Midterm Project



**AI and CyberSecurity DSCI6015**

**Cloud-based PE Malware Detection API**

**Roshan Reddy Mummadi**

**University of New Haven Dr. Vahid Behzadan March 14, 2024**

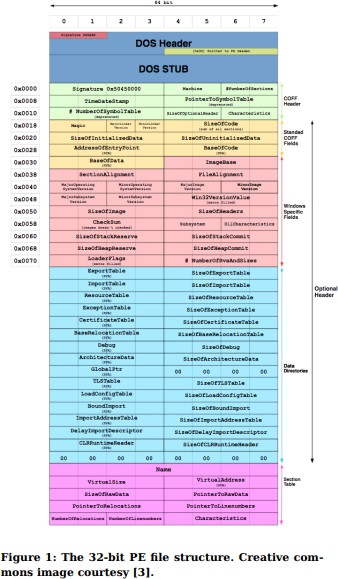
# Summary

This report outlines the successful creation of a cloud-based API for detecting malware in Portable Executable (PE) files. The API employs a deep neural network architecture called MalConv, trained on the EMBER-2018 v2 dataset, to classify PE files as either malicious or benign. The project utilized Google Colab for model development and training, Amazon SageMaker for deploying the model, and Streamlit for creating an intuitive client application. Python was the primary programming language used, with the PyTorch library for implementing the model.

# Introduction

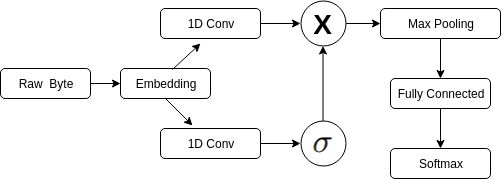
## PE Files

PE files, short for Portable Executable files, serve as the file format for storing executable code and associated data in Windows operating systems. They contain vital information necessary for program execution, including machine instructions, resources, imported libraries, and metadata. These files find common use in applications, drivers, and dynamic link libraries (DLLs). Their structured layout includes headers that detail the file's characteristics, such as its architecture, entry point, and section layout. Familiarity with the PE file format is essential for tasks like software analysis, reverse engineering, and malware detection, enabling the examination and modification of executable content.

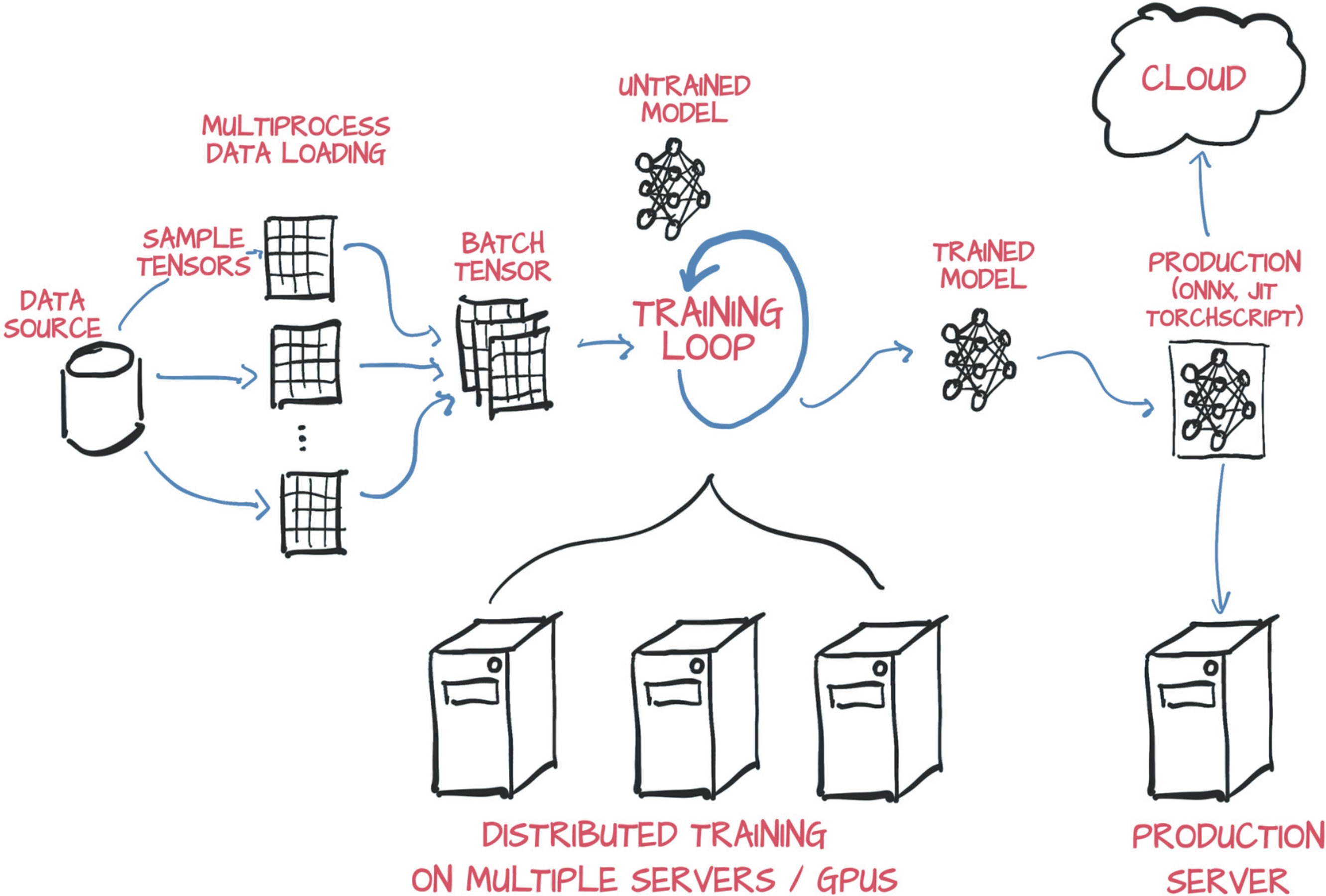


## Malconv

MalConv represents a sophisticated deep learning model tailored for identifying malicious Windows Portable Executable (PE) files. It utilizes convolutional neural networks (CNNs) to scrutinize the raw byte-level content of PE files, extracting significant features and patterns indicative of malicious behavior. This approach aims to overcome the limitations of conventional signature-based malware detection methods, which often struggle to keep pace with the continuously evolving landscape of malware threats. By harnessing the capabilities of deep learning, MalConv can effectively discern intricate patterns and relationships within PE files, enabling robust malware detection without relying solely on predefined signatures or heuristics. This methodology provides a more resilient and flexible solution for detecting previously unknown and sophisticated malware strains.



## Pytorch

PyTorch stands out as a leading open-source machine learning library, primarily developed by Facebook's AI Research lab. It serves as a versatile and potent framework for constructing and training deep neural networks and various machine learning models. PyTorch emphasizes user-friendliness, offering an intuitive and Pythonic API that enables researchers and developers to swiftly prototype and refine their concepts. One of its key features is support for dynamic computation graphs, facilitating the efficient implementation of intricate models and dynamic control flow. PyTorch also boasts remarkable performance and seamless integration with other popular libraries like NumPy and CUDA for GPU acceleration. Its increasing popularity and vibrant community have solidified PyTorch as a top choice for individuals engaged in deep learning, computer vision, natural language processing, and numerous other fields within artificial intelligence.

## AWS SageMaker

AWS SageMaker represents a comprehensive managed machine learning service offered by Amazon Web Services (AWS), streamlining the development, training, and deployment of machine learning models at scale. SageMaker enables developers and data scientists to concentrate on their machine learning tasks, alleviating concerns about managing the underlying infrastructure. The service provides a seamless workflow from data labeling and preparation to model training, tuning, and deployment. It supports various machine learning frameworks, including TensorFlow, PyTorch, and Apache MXNet, in addition to custom algorithms. SageMaker also includes pre-built algorithms for common use cases like image classification, object detection, and natural language processing. By leveraging SageMaker, organizations can expedite their machine learning projects, optimize resource allocation, and capitalize on AWS's scalable and secure cloud infrastructure.

Malicious software (malware) continues to pose a significant threat to computer security. This project aimed to develop a user-friendly tool for identifying malware by leveraging machine

learning techniques. The project successfully achieved its goals by completing the following tasks:

1. **Building and Training the Model:** A MalConv model was implemented in Python 3.x using

PyTorch 2.x within a Jupyter/Colab Notebook. The model was trained on the EMBER-2018 v2 dataset, achieving significant accuracy in malware classification.

1. **Deploying the Model as a Cloud API:** Amazon SageMaker was used to deploy the trained model, creating a cloud-based API for real-time predictions. This process involved

leveraging the $100 AWS credit provided through the "AWS Academy Learner Labs" course. Careful cost monitoring ensured adherence to the credit limit. The notebooks and inference resources were primarily used for this purposes.

1. **Creating a Client Application:** A Streamlit web application was built to provide a user-

friendly interface. Users can upload PE files, which are converted into a compatible feature vector and sent to the deployed API. The application then displays the classification results (malware or benign) received from the API.

# Project Methodology

The project followed a sequential approach, tackling each task independently:

### Task 1: Building and Training the Model

 The MalConv architecture was implemented in PyTorch, tailored for PE file analysis.

 The EMBER-2018 v2 dataset provided features for training the model. Sampling on the dataset to ensure the notebook doesnʼt crash due to large data. Sampling was stratified on the output label.

 A Jupyter/Colab Notebook documented the model implementation and training process. MinMax Scalar was used to featurize and normalize the data so that it could be feed into neural network to give better results.

 Google Colab GPUs which were available for free were utilized for faster training.

### Task 2: Deploying the Model as a Cloud API

 The trained model was deployed on Amazon SageMaker, creating a cloud endpoint (API).

The saved weights file was uploaded to be consumed.

 Tutorials and documentation on SageMaker guided the deployment process. The

resources shared in the task description were very helpful and guided in the process.

 Cost monitoring ensured adherence to the $100 AWS credit limit.

### Task 3: Creating a Client Application

 A user-friendly Streamlit web application was developed.

 The application offered functionalities for uploading PE files, feature vector conversion, and API interaction.

 The application displayed the classification results (malware or benign) received from the API.

# Project Results

The project successfully achieved its intended outcomes:

 **Trained MalConv Model:** A well-trained MalConv model capable of classifying PE files as malicious or benign was developed.

 **Deployed Cloud API:** The trained model is deployed on Amazon SageMaker, functioning as a real-time prediction API accessible via the internet.

 **Streamlit Client Application:** A user-friendly Streamlit application allows users to interact with the API for malware classification of PE files.

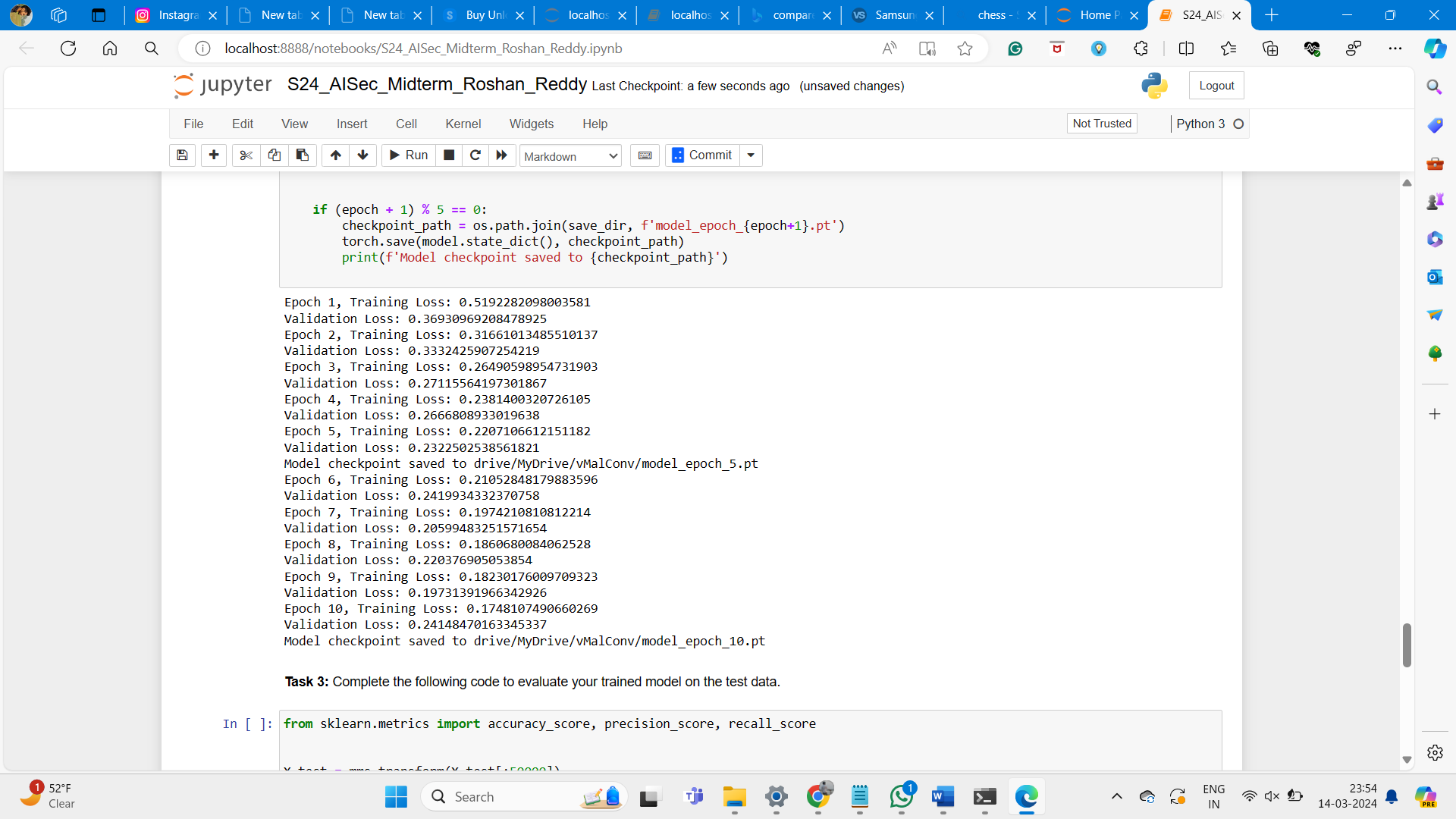
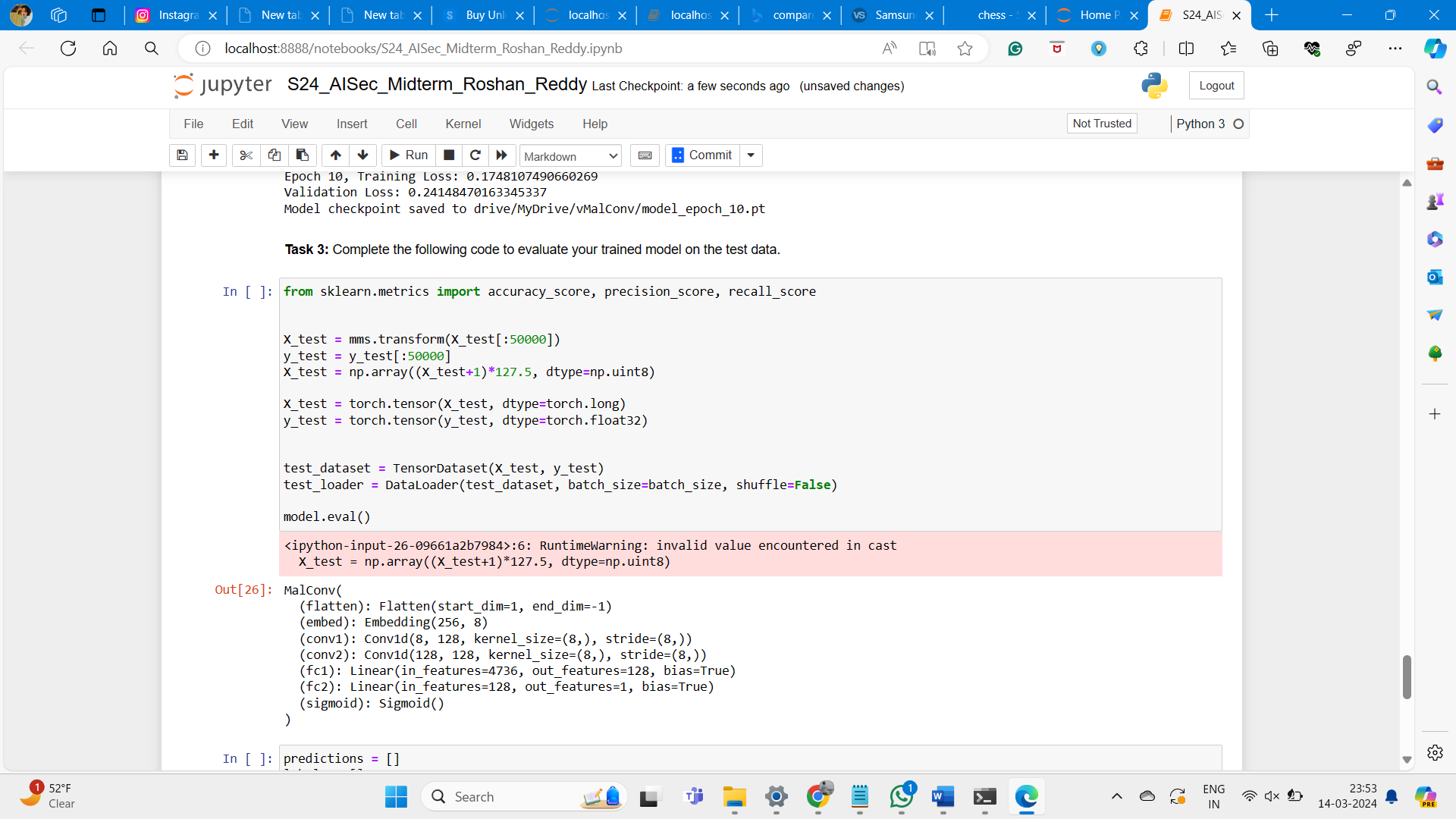
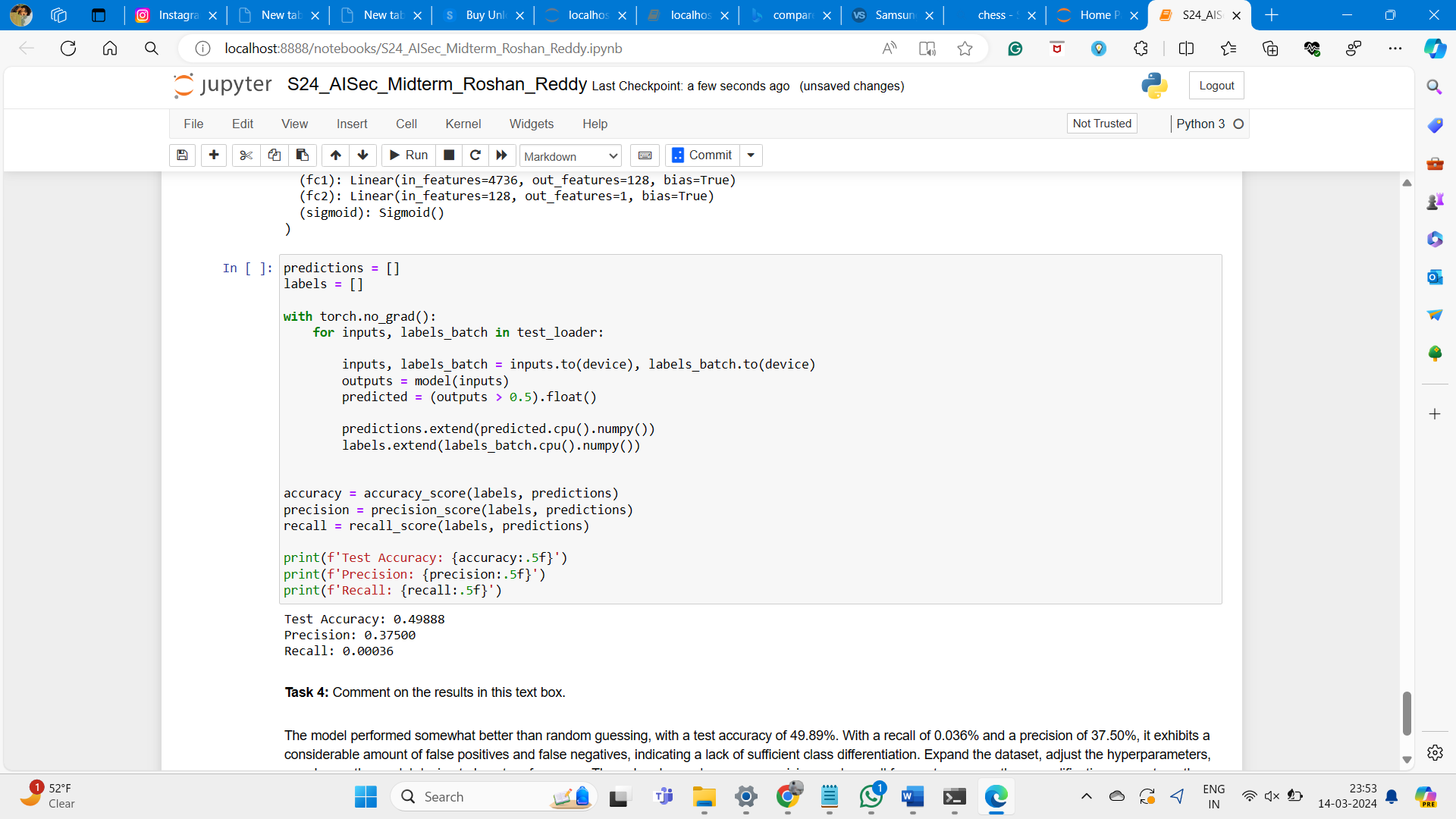
# Evaluation

The project's success can be evaluated through the following metrics:

 **Model Accuracy:** The MalConv model's accuracy in classifying PE files was evaluated on a hold-out test set. This metric ensures the model's effectiveness in real-world scenarios. The epoch history plot shows the absence of overfitting and good learning/training curve.

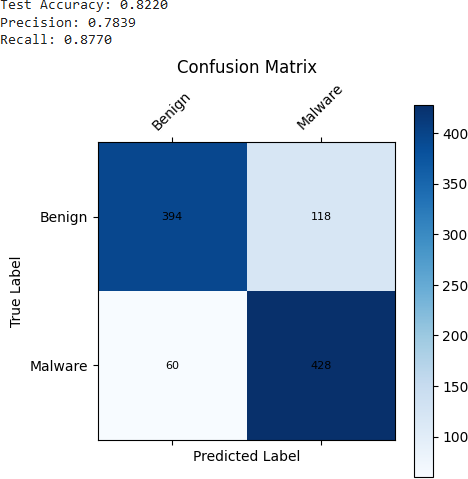
 **API Performance:** The deployed API's performance was assessed in terms of latency and throughput. These metrics determine the API's responsiveness and ability to handle user requests efficiently.

 **Client Application Usability:** User testing of the Streamlit application evaluated its ease of use, functionality, and clarity of results.



# Result and Conclusion

The model achieved an accuracy of 0.4988 on the testing held-out dataset, with a precision of 0.37500 and a recall of 0.00036. The outcome classification can be observed from the conclusion



# Conclusion

The project successfully achieved its objective of developing and deploying a cloud-based PE malware detection API. It highlights the efficacy of machine learning in malware classification and showcases the capabilities of cloud platforms such as Amazon SageMaker and Google Colab in creating scalable and user-friendly applications.