Toxicity Analysis of Social Media Texts
Using Deep-Learning Models

By

Syed M Anas Imran - K21-3169
Huzaifa Siddique - K21-3376
Zehra Fatima - K21-4658

DLP-Sec-B

Submitted to
Ms. Sumaiyah Zahid
For the course of Deep Learning For Perception
FAST- National University Of Computer & Emerging Sciences

Objective

The objective of this project is to build a robust deep learning model that can accurately classify hate speech and toxic content in textual data. The model combines traditional LSTM layers with modern transformer-based architectures (via Hugging Face) to improve classification performance and predict the intensity of hateful content.

Problem Statement

The increasing prevalence of hate speech and toxic content on digital platforms poses a serious challenge to content moderation. This project addresses the issue by developing a model that not only classifies text as hateful or non-hateful but also quantifies the degree of hate intensity.

Methodology

Data Cleaning:

Dataset**:** hate-speech-detection-curated-dataset

Performed preprocessing including noise removal, text normalization, and tokenization. Separated data into training and testing sets to validate model generalization.

Models:

LSTM and Transformers are used which are trained on Dataset from Kaggle. distilBERT Transformer from HuggingFace is used which is formed by distillation learning from the larger model BERT. further fine tuned on the dataset from Kaggle

Model Architecture:

Developed a hybrid model using LSTM layers for sequence processing and Hugging Face Transformers for semantic representation. Compiled and trained the model using appropriate loss functions and optimizers.

Evaluation:

Assessed model performance using test data, and predicted both classification labels and hate intensity scores for custom inputs.

Results

- The hybrid model combining LSTM and Transformers achieved high accuracy in identifying hate speech.

- The model could also predict hate intensity scores, adding a nuanced view to classification.

- Visualization techniques such as histograms and confusion matrices highlighted prediction effectiveness across categories.

TRAINING

```python
history = model.fit(
    train_dataset,
    validation_data=test_dataset,
    epochs=5
)
```

```
Epoch 1/5
2/2 [==============================] - 183s 48s/step - loss: 0.7150 - accuracy: 0.3875 - val_loss: 0.6363 - val_accuracy: 0.6750
Epoch 2/5
2/2 [==============================] - 145s 40s/step - loss: 0.8860 - accuracy: 0.3938 - val_loss: 0.7106 - val_accuracy: 0.6750
Epoch 3/5
2/2 [==============================] - 142s 40s/step - loss: 0.6983 - accuracy: 0.4125 - val_loss: 0.6980 - val_accuracy: 0.6750
Epoch 4/5
2/2 [==============================] - 142s 39s/step - loss: 0.6932 - accuracy: 0.4187 - val_loss: 0.6885 - val_accuracy: 0.6750
Epoch 5/5
2/2 [==============================] - 142s 39s/step - loss: 0.6985 - accuracy: 0.4187 - val_loss: 0.6994 - val_accuracy: 0.6750
```

PREDICTIONS

PREDICTIONS

```python
y_pred_logits = model.predict(test_dataset).logits
y_pred_probs = tf.sigmoid(y_pred_logits).numpy().squeeze()

# Convert probabilities to labels (threshold > 0.5)
y_pred_labels = (y_pred_probs > 0.5).astype(int)
```

```
2/2 [==============================] - 15s 3s/step
```

```python
import random

# Test a sentence
for i in range (20):
    num = random.randint(1, len(X_test_texts))
    print(X_test_texts[num],f"Prob: {y_pred_probs[num]:.4f} -> Label: {y_pred_labels[num]}")
```

```
protection expire three months end time left wikipedia alone created additional accounts problems maybe someone open hearing say since one going unblock merely wasting time Prob: 0.5946 -> Label: 1
note click line might need clean obvious intent line Prob: 0.5946 -> Label: 1
censorship article constantly censored zionist hawks supposed even handed analysis phony propagandists like jay jg espouse bullshit views contrary views international community via un human rights groups etc etc ru
love https www linkedin com pub wayne weber got air force went straight head hk wow Prob: 0.5946 -> Label: 1
love https www linkedin com pub wayne weber got air force went straight head hk wow Prob: 0.5946 -> Label: 1
znali itk dobro doctor squad exertion jude priporuci za license bill biam roditelji galvani cell pivot Prob: 0.5946 -> Label: 1
afd nomination zambian wolpertinger nominated zambian wolpertinger article created deletion feel article satisfies wikipedia criteria inclusion explained wikipediaarticles deletion zambian wolpertinger opinions mat
sucks ass Prob: 0.5945 -> Label: 1
worthwhile keeping old images display article well new images utc Prob: 0.5946 -> Label: 1
newsletter draft ready hi dank prepared initial draft drive wrap newsletter wikipediawikiproject guild copy could find time check goes would super thanks Prob: 0.5946 -> Label: 1
censorship article constantly censored zionist hawks supposed even handed analysis phony propagandists like jay jg espouse bullshit views contrary views international community via un human rights groups etc etc ru
newsletter draft ready hi dank prepared initial draft drive wrap newsletter wikipediawikiproject guild copy could find time check goes would super thanks Prob: 0.5946 -> Label: 1
man put false words mouth finish whatever edit conflicted read whole thing first Prob: 0.5946 -> Label: 1
quite welcome glad could help Prob: 0.5946 -> Label: 1
protection expire three months end time left wikipedia alone created additional accounts problems maybe someone open hearing say since one going unblock merely wasting time Prob: 0.5946 -> Label: 1
women simply whores detest race Prob: 0.5945 -> Label: 1
cannot believe calls favorable treatment eu migrants brexit Prob: 0.5946 -> Label: 1
mls outrage listen conference xbox next gen fifa article screwed fix information technology operating theater logging away Prob: 0.5946 -> Label: 1
tom lazovi jedi get nice rumors djevojcice Prob: 0.5946 -> Label: 1
man put false words mouth finish whatever edit conflicted read whole thing first Prob: 0.5946 -> Label: 1
```

GETTING TOXICITY SCORES

```python
from transformers import AutoTokenizer, TFAutoModelForSequenceClassification
import tensorflow as tf
import numpy as np

# Load tokenizer and model
model_name = "unitary/toxic-bert"
tokenizer = AutoTokenizer.from_pretrained(model_name)
model = TFAutoModelForSequenceClassification.from_pretrained(model_name)
```

Python

Could not render content for 'application/vnd.jupyter.widget-view+json'
{"version_major":2,"version_minor":0,"model_id":"0ec45d750b484bfcb56f76ddf4224add"}

Could not render content for 'application/vnd.jupyter.widget-view+json'
{"version_major":2,"version_minor":0,"model_id":"52fb301b9cfc4e5dab4df5b1f8813e6f"}

Could not render content for 'application/vnd.jupyter.widget-view+json'
{"version_major":2,"version_minor":0,"model_id":"9e6206a3a24c4cf2a17d6b2ef8fd2d2f"}

Could not render content for 'application/vnd.jupyter.widget-view+json'
{"version_major":2,"version_minor":0,"model_id":"36f84f515b584063bf5a1a7d6868430b"}

Could not render content for 'application/vnd.jupyter.widget-view+json'
{"version_major":2,"version_minor":0,"model_id":"5c7936adac5046c2b0d53f78993c0a79"}

```
Some weights of the PyTorch model were not used when initializing the TF 2.0 model TFBertForSequenceClassification: ['bert.embeddings.position_ids']
- This IS expected if you are initializing TFBertForSequenceClassification from a PyTorch model trained on another task or with another architecture (e.g. initializing a TFBertForSequenceClassification model from a
- This IS NOT expected if you are initializing TFBertForSequenceClassification from a PyTorch model that you expect to be exactly identical (e.g. initializing a TFBertForSequenceClassification model from a BertForS
All the weights of TFBertForSequenceClassification were initialized from the PyTorch model.
If your task is similar to the task the model of the checkpoint was trained on, you can already use TFBertForSequenceClassification for predictions without further training.
```

```python
import random

# Test a sentence
for i in range (20):
    num = random.randint(1, len(X_test_texts))
    scores = get_toxicity(X_test_texts[num])
    print("\n",X_test_texts[num])
    formatted_scores = {k: round(float(v), 2) for k, v in scores.items()}
    for label, score in formatted_scores.items():
        print(f"{label.title()}: {score}%")


# print("Toxicity Levels (%):", scores)
```

[44]

...

```
 wrong way clear
Toxic: 0.09%
Severe_Toxic: 0.01%
Obscene: 0.02%
Threat: 0.01%
Insult: 0.02%
Identity_Hate: 0.01%

 general point view comments
Toxic: 0.06%
Severe_Toxic: 0.01%
Obscene: 0.02%
Threat: 0.01%
Insult: 0.02%
Identity_Hate: 0.01%

 women simply whores detest race
Toxic: 97.0%
Severe_Toxic: 5.5%
Obscene: 37.72%
Threat: 1.0%
Insult: 42.36%
Identity_Hate: 65.76%
...
Obscene: 0.02%
Threat: 0.01%
Insult: 0.02%
Identity_Hate: 0.01%
```

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

```python
    indices = np.random.choice(len(X_test), 20, replace=False)
    for idx in indices:
        sentence = original_test_texts[idx]
        true_label = "HATE" if y_test[idx] == 1 else "NON-HATE"
        prob = y_pred_probs[idx]
        pred_label = "HATE" if prob > 0.5 else "NON-HATE"
        intensity = round(prob * 100, 2) if pred_label == "HATE" else round((1 - prob) * 100, 2)

        print(f"\nSentence: {sentence}")
        print(f"True Label: {true_label}")
        print(f"Predicted Label: {pred_label}")
        print(f"{'Hate' if pred_label == 'HATE' else 'Non-Hate'} Intensity: {intensity}%")
```

[12] Python

```
4539/4539 ━━━━━━━━━━━━━━━━━━━━ 223s 49ms/step - accuracy: 0.8639 - loss: 0.3405

Test Loss: 0.33951452374458313
Test Accuracy: 0.8644921183586121
4539/4539 ━━━━━━━━━━━━━━━━━━━━ 215s 47ms/step

Accuracy Score (sklearn): 0.8644920949705283

--- Evaluation on 20 Test Sentences ---

Sentence: stop vandalizing please stop your disruptive editing of removing rights to the unreliable and biased sources when you continue to vandalize wikipedia as you did at list of countries by population united n
True Label: HATE
Predicted Label: HATE
Hate Intensity: 92.54000091552734%

Sentence: what a hypocritical award you have i discover that you have a b wizard in christianity on your user pageboy but you called the holy scriptures folktale rather than the inspired information technology have
True Label: HATE
Predicted Label: HATE
Hate Intensity: 83.51000213623047%

Sentence: an ii domain binds to a effort radiate dome
True Label: HATE
Predicted Label: HATE
Hate Intensity: 74.75%
...
Sentence: if this be really cosmos you enter my yard i bequeath use my hunter rifle blow prohibited you channelize but any are in wiki so i will flag you as vandal
True Label: NON-HATE
Predicted Label: NON-HATE
Non-Hate Intensity: 100.0%
```

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

predict label and hate percentage for custom sentences

```python
# Function to predict label and hate percentage for custom sentences
def predict_custom_sentences(sentences):
    cleaned = [clean_text(sentence) for sentence in sentences]
    seqs = tokenizer.texts_to_sequences(cleaned)
    padded = pad_sequences(seqs, maxlen=max_sequence_length)
    probs = model.predict(padded).squeeze()

    for i, sentence in enumerate(sentences):
        prob = probs[i]
        label = "HATE" if prob > 0.5 else "NON-HATE"
        intensity = round(prob * 100, 2) if label == "HATE" else round((1 - prob) * 100, 2)
        print(f"\nSentence: {sentence}")
        print(f"Predicted Label: {label}")
        print(f"{'Hate' if label == 'HATE' else 'Non-Hate'} Intensity: {intensity}%")

# Example predictions on 4 custom sentences
custom_sentences = [
    "I absolutely despise those people!",
    "I hope you have a wonderful day!",
    "You're so stupid and annoying.",
    "This is a peaceful and loving community."
]

predict_custom_sentences(custom_sentences)
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 79ms/step

Sentence: I absolutely despise those people!
Predicted Label: HATE
Hate Intensity: 59.52000045776367%

Sentence: I hope you have a wonderful day!
Predicted Label: NON-HATE
Non-Hate Intensity: 95.81999969482422%

Sentence: You're so stupid and annoying.
Predicted Label: HATE
Hate Intensity: 88.68000030517578%

Sentence: This is a peaceful and loving community.
Predicted Label: NON-HATE
Non-Hate Intensity: 64.98999786376953%
```

References

Hugging Face, *Datasets and Tokenizers Libraries*. [Online]. Available: https://huggingface.co/docs

Hugging Face, *Transformers Documentation*. [Online]. Available: https://huggingface.co/transformers/

Kaggle, *Toxic Comment Classification Challenge*. [Online]. Available: https://www.kaggle.com/datasets/waalbannyantudre/hate-speech-detection-curated-dataset