Documentation

• The chosen model and rationale behind the selection:

My chosen model is BERT, which was proposed in the paper BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding by Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova.

BERT is a powerful model for text categorization, and it can be fine-tuned for our resume classification task by adding a classification layer on top of the pre-trained model and training it on the resume dataset. BERT was designed to pre-train deep bidirectional representations from unlabeled text by conditioning on both the left and right context simultaneously across all layers. As a result, BERT can understand the text of the resume better than other single-direction language models. Furthermore, because BERT develops a contextual understanding of the text, it can better comprehend the contextual meaning of different portions of people's resumes based on the relationship of words with their surrounding words. In addition, because BERT can capture long-term dependencies, it is better suited for categorizing resumes' large texts spanning from 500 to 1000 words. Lastly, I experimented with using several machine learning models (Random Forrest, K-Nearest Neighbors, and One-vs-Rest classifiers) before choosing BERT. I found that the machine learning model's results were at most 65% accurate, whereas I later obtained almost 80% accuracy with BERT.

• Preprocessing or feature extraction methods employed:

To process the resumes to convert them into a format suitable for training, I used pretrained BERT Tokenizer and encoded the text data. *No manual feature extraction of preprocessing is required for this.*

Example of tokenization applied on a sentence.

In [37]:
```
from transformers import BertTokenizer

tokenizer = BertTokenizer.from_pretrained('bert-base-uncased', do_lower_case=True);

encoded_text = tokenizer.encode_plus(
    "A quick brown fox jumps",
    add_special_tokens=True,
    return_attention_mask=True,
    padding='max_length',
    truncation=True,
    max_length=256,
    return_tensors='pt'
)

input_ids = encoded_text['input_ids'].to(device)
attention_mask = encoded_text['attention_mask'].to(device)

input_ids[0][:10], attention_mask[0][:10]
```

Out[37]:
```
(tensor([  101,  1037,  4248,  2829,  4419, 14523,   102,     0,     0,     0],
        device='cuda:0'),
 tensor([1, 1, 1, 1, 1, 1, 1, 0, 0, 0], device='cuda:0'))
```

Behind the scenes, the BERT tokenizer employs a technique called subword-based tokenization. Subword-tokenization breaks down unfamiliar words into smaller words or characters, allowing the model to deduce meaning from the tokens. For example, the word 'jumps' is broken down into 'jump' and 's'. The vocabulary is generated by BERT using the wordpiece algorithm, which constructs subwords based on the probability of letters appearing together.

• Instructions on how to run the script and expected outputs:

Ideally, the scripts should be run locally on a computer with the required python packages installed.

The script.py imports the following packages:

```
import sys
import os
import shutil
import pandas as pd
from PyPDF2 import PdfReader
import torch
from transformers import BertTokenizer
from transformers import BertForSequenceClassification
from transformers import logging
```

I recommend creating a new anaconda environment and installing the following packages with conda.

Perform the following operation one by one:

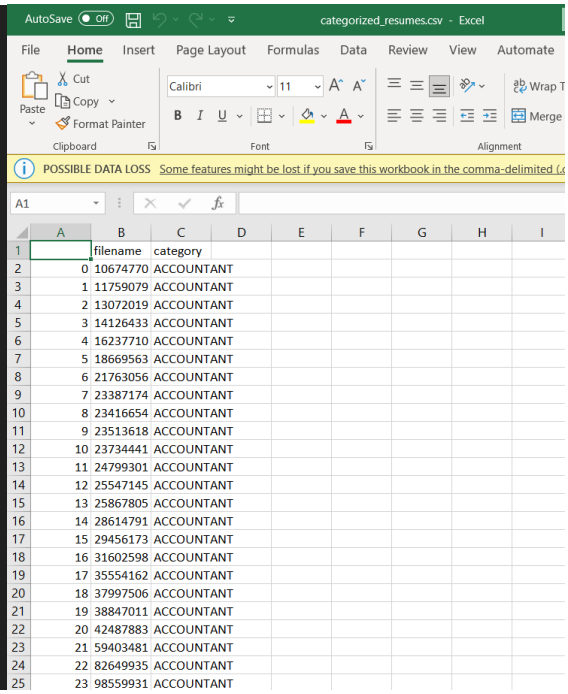
- Download and Install “Anaconda” from https://www.anaconda.com/download/
- Open “Anaconda Prompt (anaconda3)” from start menu.
- Run the follow commands one by one, and press ‘y’ when prompted.
  - ➔ conda create --name myenv
  - ➔ conda activate myenv
  - ➔ conda install pytorch torchvision torchaudio pytorch-cuda=11.7 -c pytorch -c nvidia (**or** chose your pc configuration from https://pytorch.org/ and run the command given there)
  - ➔ conda install -c anaconda pandas
  - ➔ conda install -c conda-forge pypdf2
  - ➔ conda install -c conda-forge transformers
- Download this github repository and copy the file location.
- Open “Anaconda Prompt (anaconda3)” from start menu again and run the following commands.
  - ➔ cd “downloaded folder location that you copied”. (replace “” with location path)
  - ➔ python script.py ./resume-dataset/data/test-data/ (**or** python script.py “your resume folder location”)

Expected outputs:

1. Respective category folder predicted by model name “resume-categories-predicted”
2. “categorized_resumes.csv” containing filename and predicted category.

Name	Date modified	Type	Size
.ipynb_checkpoints	8/13/23 1:06 PM	File folder	
resume-categories-predicted	8/13/23 7:49 PM	File folder	
resume-dataset	8/12/23 9:04 PM	File folder	
categorized_resumes.csv	8/13/23 7:50 PM	Microsoft Excel Co...	12 KB
exploration_preprocessing_training.ipynb	8/13/23 10:46 PM	Jupyter Source File	339 KB
finetuned_BERT.model	8/13/23 7:26 PM	MODEL File	427,823 KB
script.py	8/13/23 7:47 PM	Python File	4 KB

ACCOUNTANT	8/13/23 7:47 PM	File folder
ADVOCATE	8/13/23 7:50 PM	File folder
AGRICULTURE	8/13/23 7:47 PM	File folder
APPAREL	8/13/23 7:49 PM	File folder
ARTS	8/13/23 7:49 PM	File folder
AVIATION	8/13/23 7:49 PM	File folder
BANKING	8/13/23 7:49 PM	File folder
BUSINESS-DEVELOPMENT	8/13/23 7:48 PM	File folder
CHEF	8/13/23 7:50 PM	File folder
CONSTRUCTION	8/13/23 7:48 PM	File folder
CONSULTANT	8/13/23 7:48 PM	File folder
DESIGNER	8/13/23 7:48 PM	File folder
DIGITAL-MEDIA	8/13/23 7:50 PM	File folder
ENGINEERING	8/13/23 7:49 PM	File folder
FINANCE	8/13/23 7:49 PM	File folder
FITNESS	8/13/23 7:49 PM	File folder
HEALTHCARE	8/13/23 7:49 PM	File folder
HR	8/13/23 7:49 PM	File folder
INFORMATION-TECHNOLOGY	8/13/23 7:49 PM	File folder
PUBLIC-RELATIONS	8/13/23 7:50 PM	File folder
SALES	8/13/23 7:50 PM	File folder
TEACHER	8/13/23 7:50 PM	File folder



	filename	category
0	10674770	ACCOUNTANT
1	11759079	ACCOUNTANT
2	13072019	ACCOUNTANT
3	14126433	ACCOUNTANT
4	16237710	ACCOUNTANT
5	18669563	ACCOUNTANT
6	21763056	ACCOUNTANT
7	23387174	ACCOUNTANT
8	23416654	ACCOUNTANT
9	23513618	ACCOUNTANT
10	23734441	ACCOUNTANT
11	24799301	ACCOUNTANT
12	25547145	ACCOUNTANT
13	25867805	ACCOUNTANT
14	28614791	ACCOUNTANT
15	29456173	ACCOUNTANT
16	31602598	ACCOUNTANT
17	35554162	ACCOUNTANT
18	37997506	ACCOUNTANT
19	38847011	ACCOUNTANT
20	42487883	ACCOUNTANT
21	59403481	ACCOUNTANT
22	82649935	ACCOUNTANT
23	98559931	ACCOUNTANT