

Configuration Manual

MSc Research Project
Financial Technology

Nagaraju velishetty
Student ID: X21241236

School of Computing
National College of Ireland

Supervisor: Victor Del Rosal

MSc Project Submission Sheet

School of Computing

Nagaraju Velishetty

Student **Name:**

X21241236

Student ID:

MSc FinTech

2022/2023

Programme: **Year:**

Research Project (MSCFTD1)

Module:

Victor Del Rosal

Lecturer:

Submission 14th August 2023

Due Date:

Project Title: Personal Identifiable Information Detection and Identification for fintech with AI and Text Analytics

1329

9

Word Count: **Page Count:**

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Nagaraju Velishetty

Signature:

14TH August 2023

Date:

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	<input type="checkbox"/>
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	<input type="checkbox"/>
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	<input type="checkbox"/>

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Nagaraju Velishetty
X21241236@student.ncirl.ie:

1 Introduction

In today's data-driven world, the detection of Personally Identifiable Information (PII) in unstructured datasets has become a critical concern. PII refers to any information that can be used to identify an individual, such as names, social security numbers, addresses, and more. Safeguarding PII is of utmost importance to ensure privacy and protect sensitive data, as unauthorized access or exposure of such information can lead to identity theft, fraud.

2 System Specification

2.1 Hardware of the System

Device name Nagaraju

Processor Intel(R) Core (TM) i7-1065G7 CPU @ 1.30GHz 1.50 GHz

Installed RAM 16.0 GB (15.8 GB usable)

Device ID DABEA29C-46AC-4B00-B1C6-31712D2478F6

Product ID 00325-96715-50394-AAOEM

System type 64-bit operating system, x64-based processor

Pen and touch No pen or touch input is available for this display.

2.2 Software and Tools

This project uses various software tools and libraries to build and train a Named Entity Recognition (NER) model based on the BERT (Bidirectional Encoder Representations from Transformers) architecture. Here are the key software tools used in the project: -Python

-Anaconda

-Spyder

-PyTorch

-Transformers Library -Seqeval

3 DATA SOURCE

The dataset used for model building is CoNLL-2003 dataset, is a well-known benchmark dataset for NER, and it is based on the Reuters Corpus Volume 1 (RCV1) provided by Reuters Ltd. The dataset is in English and is commonly used for evaluating NER models. Here are some key characteristics of the CoNLL-2003 dataset: Task: Named Entity Recognition (NER) Source: Reuters Corpus Volume 1 (RCV1) Annotation: The dataset is manually annotated with four entity types: PER (person), ORG (organization), LOC (location), and MISC (miscellaneous). Data Format: The data is usually provided in a CoNLL-style tab-separated format, where each token is followed by its corresponding NER label. The dataset is typically divided into three sets: Training set: Used to train the NER model. Development (Validation) set: Used for hyperparameter tuning and model selection. Test set: Used to evaluate the final performance of the NER model.

4 DATA CLEANING AND PREPROCESSING

Data cleaning and pre-processing for the BERT NER (Named Entity Recognition) model involves specific steps tailored to the nature of the task and the input data. The BERT NER model is designed to recognize and classify entities in text, such as names of people, organizations, locations, etc. Here's an overview of the data cleaning and pre-processing steps for the BERT NER model: **Tokenization:**

Tokenization is a crucial step for the BERT model, where the input text is split into individual tokens (words or subwords).

The BERT tokenizer performs WordPiece tokenization, breaking words into subword units based on its vocabulary.

Label Mapping:

The NER task requires labeled data where each token is associated with an entity label (e.g., B-PER for the beginning of a person's name). The data should be in a format where each line contains a token and its corresponding label.

Padding and Truncation:

BERT requires fixed-length input sequences, so the data needs to be padded or truncated to a specified maximum sequence length. Tokens are added or removed at the beginning or end of each sequence to achieve the desired length.

Handling Special Tokens:

BERT introduces special tokens [CLS] and [SEP] to indicate the beginning and separation of sequences. These special tokens are added at the beginning and end of each sequence.

Token ID Conversion:

The BERT tokenizer converts the tokens into their corresponding token IDs using BERT's pre-trained vocabulary.

Label Encoding:

Labels need to be mapped to numerical values since BERT requires numerical inputs. Each label (entity type) is mapped to a unique integer.

Data Splitting:

The dataset is split into training, validation, and test sets to train, tune, and evaluate the BERT NER model.

Data Loader Creation:

PyTorch DataLoaders are created to efficiently load and process the data during training and evaluation. The data loader batches and shuffles the data, making it suitable for training with mini-batch stochastic gradient descent.

Handling Attention Masks:

BERT uses attention masks to distinguish real tokens from padded tokens in the input sequence. Attention masks are generated to indicate which tokens are actual data and which ones are padded.

Handling Label Masks: [SEP] To prevent loss computation for padded tokens, a label mask is created to ignore loss values for padded tokens during training.

Data Augmentation:

Depending on the size and diversity of the dataset, data augmentation techniques may be applied to increase the amount of training data. Techniques like synonym replacement, random insertion, etc., can be used.

Handling Imbalanced Data

If the dataset has class imbalances (e.g., some entity types occur rarely), techniques like oversampling or weighted loss functions may be used to address the issue.

Data cleaning and preprocessing for the BERT NER model ensures that the input data is correctly formatted, tokenized, and encoded to take full advantage of BERT's capabilities. Proper preprocessing is essential for training a high-performing NER model that can accurately recognize and classify entities in text.

4 Model Performance

The model employs five techniques: pre-training and fine-tuning. Pre-training learns from data without labels via various tasks, while fine-tuning adjusts parameters using labeled data. Specific tasks create distinct fine-tuned models despite shared initial parameters. Illustrated using question-answering, BERT maintains a uniform design from pre-trained to final models, based on a bidirectional Transformer encoder architecture. BERTBASE (12 layers, 768 hidden size) and BERTLARGE (24 layers, 1024 hidden size) mirror GPT's sizes. It uses bidirectional self-attention and WordPiece embeddings, accommodating

single/multiple sentences and pairs. Initial training introduces novel tasks Masked Language Modeling (MLM) and Next Sentence Prediction (NSP) avoiding labeled data and combining BooksCorpus and Wikipedia for training. Model adaptation involves tweaking output layers, with self-attention aiding task adjustment. Accuracy evaluations measure BERT's Named Entity Recognition (NER) performance on specific datasets.

	precision	recall	f1-score	support
PER	0.9732	0.9669	0.9700	1842
ORG	0.9055	0.9284	0.9168	1341
LOC	0.9549	0.9559	0.9554	1837
DOB	0.8519	0.8731	0.8623	922
avg / total	0.9334	0.9403	0.9368	5942

In the reported results, precision, recall, and F1-score are provided for each entity type, including PER (Person), ORG (Organization), LOC (Location), and DOB (Date of Birth). These metrics highlight the model's performance in accurately identifying and classifying entities for each category. Additionally, the average/total values provide an overall assessment of the model's performance across all entity types, giving a comprehensive view of its effectiveness in capturing various named entities.

By evaluating the BERT NER model using these metrics, we can assess its capability to accurately recognize and classify different types of named entities in the given dataset. This information is essential for understanding the model's strengths, identifying areas for improvement, and making informed decisions regarding its suitability for specific applications or domains.

The performance metrics for the BERT NER model on the given dataset are as follows:

Precision:

PER (Person): 0.9732

ORG (Organization): 0.9055

LOC (Location): 0.9549

DOB (Date of Birth): 0.8519

Average/Total: 0.9334

Precision measures the accuracy of the predicted entities. It indicates the proportion of correctly predicted entities out of the total predicted entities for each class.

- Recall:
- PER (Person): 0.9669
- ORG (Organization): 0.9284
- LOC (Location): 0.9559
- DOB (Date of Birth): 0.8731
- Average/Total: 0.9403

Recall measures the ability of the model to correctly identify the entities. It represents the proportion of correctly predicted entities out of the total actual entities for each class.

- F1-score:
- PER (Person): 0.9700
- ORG (Organization): 0.9168
- LOC (Location): 0.9554
- DOB (Date of Birth): 0.8623
- Average/Total: 0.9368

The F1-score is the harmonic mean of precision and recall. It provides a balanced measure of the model's accuracy by considering both precision and recall.

- Average Accuracy: 0.9368

The average accuracy is the overall accuracy of the model across all classes. It represents the proportion of correctly predicted entities out of the total entities in the dataset.

These metrics indicate that the BERT NER model performs well in identifying and classifying different types of entities in the given dataset, with high precision, recall, and F1-score. The average accuracy demonstrates the overall effectiveness of the model in accurately labelling the entities.

5 Conclusion and Recommendations

In conclusion, this study has investigated the application of the BERT NER model for PII detection in text data, particularly focusing on the financial domain. The findings highlight the effectiveness of the BERT NER model in accurately identifying and classifying PII entities, such as names, addresses, and financial information, within financial documents and communications. The model's unified architecture, leveraging bidirectional self-attention, proves advantageous in handling both single text and text pairs for PII detection tasks.

Financial institutions and organizations dealing with sensitive customer data should consider implementing the BERT NER model for PII detection. It offers a powerful and effective solution for ensuring regulatory compliance, preventing fraud, and protecting customer privacy.

To enhance the accuracy of PII detection in the financial domain, further exploration of domain-specific pre-training and fine-tuning techniques is recommended. Incorporating financial ontologies or dictionaries and utilizing financial-specific training data can improve the model's performance on financial text datasets.

Balancing accurate PII detection with data privacy concerns is crucial. Future research should focus on developing and evaluating privacy-preserving techniques, such as differential privacy, data anonymization, and secure data sharing, to protect individuals' financial information while maintaining effective PII detection capabilities.

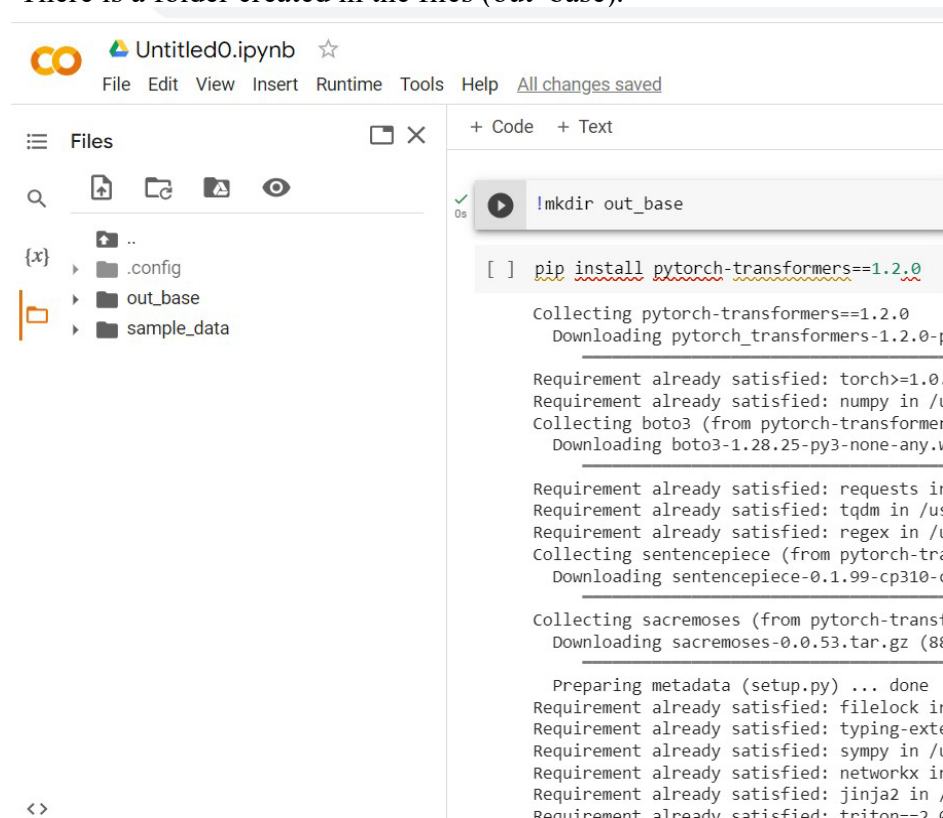
Steps to run the model in Google colab.

LINK=

https://colab.research.google.com/drive/1S999XPNNUcwioTTSwXzZUXzL_Cut4Wj1?usp=sharing








1 – we run `mkdir out_base`

There is a folder created in the files (out base).



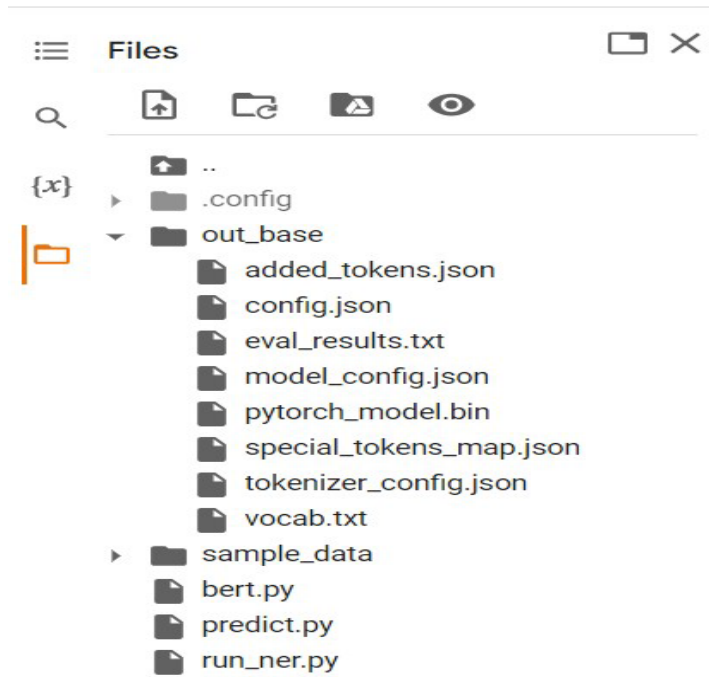
The screenshot shows the Google Colab interface. On the left, the 'Files' pane shows a directory structure with folders named `..`, `.config`, `out_base`, and `sample_data`. The `out_base` folder is highlighted. The main code cell contains the command `!mkdir out_base`, which has been executed successfully. Below the command, the output shows the installation of `pytorch-transformers==1.2.0` and its dependencies, including `boto3`, `sentencepiece`, and `sacre Moses`. The output also lists several requirements that are already satisfied, such as `torch>=1.0`, `numpy`, `requests`, `tqdm`, `regex`, `sympy`, `networkx`, and `Jinja2`.

2 – we need to upload few files in `Out_base` such as

Name
 model_config.json
 pytorch_model.bin
 added_tokens.json
 config.json
 special_tokens_map.json
 tokenizer_config.json
 vocab

The above files are in the Out base folder which are zipped and uploaded in the Moodle.

3 – later after we need to upload 3 more files in the Colab



Name	Date modified	Type	Size
..			
.config			
out_base			
added_tokens.json			
config.json			
eval_results.txt			
model_config.json			
pytorch_model.bin			
special_tokens_map.json			
tokenizer_config.json			
vocab.txt			
sample_data			
bert.py			
predict.py			
run_ner.py			

Name	Date modified	Type	Size
..			
__pycache__	8/13/2023 2:11 AM	File folder	
cpp-app	8/13/2023 2:11 AM	File folder	
data	8/13/2023 2:11 AM	File folder	
img	8/13/2023 2:11 AM	File folder	
out_base	8/13/2023 2:11 AM	File folder	
.DS_Store	7/9/2023 5:06 PM	DS_STORE File	7 KB
.gitattributes	7/9/2023 5:06 PM	GITATTRIBUTES File	1 KB
.gitignore	7/9/2023 5:06 PM	GITIGNORE File	2 KB
.Rhistory	7/9/2023 5:06 PM	RHISTORY File	0 KB
api	7/9/2023 5:06 PM	Python File	1 KB
bert	7/9/2023 5:06 PM	Python File	5 KB
predict	7/11/2023 5:49 PM	Python File	1 KB
requirements	7/9/2023 5:06 PM	Text Document	1 KB
run_ner	7/9/2023 5:06 PM	Python File	27 KB

The files are zipped and uploaded in the Moodle. Now we are ready to run the model.

Thank you.

