DSCI 6015: Artificial Intelligence and Cybersecurity

Spring 2024

Midterm Project

Cloud-based PE Malware Detection API

The purpose of this term project is to demonstrate your practical skills in implementing and deploying machine learning models for malware classification. The technical implementation of this project is comprised of three main tasks that need to be completed sequentially:

Task 1 – Building and Training the Model: In this task, you will be creating and training a deep neural network based on the MalConv architecture to classify PE files as malware or benign. As for the dataset, you will be using the EMBER-2017 v2 (https://github.com/endgameinc/ember). Besides the references provided in this repository, the following two talks at BSides San Francisco 2018 and the CAMLIS 2019 conferences present detailed overviews of this dataset, as well as hints on how to use EMBER to train malware classifiers:

- https://youtu.be/TzW_R36iv48
- https://www.youtube.com/watch?v=MsZmnUO5lkY

If you explore the EMBER repository, you will find that it comes with a sample implementation of MalConv (https://github.com/endgameinc/ember/tree/master/malconv). This sample is a wonderful resource to base your implementation on. However, note that this code is 3+ years old (i.e., a lifetime in ML), and does not precisely conform to the requirements of this project.

- **Implementation:** The model must be implemented in Python 3.x using PyTorch (2.x), and needs to be coded and documented in a Jupyter Notebook. ***A skeleton code is provided*** in the Midterm module on Python. However, using the skeleton code is optional. Similar to the notebooks we've seen in the exercises, you must also add textual description blocks to the notebook to document and explain the different parts of your code.

- **Training:** This model may take a long time to train on your personal computers (from a few hours to a couple of days, depending on your hardware), unless you already have a powerful NVIDIA GPU (1080 TI or better). Alternatively, you can use the following cloud platforms to speed up the training:

  - **Google Colab –** You are already familiar with Colab, but here are a couple of tutorial if you need to refresh your memory:
    - https://youtu.be/inN8seMm7UI
    - https://youtu.be/vVe648dJOdI

- **Amazon AWS Sagemaker:** You have received an invitation to join a course named “AWS Academy Learner Labs” on awsacademy.instructure.com. This platform provides each of you with $100 worth of free AWS credits, as well as some training materials and instructor support. Please sign up to this course and watch the 3 tutorial videos in the Midterm module to learn about using this platform. Please note that this platform is hosted on a different Canvas site than the university’s, so you will need to create a new account to join. Since you have $100 worth of AWS credits in this platform, you can also use the AWS Sagemaker to train your models on AWS. We have not talked about Sagemaker in the class before, and if you wish to go with option, you will need to spend some time on learning the basics of Sagemaker –which will be worth the time, having Sagemaker on your resume makes it far more attractive to employers. Here are some tutorials on how to use Sagemaker to build and train PyTorch models:
  - Introduction to Amazon SageMaker: https://youtu.be/YcJAc-x8XLQ
  - Getting Started with Amazon SageMaker: https://sagemakerexamples.readthedocs.io/en/latest/intro.html
  - Train an MNIST model with PyTorch: https://sagemakerexamples.readthedocs.io/en/latest/frameworks/pytorch/get_started_mnist_tra in_outputs.html
  - PyTorch in SageMaker: https://docs.aws.amazon.com/sagemaker/latest/dg/pytorch.html

IMPORTANT:
Make sure that you monitor the spending of your charges on AWS. You only have $100 of credits and you will need to keep some of it for the
next tasks. Ideally, you should limit the cost of training to $10. See the following tutorial to learn how to define budgets on AWS:
https://aws.amazon.com/getting-started/tutorials/control-your-costsfree-tier-budgets/

- **Post-Training:** Once your model is trained, save and store the model. Then, create a function (or method) that takes a PE file as its argument, runs it through the trained model, and returns the output (i.e., Malware or Benign).

Task 2 - Deploy your model as a cloud API: In this task, you will be using Amazon Sagemaker to deploy your model on the cloud, and create an endpoint (~ API) so that other applications can make use of the model. While this might sound very complicated, you will find that it is actually quite simple to deploy models using Sagemaker. To learn about the procedure, you can follow these references:

- Deploy Models for inference: https://docs.aws.amazon.com/sagemaker/latest/dg/deploymodel.html

- AWS SageMaker Tutorial | Build and Deploy a Machine Learning API: https://youtu.be/OfzAl3K0s0U?si=YqXL_nR-r0P2M5Da&t=2121

Task 3 – Create a client: This task is quite simple as well: create a web application using Streamlit, in which the user uploads a PE file, then the application converts it into a feature vector that is compatible with your MalConv/EMBER model, runs the vector on the cloud API, and then prints the results (i.e., Malware or Benign – or probabilities of each). You can find a sample implementation here: https://github.com/endgameinc/ember/tree/master/malconv

Simplified Mid-Term

1. Deploy the model trained for Lab 5.4 as an API endpoint on AWS SageMaker.
2. Develop a Python client which takes in an executable file, extracts relevant features, and retrieves classification results from the SageMaker endpoint.
3. Test your client and endpoint with one malware PE file and one benign PE file from the test dataset (created during Lab 5.4) and demonstrate it in your demo video.

Deliverables:

- **Presentation and demo video:** A short (<15m) video, in which you describe the technical details of your project, and demonstrate your implementation of all three tasks. You can assume that your audience are peers who want to learn how to replicate your work.
  - To record the video, you can use any screen recording software. The default option is Zoom (share screen, and record).
  - The video must be uploaded on YouTube. However, if you wish to keep the video private (i.e., not visible to anyone outside of the class), you can post it as an "unlisted" video (see: https://support.google.com/youtube/answer/157177?hl=en&ref_topic=9257428).
- **Report:** A report, between 4 to 10 pages, comprised of:
  - An overview of the project and requirements ○ Details of your technical approach for each task
  - Performance analysis (e.g., accuracy, precision, recall, F-1, confusion matrix, average latency of malware detection with the API, etc.)
  - References (bibliography)
- **GitHub Repository:** Create a dedicated GitHub repository for this project, push the corresponding files (incl. the notebooks, code files, and report) to this repository, and create a README.MD file to introduce the project, describe the file structure, and provide links to your report and video demo.

  - You can make this repository private if you wish, but note that the entire point of these deliverables is to help with building your online portfolio.

Submission: Please submit a clone of your repository as a .zip file. Also, include a link to your repository in submission description. Note that the repository must contain the report, as well as a link to the presentation.