Management and Analytics for Business
UDA Final Project

Tweet Sentiment Extraction

Malinskaya Evgenia, Sidikova Yana, MMA 214

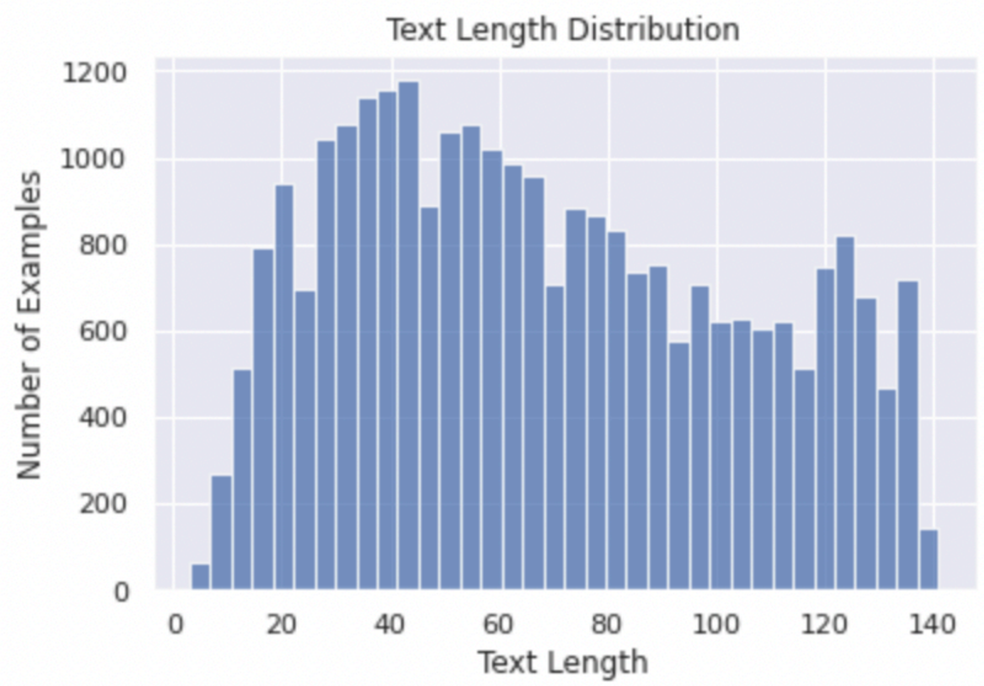
Saint Petersburg, 2023

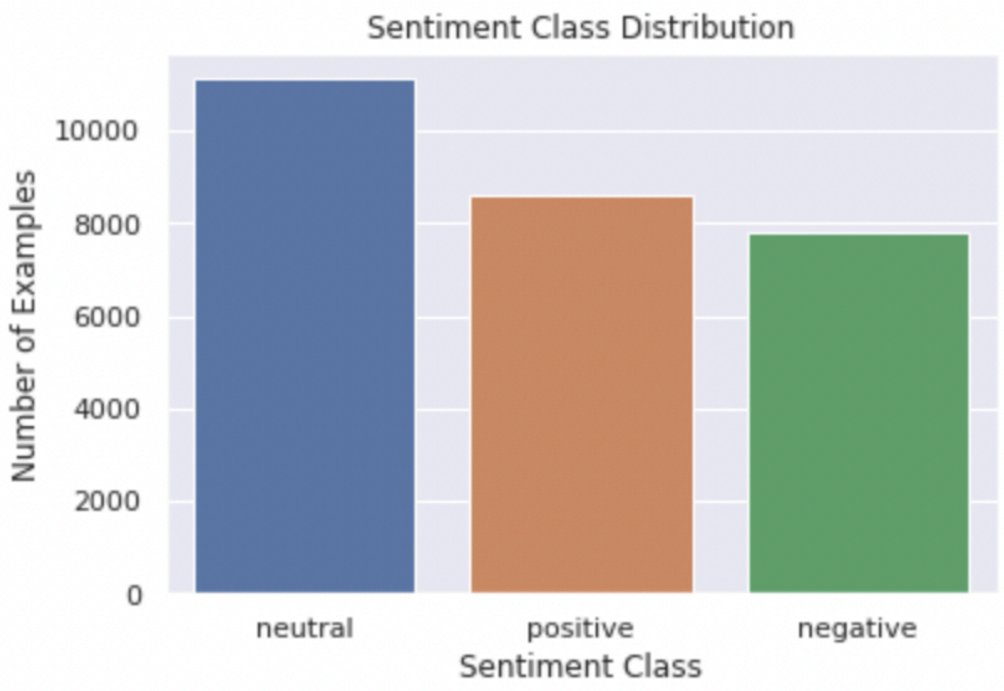
Our research is aimed at developing model that accurately predicts the sentiment label of a given tweet

Data Description

- Number of unique tweets: 27481
- The length varies from 3 to 140 words

- Number of unique sentiments: 3
- Number of missing sentiments: 0





Methodology

1. Tokenize Tweets: tweets are split into words keeping the maximum number of words based on word frequency
2. Encode the sentiment labels
3. Define the Bi-LSTM model and its architecture
4. Compile (define loss, metrics, optimizer) and train the model
5. Evaluate the model on the test set and compute evaluation metrics
6. Compare model with the performance of others relevant models (such as SVM, Naive Bayes etc.)
7. Make a decision what model is the most accurate in tweets sentiment predictions

Model Description

Bidirectional Long Short-Term Memory (Bi-LSTM) model is chosen as the most effective model for tweets sentiment analysis

Developed architecture:

- An embedding layer to map each word in the tweet to a high-dimensional vector
- A dropout layer to prevent overfitting
- A Bi-LSTM layer to process the sequence of word vectors in both forward and backward directions
- Added a TimeDistributed layer with a Softmax activation function to predict the sentiment label for each word in the tweet

```
# Define the Bi-LSTM model
input_layer = Input(shape=(max_len,))
embedding_layer = tf.keras.layers.Embedding(input_dim=len(tokenizer.word_index) + 1, output_dim=128, input_length=max_len)(input_layer)
dropout_layer = Dropout(0.2)(embedding_layer)
lstm_layer = Bidirectional(LSTM(128, return_sequences=True))(dropout_layer)
output_layer = TimeDistributed(Dense(len(sentiments), activation='softmax'))(lstm_layer)
```

Model Description

The model is trained on 100 epochs using the sparse categorical cross-entropy loss function and the Adam optimizer

```
# Check the shapes of the input data and labels
print('X_train shape:', X_train.shape)
print('y_train shape:', y_train.shape)

# Define the model architecture
model = Sequential()
model.add(Embedding(input_dim=vocab_size, output_dim=32, input_length=max_len))
model.add(Bidirectional(LSTM(32)))
model.add(Dense(3, activation='softmax'))

# Compile the model
model.compile(loss='sparse_categorical_crossentropy', optimizer='adam', metrics=['accuracy'])

# Train the model
history = model.fit(X_train, y_train, validation_split=0.2, epochs=100, batch_size=len(X_train))
```

Evaluation Results

```
Epoch 99/100
1/1 [==============================] - 4s 4s/step - loss: 0.3393 - accuracy: 0.8804 - val_loss: 1.0072 - val_accuracy: 0.6269
Epoch 100/100
1/1 [==============================] - 4s 4s/step - loss: 0.3317 - accuracy: 0.8846 - val_loss: 1.0147 - val_accuracy: 0.6267
```

Evaluate the model on the test set:

```
Test accuracy: 0.6501
```

After training the Bi-LSTM model on the dataset, we achieved an accuracy of 65% on the test set, which is comparable to the state-of-the-art results reported in prior research. The model showed good performance in predicting the sentiment labels for tweets in the dataset

Model Comparison

Metric	Bi-LSTM	Naive Bayes
Accuracy	0.650	0.632
Precision	0.658	0.634
Recall	0.650	0.632
F1	0.651	0.632