GROUP K

June, 2023

TOXIC COMMENT DETECTOR

THINK TWICE
BEFORE MAKE A COMMENT

財金二 柯宥圻
歷史四 陳彤樺
經濟四 楊永晨
土木碩一 林承平

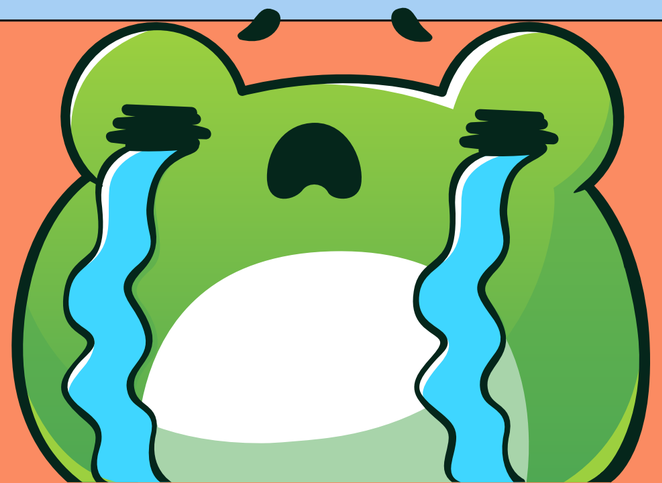
BUBBLE TEA?

Bubble tea project

WHY WE CHANGE OUR TOPIC??

1. Data quality issues
(lack of representativeness, and data relevance + validity)
2. We wanted to do something with more machine learning related to language

goodbye!



AGENDA ★

- Purpose of our project
- Package and Tools
- Data Collection
- ML Model & Outcome
- Conclusion

OUR PURPOSE

Toxic Comment

DEFINITION

A rude, disrespectful, or unreasonable comment that makes it difficult to engage in rational discussion.

OUR GOAL

We want to build an unique model, which is a toxic comment detector, to make people think twice before making a comment, <u>particularly for some NTU students.</u>

RAW DATA

Toxic Comment

Toxic Comment Classification Challenge

Identify and classify toxic online comments

Jigsaw/Conversation AI · 4,539 teams · 5 years ago

Overview Data Code Discussion Leaderboard Rules Team Su

Dataset Description

ou are provided with a large number of Wikipedia comments which have been labeled by hur
aters for toxic behavior. The types of toxicity are:

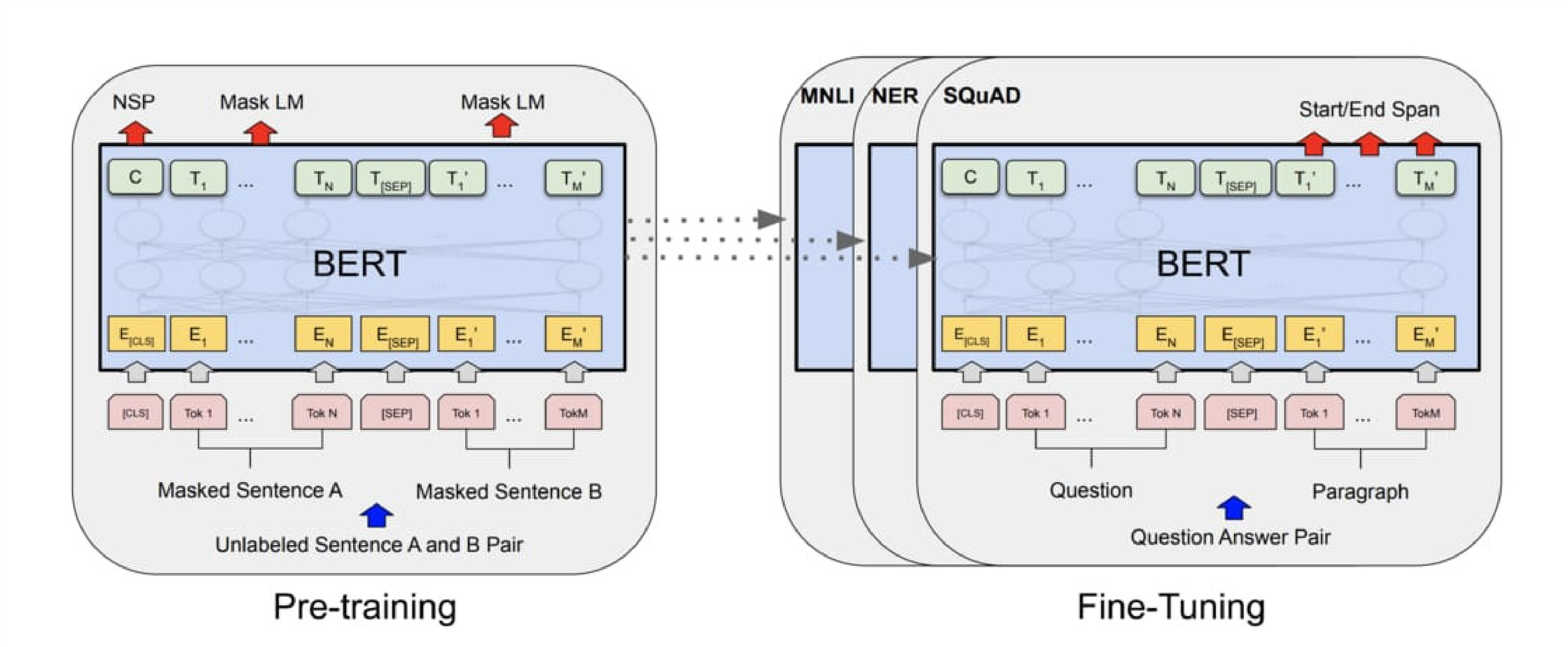
- `toxic`
- `severe_toxic`
- `obscene`
- `threat`
- `insult`
- `identity_hate`

We find data on Kaggle
It is quite messy and we clean it by deleting missing values and unnecessary symbols.

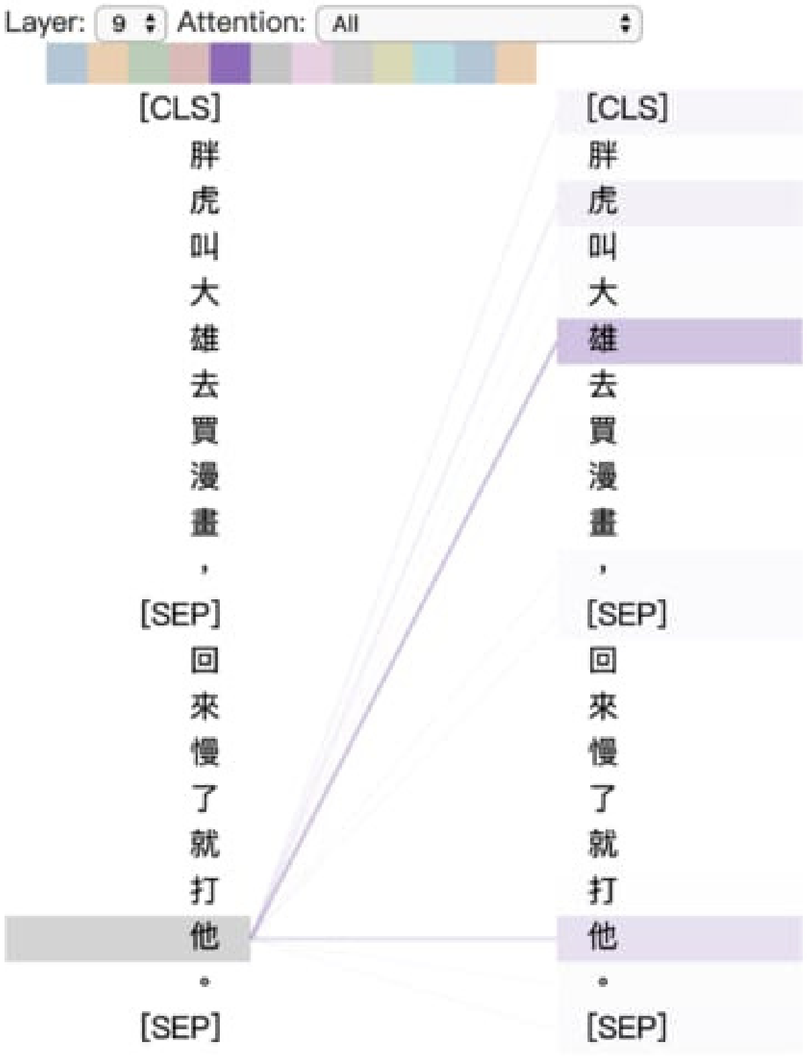
PACKAGE & TOOLS

Back to Agenda Page

Powerful BERT can read and understand our language!



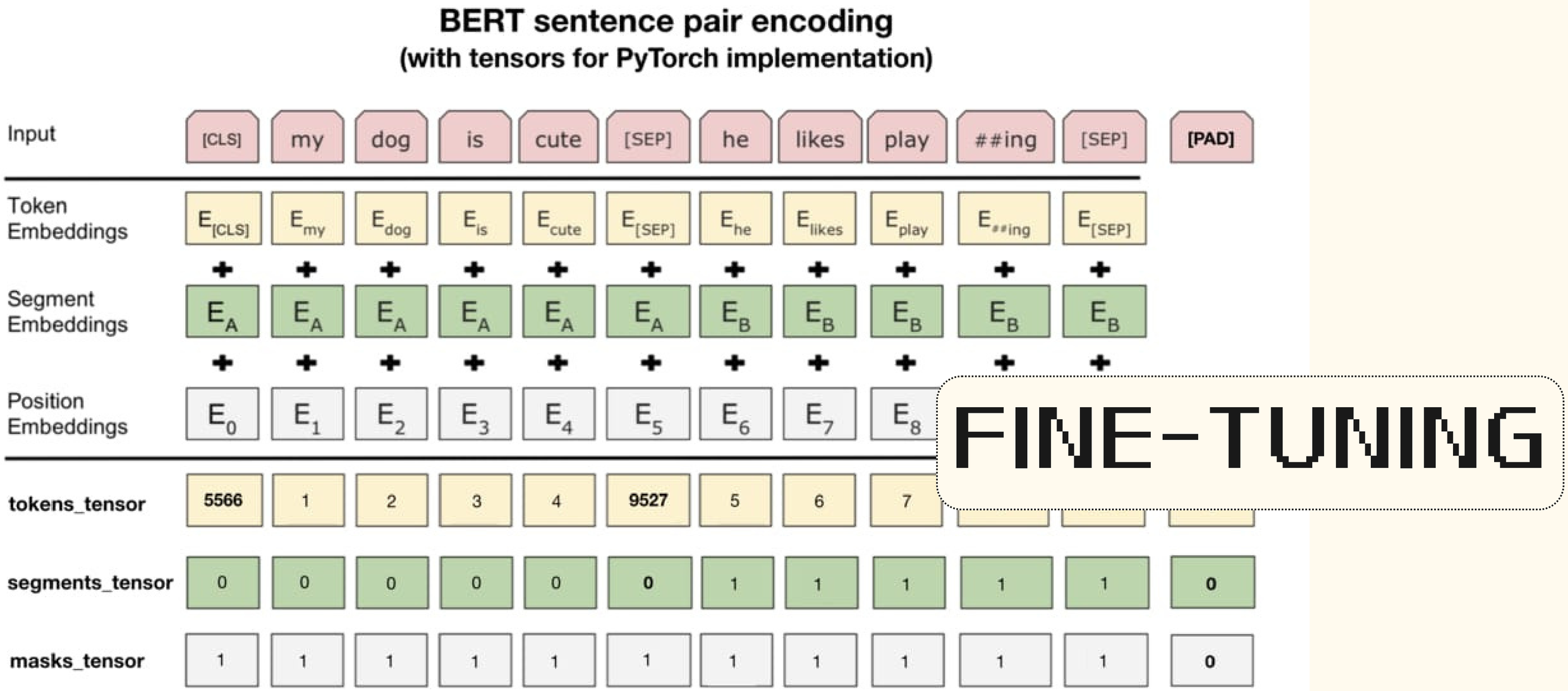
依照不同情境，BERT 在更新「他」的 repr. 時關注的上下文相異



情境 1：「他」指的是大雄

SELF-ATTENTION

情境 2：「他」指的是胖虎



Tokenizer can transform sentence into corresponding several vectors, by transformer and our training data, model learn what causes toxic.

FLASK

Ps. We download the model in .h5 file, so we don't need to train the model whenever used.

```
from flask import Flask, render_template, request
import numpy as np
import tensorflow as tf
from transformers import BertTokenizerFast, TFBertForSequenceClassificatior
app = Flask(__name__)

PRETRAINED_MODEL = 'bert-base-uncased'
new_tokenizer = BertTokenizerFast.from_pretrained(PRETRAINED_MODEL)
new_model = TFBertForSequenceClassification.from_pretrained(PRETRAINED_MODE
new_model.load_weights("../model_weights.h5")

@app.route('/', methods=['GET', 'POST'])
def home():
    if request.method == 'POST':
        blackmail_newinput = request.form['text_input']
        # process the input user give
        return render_template('index.html', prediction="""predicted value'
    return render_template('index.html')

if __name__ == '__main__':
    app.run(debug=True)
```

When running the app.py, the flask framework will start and user can put any sentence, and the model will calculate the result.

HTML

```html
<head>
    <title>Potentially Toxic Comment Detection</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css
        integrity="sha384-9ndCyUaIbzAi2FUVXJi0CjmCapSmO7SnpJef0486qhLnuZ2cdeRhO02iuK6FU
    <link rel=stylesheet type="text/css" href="style.css">
</head>
<body class="container p-2">
    <div class="pt-3">
        <div class="content">
            <h1>Potentially Toxic Comment Detection</h1>
            <form method="POST" action="/">
                <div class="form-group">
                    <label for="text_input" class="form-label">Enter a comment:</label>
                    <textarea class="form-control" id="text_input" name="text_input" ty
                </div>
                <div class="form-group">
                    <button type="submit" class="btn btn-primary" value="Detect">Detect
                </div>
            </form>
```

CSS

```css
@import url('https://fonts.googleapis.com/cs
.container{
    width: 100vw;
    height: 100vh;
    display: flex;
    justify-content: center;
    align-items: start;
    background-color: #ECF8F9;
    padding: 0;
    margin: 0;
    font-family: 'Roboto', sans-serif;
}
.content{
    width: 100%;
    height: 100%;
    padding: 0.1rem;
    margin: 0.1rem;
}
.card{
    width: 100% !important;
}
```

FINAL PRESENTATION

Toxic Comment

SHOW TIME!

VIDEO

UPLOAD

WEB CRAWLER FT.REDDIT

Toxic Comment

```python
df = pd.DataFrame()  # initialize dataframe

# loop through 50 pages of trending posts in /r/AmITheAsshole
for page in range(1):
    # make a request for the trending posts in /r/AmITheAsshole
    url = f"https://oauth.reddit.com/r/AmITheAsshole/hot?page={page+1}"
    res = requests.get(url, headers=headers)

    postList = res.json()['data']['children']
```

This gets the posts that we needed in the subreddit!

WEB CRAWLER FT.REDDIT

Toxic Comment

```python
    # loop through each post retrieved from GET request
    for post in postList:
        new_row = pd.DataFrame({
            'subreddit': ['AmITheAsshole'],
            'title': [post['data']['title']],
            'selftext': [post['data']['selftext']],
            'upvote_ratio': [post['data']['upvote_ratio']],
            'ups': [post['data']['ups']],
            'downs': [post['data']['downs']],
            'score': [post['data']['score']]
        })
        df = pd.concat([df, new_row], ignore_index=True)

        postId = post['data']['id']
        sort = "old"
        threaded = "false"
        res = requests.get(
            f"https://oauth.reddit.com/comments/{postId}?sort={sort}&threaded={threaded}", headers=headers)
        commentList = res.json()[1]['data']['children']
```

We get the details for each post and the list of comments.

WEB CRAWLER FT.REDDIT

Toxic Comment

```
        # loop through each comment retrieved from GET request
        for comment in commentList:
            # append relevant data to dataframe
            new_comment_row = pd.DataFrame({
                'comment_author': [comment['data'].get('author', None)],
                'comment_body': [comment['data'].get('body', None)]
            })
            df = pd.concat([df, new_comment_row], ignore_index=True)

# save dataframe to Excel
df.to_excel(os.path.join(os.getcwd(), 'reddit_comments_AmITheAsshole.xlsx'), index=False)
```

We save the comments to the excel file so that our machine learning model can analyze it.

THE NEXT STEP

Toxic Comment

Connecting with ChatGPT and other API to enhance and add new usage.	Create a google chrome or LINE extension

HOW IT HELPS?

Toxic Comment

1. **Promoting a Safe Online Environment:** By automatically identifying and flagging toxic comments, the model can help maintain a respectful and safe environment for students.
2. **Enhancing Online Learning Experiences:** As universities increasingly adopt online learning platforms, the toxic comment detector can be integrated into discussion forums to identify.
3. **Proactive Intervention and Early Warning System**: By monitoring online platforms and social media channels, the toxic comment detector can act as an early warning system

TEAM SPLIT

Toxic Comment

柯宥圻	BERT MODEL CONSTRUCTION, FRONT-END
楊永晨	PREPROCESSING, LITERATURE REVIEW
林承平	FRONT-END, WEB-APP RELATED WORKS
陳彤樺	DATA EVALUATION, WEB CRAWLER, SLIDES

Thanks!
(Q&A)

Reference

https://huggingface.co/docs/transformers/
https://huggingface.co/models
https://huggingface.co/docs/transformers/main/model_doc/bert#tfbertmodel
https://www.oreilly.com/library/view/getting-started-with/9781838821593/2f41b723-d4d7-4bd9-a7e9-82ecb3d76bb4.xhtml#uuid-a03529e7-7b9c-4770-bb02-9cb564ac3e68
https://medium.com/%E4%BA%BA%E5%B7%A5%E6%99%BA%E6%85%A7-%E5%80%92%E5%BA%95%E6%9C%89%E5%A4%9A%E6%99%BA%E6%85%A7/epoch-batch-size-iteration-learning-rate-b62bf6334c49
https://ithelp.ithome.com.tw/articles/10241789