SYMBIOSIS INSTITUTE OF TECHNOLOGY (SIT)
Constituent of Symbiosis International (Deemed University), Pune
(Established under Section 3 of the UGC Act of 1956 vide notification number F-9-12/2001-U-3 of the Government of india)
Re-Accredited by NAAC with 'A' Grade
॥वसुधैव कुटुम्बकम्॥

NATURAL LANGUAGE PROCESSING
MINI PROJECT

ARTICLES HEADLINE GENERATOR

Group Members:

1. Bunny Saini – 19070122147
2. Shreya Bhongale – 19070122162

Github link of the project:
https://github.com/Shreya2001/headline-generator.git

Step 1: Merging all the files

The dataset consisted of three csv files. Those csv files were converted and merged into one dataframe using the 'pandas' library.

```
curr_dir = 'Articles/'
all_headlines = []
for filename in os.listdir(curr_dir):
    print(filename)
    if 'articles' in filename:
        article_df = pd.read_csv(curr_dir + filename)
        all_headlines.extend(list(article_df.title.values))

all_headlines = [h for h in all_headlines if type(h) == str]
len(all_headlines)
```

```
articles1.csv
articles2.csv
articles3.csv
```

Step 2: Cleaning the data

2.1. Removing '- The New York Times' from the headlines
2.2. Removing Punctuations
2.3. Turning the data into lowercase
2.4. Ignoring Symbols

```
import re
def clean_text(txt):
    t = txt
    if 'The New York Times' in txt:
        t = txt.replace('The New York Times','')
    t = re.sub(r'[^\w\s]', '', t).lower()
    t = t.encode("utf8").decode("ascii",'ignore')
    return t

corpus = [clean_text(x) for x in all_headlines]
corpus[:10]
```

```
['house republicans fret about winning their health care suit  ',
 'rift between officers and residents as killings persist in south bronx  ',
 'tyrus wong bambi artist thwarted by racial bias dies at 106  ',
 'among deaths in 2016 a heavy toll in pop music  ',
 'kim jongun says north korea is preparing to test longrange missile  ',
 'sick with a cold queen elizabeth misses new years service  ',
 'taiwans president accuses china of renewed intimidation  ',
 'after the biggest loser their bodies fought to regain weight  ',
 'first a mixtape then a romance  ',
 'calling on angels while enduring the trials of job  ']
```

Step 3: Tokenization

```
def get_sequence_of_tokens(corpus):
    ## tokenization
    tokenizer.fit_on_texts(corpus)
    total_words = len(tokenizer.word_index) + 1

    ## convert data to sequence of tokens
    input_sequences = []
    for line in corpus:
        token_list = tokenizer.texts_to_sequences([line])[0]
        for i in range(1, len(token_list)):
            n_gram_sequence = token_list[:i+1]
            input_sequences.append(n_gram_sequence)
    return input_sequences, total_words
```

Step 4: Padding the Sequences and obtaining Variables

```
def generate_padded_sequences(input_sequences):
    max_sequence_len = max([len(x) for x in input_sequences])
    input_sequences = np.array(pad_sequences(input_sequences, maxlen=max_sequence_len, padding='pre'))

    predictors, label = input_sequences[:,:-1],input_sequences[:,-1]
    label = np_utils.to_categorical(label, num_classes=total_words)
    return predictors, label, max_sequence_len

predictors, label, max_sequence_len = generate_padded_sequences(inp_sequences)
```

Step 5: Creating LSTM Model

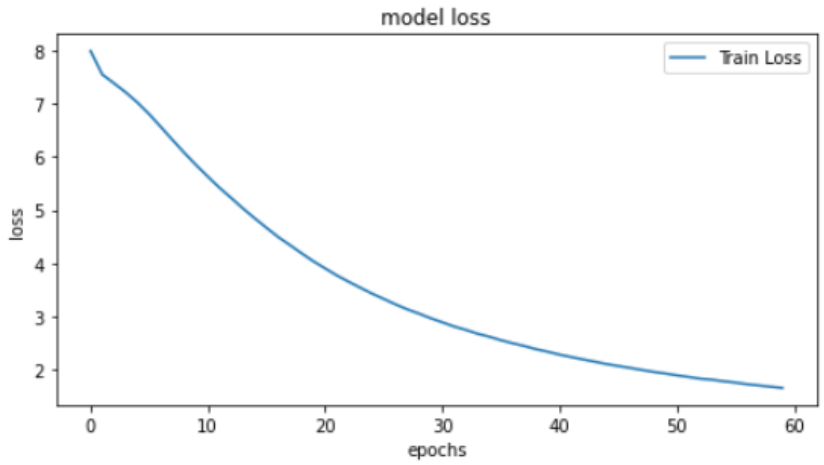
```
def create_model(max_sequence_len, total_words):
    input_len = max_sequence_len - 1
    model = Sequential()

    # Add Input Embedding Layer
    model.add(Embedding(total_words, 10, input_length=input_len))

    # Add Hidden Layer 1 - LSTM Layer
    model.add(LSTM(100))
    model.add(Dropout(0.1))

    # Add Output Layer
    model.add(Dense(total_words, activation='softmax'))

    model.compile(loss='categorical_crossentropy', optimizer='adam')

    return model

model = create_model(max_sequence_len, total_words)
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 17, 10)            96200

 lstm (LSTM)                 (None, 100)               44400

 dropout (Dropout)           (None, 100)               0

 dense (Dense)               (None, 9620)              971620

=================================================================
Total params: 1,112,220
Trainable params: 1,112,220
Non-trainable params: 0
_________________________________________________________________
```

Step 6: Training the model with 60 epochs

```
In [9]: lstm = model.fit(predictors, label, epochs=60, verbose=1, batch_size=32)
1218/1218 [==============================] - 21s 17ms/step - loss: 1.9003
Epoch 52/60
1218/1218 [==============================] - 22s 18ms/step - loss: 1.8697
Epoch 53/60
1218/1218 [==============================] - 22s 18ms/step - loss: 1.8365
Epoch 54/60
1218/1218 [==============================] - 22s 18ms/step - loss: 1.8180
Epoch 55/60
1218/1218 [==============================] - 22s 18ms/step - loss: 1.7890
Epoch 56/60
1218/1218 [==============================] - 22s 18ms/step - loss: 1.7637
Epoch 57/60
1218/1218 [==============================] - 22s 18ms/step - loss: 1.7332
Epoch 58/60
1218/1218 [==============================] - 22s 18ms/step - loss: 1.7101
Epoch 59/60
1218/1218 [==============================] - 20s 16ms/step - loss: 1.6867
Epoch 60/60
1218/1218 [==============================] - 17s 14ms/step - loss: 1.6630
```

Loss Graph:



Snapshots of the OUTPUT

```
print(generate_text("Europe", 7, model, max_sequence_len))
print(generate_text("donald trump", 6, model, max_sequence_len))
print(generate_text("elon musk", 9, model, max_sequence_len))
print(generate_text("Sweden", 6, model, max_sequence_len))
```

```
1/1 [==============================] - 0s 38ms/step
1/1 [==============================] - 0s 40ms/step
1/1 [==============================] - 0s 40ms/step
1/1 [==============================] - 0s 36ms/step
1/1 [==============================] - 0s 40ms/step
1/1 [==============================] - 0s 42ms/step
1/1 [==============================] - 0s 37ms/step
Europe Combats A New Foe Of Political Stability
1/1 [==============================] - 0s 34ms/step
1/1 [==============================] - 0s 44ms/step
1/1 [==============================] - 0s 43ms/step
1/1 [==============================] - 0s 40ms/step
1/1 [==============================] - 0s 40ms/step
1/1 [==============================] - 0s 36ms/step
Donald Trump And Hillary Clinton During The Debate
1/1 [==============================] - 0s 48ms/step
1/1 [==============================] - 0s 50ms/step
1/1 [==============================] - 0s 43ms/step
1/1 [==============================] - 0s 43ms/step
1/1 [==============================] - 0s 41ms/step
1/1 [==============================] - 0s 43ms/step
1/1 [==============================] - 0s 38ms/step
1/1 [==============================] - 0s 40ms/step
1/1 [==============================] - 0s 40ms/step
Elon Musk Says Pending Tesla Updates Could Have Prevented Fatal Crash
1/1 [==============================] - 0s 45ms/step
1/1 [==============================] - 0s 45ms/step
1/1 [==============================] - 0s 36ms/step
1/1 [==============================] - 0s 43ms/step
1/1 [==============================] - 0s 56ms/step
1/1 [==============================] - 0s 50ms/step
Sweden To Close Your Thursday Evening Briefing
```