique.
Remember that not all tools and techniques are applicable to each and every task. For certain problems, only some techniques are appropriate. It may be that only one tool or technique is available to solve the problem at hand—and that the tool may not be absolutely the best, from a technical standpoint.

1.1. Modeling technique

Residual Networks (ResNet18) are a form of deep learning that pass the original data as a residual block of input throughout the network to prevent image information loss through the propagation of the model layers in training. This was selected as our control model as it's a well-researched model known to be successful for standard classification tasks. Normal CNNs can suffer from information loss, and more traditional machine learning is not well-suited for image classification tasks. The ResNet18 in particular was selected as it's the simplest variation of the ResNet family of models and no significant performance benefits are found by using larger ResNet models. We were recommended this as a choice by Rajdeep.

Adversarial Attacks:

Several Adversarial Attacks will be implemented to test efficacy against our control model and then will be used in the evaluation phase to retrain the models to gauge robustness. The following attacks will be used

White Box Attacks:

- Fast Gradient Sign Method
- Projected Gradient Descent
- Carlini & Wagner
- Jacobian Saliency Map
- DeepFool

Black Box (Score-Based):

- Natural Evolution Strategies
- Square Attack

Black Box (Decision Based):

Boundary Attack

HopSkipJumpAttack

1.2. Modeling assumptions

Many modeling techniques make specific assumptions about the data.

- Define any built-in assumptions made by the technique about the data (e.g., quality, format, distribution)
- Compare these assumptions with those in the Section 2 (Describe Data) of the Data Understanding report
- Make sure that these assumptions hold and go back to the Data Preparation report, if necessary

It is assumed that all data is in the form of integers. Labels range from 0-9 for MNIST and CIFAR10, and 1-10 for SVHN. Pixel values range from 0-255. This is consistent with the descriptions from section 2 of the Data Understanding report. This assumption leads us to modify the implementation of the attacks since most papers normalized prior to the attack, but we normalized during classification using batch normalization so we need to account for the shift in normalization timing in our attack implementation with a few attacks.

For adversarial attacks, it is assumed that the control model is accurate prior to the attack so that we know that our results from the attack are from attacking the model, and not the model being a poor classifier.

We also assume that the data does not contain outliers.

2. GENERATE TEST DESIGN

Prior to building a model, it is necessary to define a procedure to test the model's quality and validity. For example, in supervised data mining tasks such as classification, it is common to use error rates as quality measures for data mining models. Therefore, the test design specifies that the dataset should be separated into training and test sets. The model is built on the training set and its quality estimated on the test set. Describe the intended plan for training, testing, and evaluating the models. A primary component of the plan is to decide how to divide the available dataset into training data, test data, and validation test sets. • Check existing test designs for each data mining goal separately

- Decide on necessary steps (number of iterations, number of folds, etc.)
- Prepare data required for test

Each of the datasets already have partitions (of varying proportions) for testing and training which we used to stay consistent with the existing research in the domain. The ResNet control models were trained on the training dataset for each of the three datasets and were tested using the test dataset to evaluate model accuracy. Then, adversarial attacks attacked the test dataset, and the attacked test data was put through the classifier to quantify how much model classification accuracy decreases on attacked images with some adversarial attack.

No accuracy improvement techniques were applied. Image transformations and bootstrap sampling (not bagging) were attempted, but decreased accuracy marginally and we defaulted to the standard dataset.

3. BUILD MODEL

Run the modeling tool on the prepared dataset to create one or more models.

3.1. Parameter settings

With any modeling tool, there are often a large number of parameters that can be adjusted. List the parameters and their chosen values, along with the rationale for the choice.

- Set initial parameters
- Document reasons for choosing those values

1) Resnet18 Models were trained for 5 epochs with the adam optimizer using a learning rate of 1e-3. Batch sizes of 256 were used. These hyperparameters were common in implementations in other work and gave us relatively quick results.

2) White Box Attacks:

FGSM: Epsilon is the only hyperparameter, and in early testing we used varying epsilons from 0-0.5. A middle ground must be chosen per dataset depending on the results.

Deepfool: overshoot was set to 0.02 (recommended in the source material) and Max_iterations was set to 100 as a balance of runtime and efficacy

PGD: Epsilon was set to 8 and alpha was set to 0.05. PGD hyperparameters needed to be manually verified and performance didn't seem to match the original paper, likely due to differences in normalization implementations

CW: the c constant to balance perturbation and perturbation apparency was set to 0.3, learning rate set to 0.05. In the paper, c is selected via binary search to be > 0.1 and <1 where it is most effective. Kappa is the "confidence" in the paper which is the confidence with which the misclassification occurs.

JSMA: Theta, the "perturb length" after the saliency map is obtained from the gradient.

3) Score Based Black Box Attacks:

NES: Hyperparameters from PGD were repeated as NES functionally works the same with gradient estimation instead of extraction

4) Decision-Based Black Box Attacks:

Boundary Attack: Hyperparameters are dynamically adjusted as the attack is run. Initial parameters are set to epsilon = 1 and delta = 0.1 to balance runtime and efficacy of the initial iterations.

HopSkipJumpAttack: HSJA is designed to be hyperparameter-free, relying only on classification output to find the decision boundary and effective perturbations

3.2. Models

Run the modeling tool on the prepared dataset to create one or more models.

- Run the selected technique on the input dataset to produce the model
- Post-process data mining results (e.g., edit rules, display trees)

We reimplement the listed models/attacks from scratch:
ResNet18 model architecture was duplicated and matched abstracted PyTorch implementations but added visibility made later tasks easier (ex. Gradient extraction)

The scripts created for the ResNet models are:

cifar10_resnet18.ipynb
mnist_resnet18.ipynb
svhn_resnet18.ipynb

The scripts created for the attacks are:

model_architectures.py
attacks.py
test_perturbations.ipynb
boundary-attack.py

ResNet18 model weights are saved to the artifacts folder as .pth files. Adversarial attacks don't require any training to save but will be used later to retrain the ResNet18 models with more robust data in the evaluation phase.

ResNet models output probabilities of each of the classes and are argmaxed to pick the classified label by the model.

3.3. Model description

Describe the resulting model and assess its expected accuracy, robustness, and possible shortcomings. Report on the interpretation of the models and any difficulties encountered.

• Describe any characteristics of the current model that may be useful for the future •
Record parameter settings used to produce the model

- Give a detailed description of the model and any special features
- For rule-based models, list the rules produced, plus any assessment of per-rule or overall model accuracy and coverage
- For opaque models, list any technical information about the model (such as neural network topology) and any behavioral descriptions produced by the modeling process (such as accuracy or sensitivity)
- [Optional] Describe the model's behavior and interpretation
- [Optional] State conclusions regarding patterns in the data (if any); sometimes the model reveals important facts about the data without a separate assessment process (e.g., that the output or conclusion is duplicated in one of the inputs)

ResNet18: A well-researched image classification deep learning model that circumvents the issue of vanishing gradient in deep learning models by periodically passing in the input data as a residual block through skip connections. The 18-layer variant was chosen as it was the simplest and more layers did not give significant accuracy improvements. A modification was made to the output layer to list class probabilities (as opposed to binary classification with a softmax)

White Box Attacks: Highly effective adversarial attacks that often easily reach perfect attacks by using information from the model to generate a perturbation quickly and with minimal noise. Less practical than Black Box attacks in real-world scenarios.

Black Box: Difficult alternatives that allow for attacks with the constraints of the limited information available. Requires more computation power to search for solutions without free access to the original model. Challenging to get satisfactory results through implementation. Less effective than White Box attacks.

Our models lack interpretability. ResNet models are deep learning models that present little to no usable information about why classifications were made. Unfortunately, models with high interpretability are ineffective in image classification due to the nature of image data. Decision boundaries exist in higher dimensions that cannot be perceived. Adversarial attacks aim to move images around these decision boundaries without image degradation but these decision boundaries cannot actually be perceived and aren't interpretable.

4. ASSESS MODEL

The model should now be assessed to ensure that it meets the data mining success criteria and passes the desired test criteria. This is a purely technical assessment based on the outcome of the modeling tasks.

4.1. Model assessment

Summarize results of this task, list qualities of generated models (e.g., in terms of accuracy), and rank their quality in relation to each other.

• Evaluate results with respect to evaluation criteria

• Test result according to a test strategy (e.g.: Train and Test, Cross-validation, bootstrapping, etc.) • Compare evaluation results and interpretation

• Create ranking of results with respect to success and evaluation criteria

• Select best models

• Interpret results in business terms (as far as possible at this stage)

• [Optional] Get comments on models by domain or data experts

• Check plausibility of model

• Check effect on data mining goal

• Check model against given knowledge base to see if the discovered information is novel and useful

• Check reliability of result

• Analyze potential for deployment of each result

• If there is a verbal description of the generated model (e.g., via rules), assess the rules: Are they logical, are they feasible, are there too many or too few, do they offend common sense? • Assess results

• Get insights into why a certain modeling technique and certain parameter settings lead to good/bad results

For ResNet models, we used test accuracy as a metric of success. Each dataset got its own ResNet18 model with varying accuracy

MNIST ResNet18 Accuracy: ~99%

CIFAR10 ResNet18 Accuracy: ~78%

SVHN ResNet18 Accuracy: ~93%

We use test accuracy following perturbations and inspect manually sampled perturbed images to evaluate the effectiveness of adversarial attacks. Adversarial attacks need to balance a tradeoff of preserving the image quality while perturbing it enough to fool a classifier. Unfortunately, evaluating image fragmentation requires human observations, which makes quantifying results somewhat difficult and finding “optimal” hyperparameters difficult. Accuracy for algorithms also varies from dataset to dataset.

Adversarial attacks had different performances that suit different circumstances. For example, Black Box algorithms are universally worse than White Box methods but are more applicable in real-world scenarios when White Box attacks are unavailable. Some White Box methods such as DeepFool are considered “perfect” attacks and always return a positive result but require many more iterations, whereas others such as FGSM make quick attacks, but have accuracies that can range from 40% to 0% depending on parameters and different datasets. By comparison, Black Box methods, which rely mainly on the final classification of the images, cannot be tuned to achieve perfect results without sacrificing a noticeable loss in image quality.

4.2. Revised parameter settings

According to the model assessment, revise parameter settings and tune them for the next run in the Build Model task. Iterate model building and assessment until you find the best model. • Adjust parameters to produce better models.

Since our ResNet model was so large, we could not perform exhaustive hyperparameter tuning techniques. We tested with a variety of hyperparameters and saw either little changes in accuracy or significant decreases in accuracy. We settled on the recommended parameters from Rajdeep and increased epochs for our final models as deep neural networks do not overtrain due to double descent.

Resnet18 Models were trained for 20 epochs with the adam optimizer using a learning rate of 1e-3 and a step learning rate scheduler with step_size = 5 and gamma = 0.5. Batch sizes of 256 were used. These hyperparameters were chosen after extensive manual testing along with recommendations by Rajdeep.

Parameters for adversarial attacks must be tuned based on desired image quality preservation and available computational resources. We maintained the same parameters from before as an arbitrary choice for now and will make more revisions for the evaluation phase. Parallelization of the attacks (unviable during initial implementation) could perhaps change how many resources are available and what parameters are selected.