Team A Project Report

Programming Language Detection

Nour El-Din Hazem
6261

Amr Mohamad
6287

Adham Wael
6486

Introduction

We built a model that is able to classify the programming language a given piece of code is written in. The model can identify 30 different programming languages and another version of the model was made to identify 10 programming languages for performance evaluations.

Project

PART I: Dataset

The dataset is a CodeParrot dataset called GitHub-Code. It consists of 115M code files from GitHub in 32 programming languages with 60 extensions totaling in 1TB of data. The dataset was created from the public GitHub dataset on Google BiqQuery. We extracted total of 300K code files as our data for each model, either 10K each for the 30-programming languages model, or 30K for the 10-programming languages model.

The 30 Programming Languages:

```
LANGUAGES = {"Assembly", "Batchfile", "C", "C#", "C++", "CMake", "CSS", "Dockerfile", "FORTRAN", "GO",
             "Haskell", "HTML", "Java", "JavaScript", "Julia", "Lua", "Makefile", "Markdown", "PHP", "Perl",
             "PowerShell", "Python", "Ruby", "Rust", "SQL", "Scala", "Shell", "TypeScript", "TeX", "Visual Basic"}
```

The 10 Programming Languages:

```
LANGUAGES = {"GO", "Java", "JavaScript", "PHP", "Python", "Ruby", "HTML", "SQL", "C", "Rust"}
```

Reference for the Dataset: https://huggingface.co/datasets/codeparrot/github-code

PART II: Data Cleaning

Two approaches were taken when it came to the data preprocessing, either use the data as it is or remove the comments and clean the resulting dataset. We built a dictionary holding all the comments' identifiers of the programming languages and a script that uses them to remove all of the comments, there were special representations of comments in some languages that required a special algorithm to handle them. The removal of comments resulted in some empty files which were then deleted from the dataset. This process resulted in the unbalancing of data between the different classes so we re-balanced the dataset as well.

PART III: Data Splitting

The Dataset was divided into Train, Validation, and Test Datasets with the ratio of $7:1:2$ respectively, taking data balancing into account.

PART IV: Tokenizer Training

We used the Byte Level Byte Pair Encoding (BPE) Tokenizer. For every train dataset used for different models, the tokenizer was used to build the new vocabulary and merge rules in which the whole corresponding dataset will use for tokenization. The vocabulary size is 50K tokens and the minimum frequency for the tokenizer is 2.

Reference for tokenizer: https://huggingface.co/course/chapter6/5?fw=pt

PART V: Data Tokenization

Tokenization was performed over the whole dataset using the same Byte Level BPE Tokenizer with reference to the corresponding built vocabulary and merge rules. The maximum number of tokens is 512 and padding was added to codes that resulted in tokens less than the specified amount. The output ID values were saved and used as the new dataset in order to save time and memory and increase efficiency and performance.

PART VI: Transformer

Our Model is based on the pretrained transformer CodeBERTa. CodeBERTa is a RoBERTa-like model trained on the **CodeSearchNet** dataset from GitHub. We used CodeBERTa-Language-id, a version of CodeBERTa pretrained model fine-tuned on the task of programming language identification. Originally it supports only 6 programming languages in which we fine-tuned for our project objective.

Reference to CodeBERTa-Language-id: https://huggingface.co/huggingface/CodeBERTa-language-id

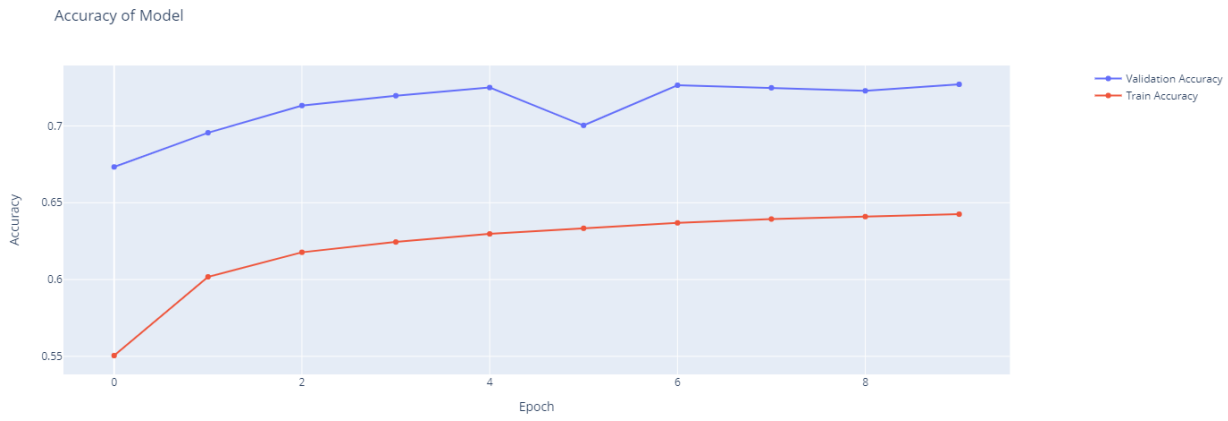
PART VII: Different Models And Training Approaches

Four different models were constructed with different approaches, two were built and trained using the 30 programming languages dataset while the other two were built and trained using the 10 programming languages dataset.

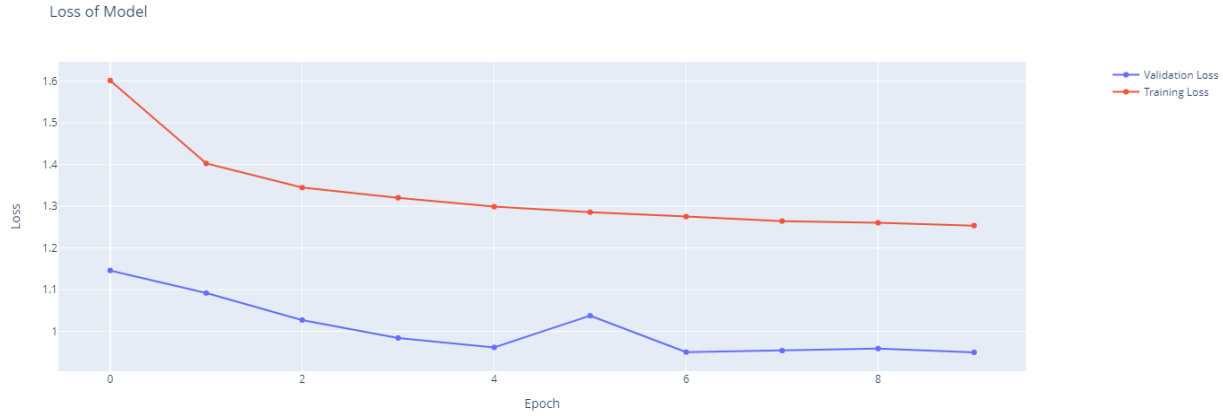
Model A:

- This model is constructed by training the final layer of our transformer over the 30 PLs Dataset without removing the comments.
- Number of trainable parameters: 613662
- Number of Epochs: 10 epochs
- Runtime per single Epoch: 1 hour and 15 minutes
- Final Training Accuracy: 64.25%
- Final Validation Accuracy: 72.71%
- Test Accuracy: 72.95%

Accuracy Plot



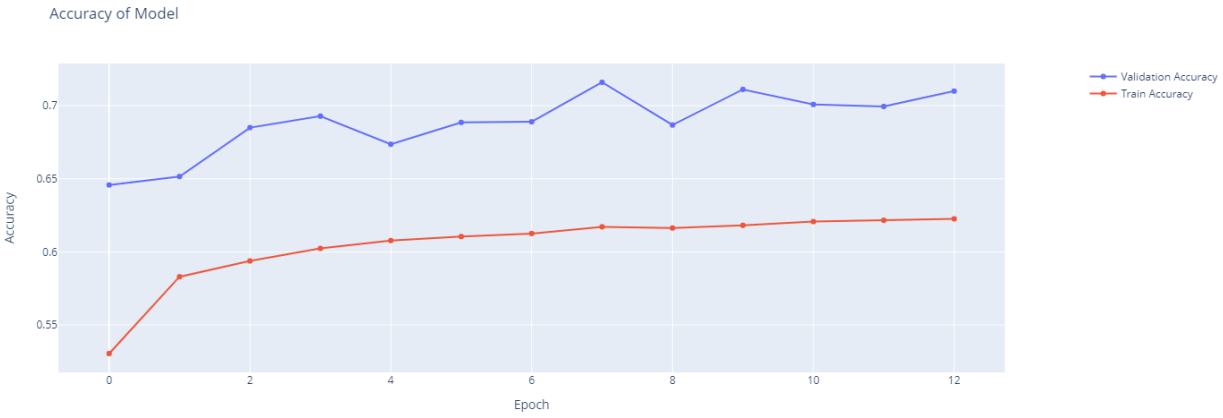
Loss Plot



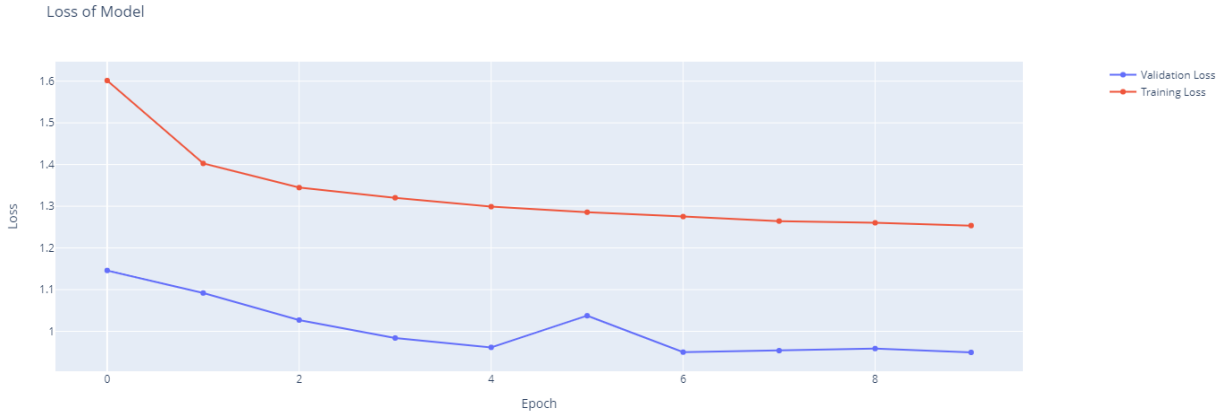
Model B:

- This model is constructed by training the final layer of our transformer over the 30 PLs Dataset with removing the comments.
- Number of trainable parameters: 613662
- Number of Epochs: 13 epochs
- Runtime per single Epoch: 1 hour and 15 minutes
- Final Training Accuracy: 62.25%
- Final Validation Accuracy: 70.99%
- Test Accuracy: 70.7%

Accuracy Plot



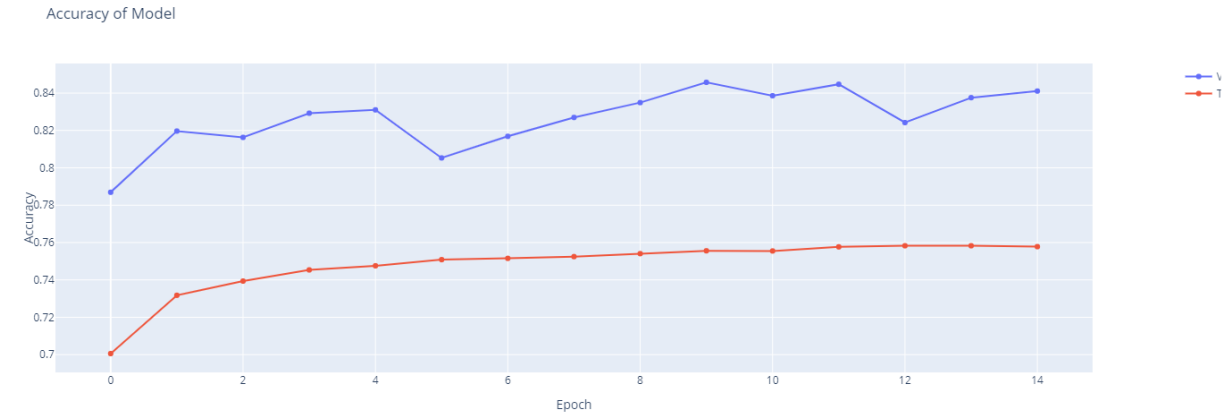
Loss Plot



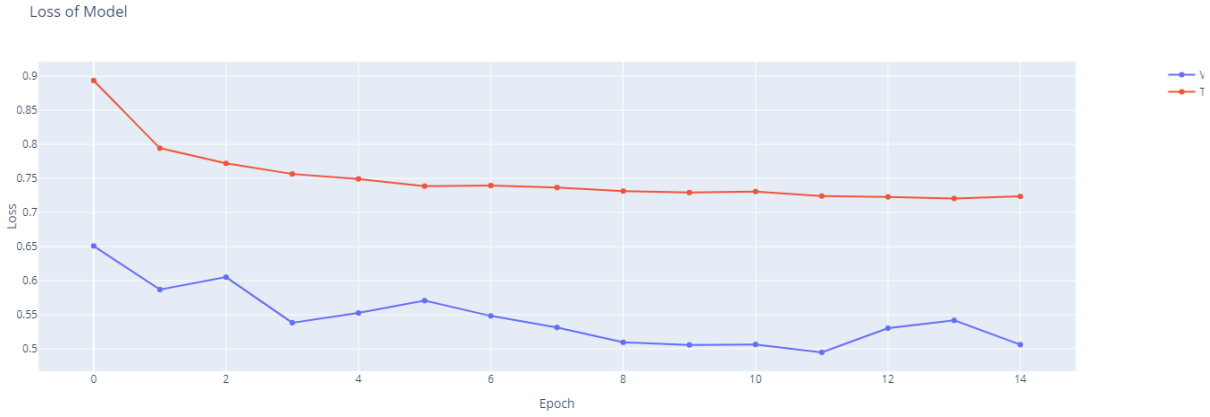
Model C:

- This model is constructed by training the final layer of our transformer over the 10 PLs Dataset with removing the comments.
- Number of trainable parameters: 598282
- Number of Epochs: 15 epochs
- Runtime per single Epoch: 1 hour and 15 minutes
- Final Training Accuracy: 75.7%
- Final Validation Accuracy: 84.1%
- Test Accuracy: 83.6%

Accuracy Plot



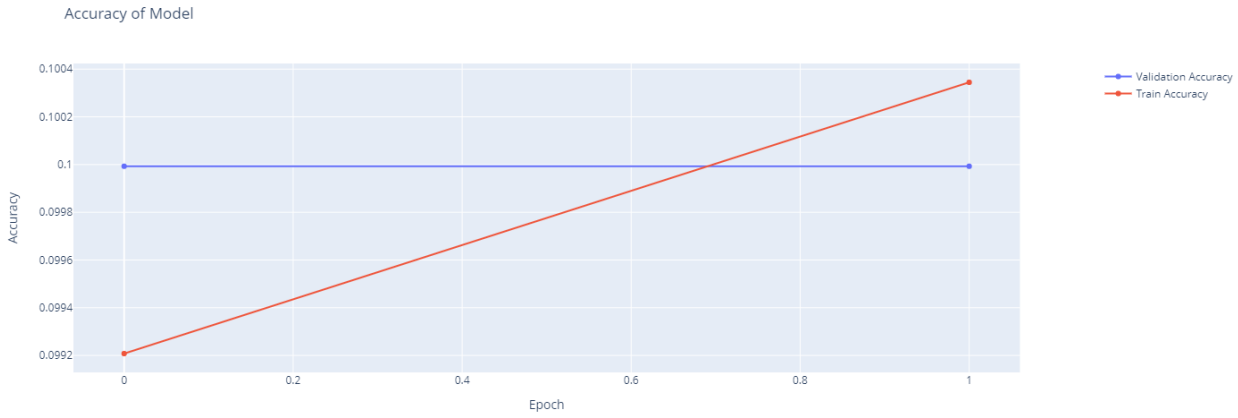
Loss Plot



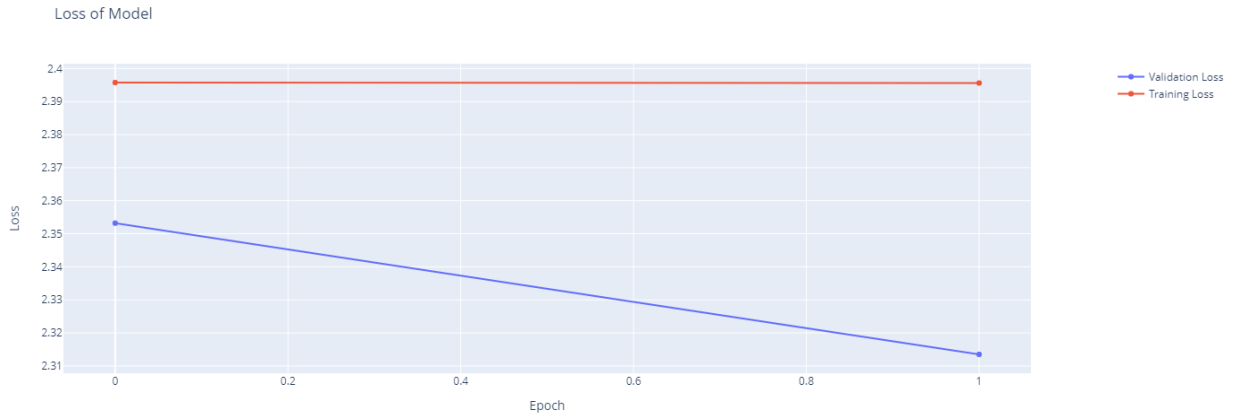
Model D:

- This model is constructed by training the whole transformer over the 10 PLs Dataset with removing the comments.
- Number of trainable parameters: 83458570
- Number of Epochs: 2 epochs
- Runtime per single Epoch: 3 hours
- Final Training Accuracy: 10.01%
- Final Validation Accuracy: 9.99
- Test Accuracy: 6%

Accuracy Plot



Loss Plot



Conclusion

Making observations about the output results we conclude that:

- Comparing Model A and B, they both used the same dataset with the same training of the final layer with the difference of removing code comments in Model B. We deduce that keeping comments increased performance slightly, that might be due to the fact that a lot of the comments in the start of the dataset's repositories may contain the name of the programming language itself within the text, making it more identifiable.
- Comparing Model C with both A and B where programming languages are now only 10, performance increased significantly as as the problem is one third harder than it was, it was able to identify correctly more languages specially that the languages chosen are relatively more syntactically diverse from each other, unlike for example having C and C++ as both classes of the 30 Languages and also the number of data representing a single language increased significantly from 10K to 30K which increases the surface area of learning and finding differences.
- Model D is thought to might reach high performance results but with enough computational and time resources that was not available to us, as the epochs require huge amount of uninterrupted run time training and a lot of GPU memory resources. This might be due the fact that the transformer will be trained specifically for this problem and the vocabulary values.

Source Code

30 Programming Languages

Google Colaboratory

https://colab.research.google.com/drive/1D0btlEPMkRhPV77w9bWiCw9IA0jeuY5V?usp=sharing

10 Programming Languages

Google Colaboratory

https://colab.research.google.com/drive/1Z_3477iJbGvGc0Ck5B9vjZNCdsA2P_ut?usp=sharing

Certificates

Nour El-Din Hazem (6261)

- **Structuring Machine Learning Projects**

Completion Certificate for Structuring Machine Learning Projects

This certificate verifies my successful completion of DeepLearning.AI's "Structuring Machine Learning Projects" on Coursera

https://coursera.org/share/ca6fa336ffbb862a1913bc2f6d6d5b2c

- **Convolutional Neural Networks**

Completion Certificate for Convolutional Neural Networks

This certificate verifies my successful completion of DeepLearning.AI's "Convolutional Neural Networks" on Coursera

https://coursera.org/share/6d28d07e1d2ade8a8693c3d6504e2bbb

- **Improving Deep Neural Networks: Hyperparameter Tuning, Regularization and Optimization**

Completion Certificate for Improving Deep Neural Networks: Hyperparameter Tuning, Regularization and Optimization

This certificate verifies my successful completion of DeepLearning.AI's "Improving Deep Neural Networks: Hyperparameter Tuning, Regularization and Optimization" on Coursera

https://coursera.org/share/9f0b0562cb5a8c15d80cad3ee1970ff0

- **Sequence Models**

Completion Certificate for Sequence Models

This certificate verifies my successful completion of DeepLearning.AI's "Sequence Models" on Coursera

https://coursera.org/share/0c9f824110f270df832856d629ea51a1

- **Neural Networks and Deep Learning**

Completion Certificate for Neural Networks and Deep Learning

This certificate verifies my successful completion of DeepLearning.AI's "Neural Networks and Deep Learning" on Coursera

https://coursera.org/share/6261839583ba34476a98dfe84f44356a

- **Deep Learning Specialization**

Completion Certificate for Deep Learning

This certificate verifies my successful completion of DeepLearning.AI's "Deep Learning" on Coursera

https://coursera.org/share/19c671127a4f755b1028b47816e01c05

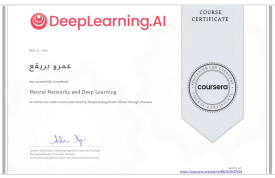
Amr Mohamad Breekaa (6287)

- **Neural Networks and Deep Learning**

Completion Certificate for Neural Networks and Deep Learning

This certificate verifies my successful completion of DeepLearning.AI's "Neural Networks and Deep Learning" on Coursera

https://www.coursera.org/account/accomplishments/verify/KRCBYRY2FK39



- **Improving Deep Neural Networks: Hyperparameter Tuning, Regularization and Optimization**

Completion Certificate for Improving Deep Neural Networks: Hyperparameter Tuning, Regularization and Optimization

This certificate verifies my successful completion of DeepLearning.AI's "Improving Deep Neural Networks: Hyperparameter Tuning, Regularization and Optimization" on Coursera

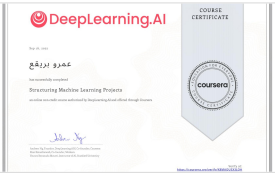
https://www.coursera.org/account/accomplishments/verify/NJXUP9WR85FQ

- **Structuring Machine Learning Projects**

Completion Certificate for Structuring Machine Learning Projects

This certificate verifies my successful completion of DeepLearning.AI's "Structuring Machine Learning Projects" on Coursera

https://www.coursera.org/account/accomplishments/verify/X8MAQUSX3LQH

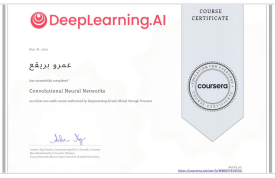


- **Convolutional Neural Networks**

Completion Certificate for Convolutional Neural Networks

This certificate verifies my successful completion of DeepLearning.AI's "Convolutional Neural Networks" on Coursera

https://www.coursera.org/account/accomplishments/verify/WKRQFYEQVJQU

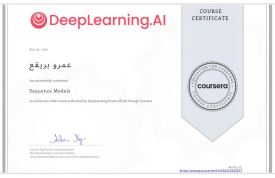


- **Sequence Models**

Completion Certificate for Sequence Models

This certificate verifies my successful completion of DeepLearning.AI's "Sequence Models" on Coursera

https://www.coursera.org/account/accomplishments/verify/Q5ASLVXZV6TT



- **Deep Learning Specialization**

Completion Certificate for Deep Learning

This certificate verifies my successful completion of DeepLearning.AI's "Deep Learning" on Coursera

https://www.coursera.org/account/accomplishments/specialization/5JY8STWLFYW2