Shraddha Bangad
Shalin Parikh
Shivangi Sharma

Checkpoint 3: Natural Language Processing

We decided to answer two questions from the data using NLP - whether the complaint narrative is indicative of the allegation category, and whether the narrative is indicative of the sustainment of a complaint against a police officer.

Our team has been analyzing complaints against officers to see if there are any metrics that can be indicative of why a complaint against a police officer is sustained/unsustained. The first 4 checkpoints have proven to be quite informative in terms of how the race, gender, rank, salary, age etc. could be indicative of the same. We have also looked into what allegation categories are complaints the most sustained for.

In this checkpoint, we decided to delve into the textual data a bit more to see if we could find any significant results.

1. Multi class text classification using LSTMs - Predicting allegation category on the basis of the complaint narrative:

For this question, we tried a lot of different algorithms and methods. We first tried to use BERT on the data. Although the accuracy of the predictions was satisfactory, we decided to scrap this method as the inference process for the data was taking a really long time. We then decided to use LSTM for this process.

The process includes installing all the required libraries, cleaning and preprocessing the data, specifying the neural network architecture and finally training and testing the model on the data.

We obtained an accuracy of ~84% on the test data, with the model returning satisfactory classifications on user entered data. Given below are the screenshots of the model and the output.

```
[95] df = df[['crid','cr_text','final_finding','category']]
```

0s

```
df.info()
```

0s

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 40844 entries, 0 to 40843
Data columns (total 4 columns):
 #   Column         Non-Null Count  Dtype 
---  ------         --------------  ----- 
 0   crid           40844 non-null  object
 1   cr_text        1239 non-null   object
 2   final_finding  33215 non-null  object
 3   category       40844 non-null  object
dtypes: object(4)
memory usage: 1.2+ MB
```

```
df.category.value_counts()
```

```
Operation/Personnel Violations    78356
Use Of Force                      57574
Illegal Search                    37611
Lockup Procedures                 14396
Verbal Abuse                      13795
False Arrest                       9160
Conduct Unbecoming (Off-Duty)      8381
Traffic                            8193
Criminal Misconduct                6665
Domestic                           6506
Supervisory Responsibilities       4826
Drug / Alcohol Abuse               1358
Bribery / Official Corruption       885
Money / Property                    357
First Amendment                      77
Unknown                              64
Racial Profiling                     56
Excessive Force                      31
Medical                              30
Name: category, dtype: int64
```

```
model = Sequential()
model.add(Embedding(MAX_NB_WORDS, EMBEDDING_DIM, input_length=X.shape[1]))
model.add(SpatialDropout1D(0.2))
model.add(LSTM(100, dropout=0.2, recurrent_dropout=0.2))
model.add(Dense(12, activation='softmax'))
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
print(model.summary())
```

```
Model: "sequential_4"
_________________________________________________________________
 Layer (type)                 Output Shape              Param #
=================================================================
 embedding_4 (Embedding)     (None, 250, 100)          5000000

 spatial_dropout1d_4 (Spatia  (None, 250, 100)         0
 lDropout1D)

 lstm_4 (LSTM)               (None, 100)               80400

 dense_4 (Dense)             (None, 12)                1212

=================================================================
Total params: 5,081,612
Trainable params: 5,081,612
Non-trainable params: 0
_________________________________________________________________
None
```

```
new_complaint = ['The police officer forcefully entered my house and planted a weapon on me']
```

```
[[0.02664435 0.03742788 0.01714419 0.03418231 0.02543867 0.19262134
  0.01439155 0.18102702 0.01664435 0.12957676 0.05589838 0.11655685
  0.11818523 0.0342612 ]] False Arrest
```

2. Multi class text classification using Scikit-Learn - Predicting sustainment on the basis of the complaint narrative:

We next analyzed whether we can predict whether a complaint is sustained or not on the basis of the text given in the police complaint. We used LSTMs to train this model too. But unlike the previous model, this model did not give us accurate results. Even after repeated adjustments to the hyperparameters and iterations and changes to the model, the accuracy did not improve. We tried to use scikit-learn to train the model too, but to no avail. Given below are the screenshots for the model.

```
[7]  df.final_finding.value_counts()
```

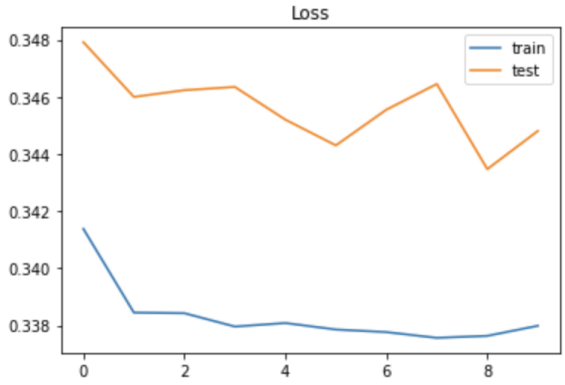
```
NS    114166
UN     53094
SU     21989
EX     17322
NC        47
DS        12
Name: final_finding, dtype: int64
```

```
[11] df = df[['crid','cr_text','final_finding','category']]
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 248321 entries, 0 to 248320
Data columns (total 4 columns):
 #   Column         Non-Null Count   Dtype 
---  ------         --------------   ----- 
 0   crid           248321 non-null  object
 1   cr_text        6222 non-null    object
 2   final_finding  206630 non-null  object
 3   category       248321 non-null  object
dtypes: object(4)
memory usage: 7.6+ MB
```

```
[28] plt.title('Loss')
     plt.plot(history.history['loss'], label='train')
     plt.plot(history.history['val_loss'], label='test')
     plt.legend()
     plt.show();
```




```
new_complaint = ['The police officer forcefully entered my house and planted a weapon on me']
seq = tokenizer.texts_to_sequences(new_complaint)
padded = pad_sequences(seq, maxlen=MAX_SEQUENCE_LENGTH)
pred = model.predict(padded)
labels = ['NS', 'UN', 'SU', 'EX','NC','DS']
print(pred, labels[np.argmax(pred)])
```

```
[[0.23494983 0.3895837  0.15206465 0.22340186]] UN
```

Summary:

Over the course of creating our NLP model, we found out that while the complaint narrative was a good indicative of the allegation category of the complaint, it cannot be used to predict the final finding associated with the complaint. This could be due to many reasons -

1. There is too little data to train our model on: We had ~6000 data points to train our model, this could have been one of the reasons why the model could not perform well on test and validation data.

2. There could be other factors affecting complaint sustainment: Complaint narrative alone cannot be an indication of the final finding. It could depend on many other factors such as the investigator assigned to the case, the rank, race, age, gender etc. of the police officer against whom the complaint is filed, the previous record of that policeman etc.

On the other hand, the NLP model seems to hold true for predicting the allegation category of the complaint, since the complaints are most likely to be categorized based on the context of the narrative. There could be a slight bias in the categorization depending on who exactly is categorizing the complaints, but the model seems to overcome that and categorizes

comments with a high accuracy.

Future NLP research using the data could be improved by using a more balanced dataset, either by considering only the well-represented categories, or by gathering more data for the underrepresented categories. We could also try using a higher power GPU to run pre-trained BERT models on the given data.