Cloud-based PE Malware Detection API

University *of*
New Haven

Under the Guidance of

Prof. Vahid Behzadan

October 14, 2021

Muvva Mukesh

This document talks about how to design a Cloud Based PE Malware Detection API

This project deals with the Ember dataset. The EMBER dataset is a collection of features from PE files that serve as a benchmark dataset for researchers. We are trying to build a classifier on top of ember dataset.

We are using extensively google collab for data processing. After the data processing is done we would switch on to AWS sage Maker to deploy the model. we would create a normal client by sending a simple normal .exe file to check whether it's malware or not.

First Step.

1. Connect your google drive to the collab file
2. Before you start your collab Lab, Please install lief version 0.10.0
3. The dataset is at the following link "https://ember.elastic.co/ember_dataset_2017_2.tar.bz2"
4. After downloading the dataset into your local collab.
5. Extract all the data.
6. After extracting the data, Please save all the data into your google drive.

Why Save data into google drive? It's because as we are using a large dataset, it's common for collab file to run out of the given RAM and lose all the temporary storage. So, we may have to sownload again. Instead of doing this again and again we could simply get all the files and data from drive. It saves you lot of time.

7. After saving data into the drive, to preprocess this humongus data we may need to convert the given data into vectors. To convert ourself manually into vectors would take lot of time for us.
8. We can use the Ember data library to our advantage, where it would convert our data into the vectors we need to process.
9. Once we start our data processing, it's common to lose our data files like train data and test data, In a case like this you may have to execute everything again from the start
10. We can fix this by saving our important data files into hdf5 format.
11. What we are trying to do is a Classification problem, But our dataset has Unclassified data too. Remove the miscellanous data like this.
12. Scale the dataset, so that you don't overload the memory.
13. Build a good model of your choice mentioned accordingingly to the paper.
14. Train the model.
15. Test the model
16. Note the metrics for the model
17. Save the model into your drive in h5 format.
18. Save your model weights into your drive in h5 format.
19. Save your model json in your drive, so that you could use this to your advantage in the next level(deployment)
20. Sample Test a .exe file to check whether it's malign or benign with your model.

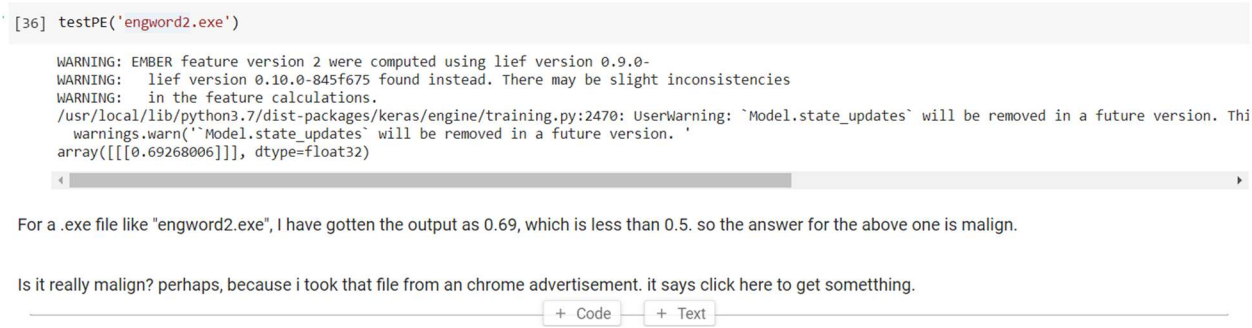
My Takeaways from data preprocessing:

1. Appreciate the ember library.
2. Follow the versions which the ember library was using, If not you will be landing into a serious problem
3. Debugging the code.
4. How to debug a given library.
5. How to handle RAM crashes
6. How to save your data and model state and retrieve them back.
7. How to increase your google collab RAM by running a continous loop.
8. Testing a .exe file
9. How important it is for you test the model beofre you deploy it.

Results:

My model gave the following results

Accuracy : 95% Loss : 20.72



SECOND STEP

The second step in completing your classifier was to deploy your model to AWS SAGEMAKER.

1. Once you are all set with AWS credentials. Open your AWS SageMaker.
2. Your IAM role is so important before you create your sagemaker instance
3. choose your instance as "LAB".
4. without proper IAM role, It would be so hard to cope up with the things you do in aws sagemaker.
5. Create a notebook instance with appropriate conda tensorflow version.
6. After choosing the version, upload your model weights, json, model.h5 files into your sagemaker.
7. Design a malware classifier with the trained model you have.
8. wrap your classifier around this model and weights.
9. Sample test your classifier before you deploy it.
10. Once, you are all set with your model. you should deploy the model.
11. After the deployment of your model, get the service endpoint of your model.
12. you would only communicate with the endpoint, the endpoint would talk with your classifier.
13. please follow the apporpriate security measures while deploying your model, like access your endpoint.

My takeaways from deployment of my model into AWS SAGEMAKER

1. How we could host Machine leanring algorithms as a service.
2. The service would be a game changer, because many of applicaions wer following asynchronous way of getting their data.
3. The Web API Call syntax.
4. IAM role importance
5. SageMaker usage.

Results:

```
%%time
predictor = sagemaker_model.deploy(initial_instance_count=1,
                                   instance_type='ml.t2.medium')
```

```
update_endpoint is a no-op in sagemaker>=2.
See: https://sagemaker.readthedocs.io/en/stable/v2.html for details.
```

```
----------------!CPU times: user 1.47 s, sys: 129 ms, total: 1.59 s
Wall time: 8min 34s
```

```
predictor.endpoint
```

```
The endpoint attribute has been renamed in sagemaker>=2.
See: https://sagemaker.readthedocs.io/en/stable/v2.html for details.
```

```
'sagemaker-tensorflow-serving-2021-10-13-22-34-33-929'
```

```
endpoint_name = 'sagemaker-tensorflow-serving-2021-10-13-22-34-33-929'
```

The enpoint is created successfully.

THIRD STEP

1. The third step in getting my job done was to create a client file for the service I have created just noww.
2. First test the service endpoint in any application like postbox, to get to know about whether service is running or not.
3. Get your AWS CLI credentals
4. Create a client to access the endpoint by using these credentials
5. Use the endpoint and proper content type to get your service from the endpoint
6. Test your client.

My Takeaways from doing this:

1. How important is the api call syntax
2. The importance json in API calls
3. Serilization necessity in API call
4. How to find a service is up and running

RESULTS:

```
[3] !wget https://web.archive.org/web/20070203100217/http://www.caltrox.com/downloads/down1/engword2.exe

    --2021-10-14 13:30:30--  https://web.archive.org/web/20070203100217/http://www.caltrox.com/downloads/down1/engword2.exe
    Resolving web.archive.org (web.archive.org)... 207.241.237.3
    Connecting to web.archive.org (web.archive.org)|207.241.237.3|:443... connected.
    HTTP request sent, awaiting response... 200 OK
    Length: unspecified [application/octet-stream]
    Saving to: 'engword2.exe'

    engword2.exe            [     <=>            ]   2.28M  2.23MB/s    in 1.0s

    2021-10-14 13:30:33 (2.23 MB/s) - 'engword2.exe' saved [2386971]
```

```
[15] !python clientPE.py 'engword2.exe'

     WARNING: EMBER feature version 2 were computed using lief version 0.9.0-
     WARNING:   lief version 0.10.0-845f675 found instead. There may be slight inconsistencies
     WARNING:   in the feature calculations.
     {'predictions': [[0.5]]}
```

Yep, It's Malicious.

Overall Learnings

1. How to handle huge datasets
2. How to deploy a keras model in AWS Sagemaker
3. How to create a client for a service.
4. How to Analyze errors in CLOUDWATCH.

The YouTube url for the video is: https://youtu.be/ikGxkO5bpaI