In [1]:
```
import json
import numpy as np
import tensorflow as tf
from tensorflow import keras
from sklearn.model_selection import train_test_split
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Embedding, LSTM, Dense
```

In [2]:
```
# Define the emotions mapping
emotions = {
    0: 'sadness',
    1: 'joy',
    2: 'love',
    3: 'anger',
    4: 'fear',
    5: 'surprise'
}
```

In [3]:
```
# Load the data from the JSONL file
data = []
with open('data.jsonl') as f:
    for line in f:
        record = json.loads(line.strip())
        data.append((record['text'], record['label']))
```

```
#Load the data
with open("data.jsonl", "r") as f:
    data = [json.loads(line) for line in f]
```

In [4]:
```
data
```

```
o succeed and it just didn t happen here',
  0),
 ('im alone i feel awful', 0),
 ('ive probably mentioned this before but i really do feel proud of myself
for actually keeping up with my new years resolution of monthly and weekly
goals',
  1),
 ('i was feeling a little low few days back', 0),
 ('i beleive that i am much more sensitive to other peoples feelings and te
nd to be more compassionate',
  2),
 ('i find myself frustrated with christians because i feel that there is co
nstantly a talk about loving one another being there for each other and pra
ying for each other and i have seen that this is not always the case',
  2),
 ('i am one of those people who feels like going to the gym is only worthwh
ile if you can be there for an hour or more',
  1),
 ('i feel especially pleased about this as this has been a long time comin
g',
```

In [5]:
```
# Convert the labels to one-hot encoding
labels = [label for _, label in data]
num_classes = len(emotions)
y = []
for label in labels:
    one_hot = [0] * num_classes
    one_hot[label] = 1
    y.append(one_hot)
```

In [6]:
```
labels
```

Out[6]:
```
[0,
 0,
 1,
 0,
 2,
 2,
 1,
 1,
 1,
 3,
 3,
 0,
 1,
 3,
 3,
 1,
 0,
 4,
 1,
```

In [7]:
```
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split([text for text, _ in data]
```

In [8]:
```
# Tokenize the text and pad the sequences
max_words = 1000
max_len = 100
tokenizer = Tokenizer(num_words=max_words)
tokenizer.fit_on_texts(X_train)
X_train_seq = tokenizer.texts_to_sequences(X_train)
X_test_seq = tokenizer.texts_to_sequences(X_test)
X_train_pad = pad_sequences(X_train_seq, maxlen=max_len)
X_test_pad = pad_sequences(X_test_seq, maxlen=max_len)
```

In [9]:
```
# Define the model architecture
embedding_size = 32
model = Sequential()
model.add(Embedding(max_words, embedding_size, input_length=max_len))
model.add(LSTM(32))
model.add(Dense(num_classes, activation='softmax'))
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['acc
```

In [11]:
```
import numpy as np

y_train = np.array(y_train)
y_test = np.array(y_test)
```

In [12]:
```
# Train the model
batch_size = 32
epochs = 10
model.fit(X_train_pad, y_train, batch_size=batch_size, epochs=epochs, validati
```

```
Epoch 1/10
10421/10421 [==============================] - 316s 30ms/step - loss: 0.3856
- accuracy: 0.8469 - val_loss: 0.2673 - val_accuracy: 0.8793
Epoch 2/10
10421/10421 [==============================] - 333s 32ms/step - loss: 0.2580
- accuracy: 0.8831 - val_loss: 0.2552 - val_accuracy: 0.8804
Epoch 3/10
10421/10421 [==============================] - 371s 36ms/step - loss: 0.2467
- accuracy: 0.8864 - val_loss: 0.2502 - val_accuracy: 0.8834
Epoch 4/10
10421/10421 [==============================] - 419s 40ms/step - loss: 0.2402
- accuracy: 0.8883 - val_loss: 0.2465 - val_accuracy: 0.8828
Epoch 5/10
10421/10421 [==============================] - 342s 33ms/step - loss: 0.2355
- accuracy: 0.8898 - val_loss: 0.2454 - val_accuracy: 0.8834
Epoch 6/10
10421/10421 [==============================] - 404s 39ms/step - loss: 0.2317
- accuracy: 0.8916 - val_loss: 0.2453 - val_accuracy: 0.8818
Epoch 7/10
10421/10421 [==============================] - 394s 38ms/step - loss: 0.2282
- accuracy: 0.8929 - val_loss: 0.2462 - val_accuracy: 0.8843
Epoch 8/10
10421/10421 [==============================] - 36668s 4s/step - loss: 0.2249
- accuracy: 0.8946 - val_loss: 0.2474 - val_accuracy: 0.8830
Epoch 9/10
10421/10421 [==============================] - 276s 26ms/step - loss: 0.2221
- accuracy: 0.8951 - val_loss: 0.2471 - val_accuracy: 0.8822
Epoch 10/10
10421/10421 [==============================] - 280s 27ms/step - loss: 0.2193
- accuracy: 0.8959 - val_loss: 0.2491 - val_accuracy: 0.8795
```

Out[12]: <keras.callbacks.History at 0x13df05b5e40>

In [13]:
```
# Make predictions on new data
text = 'I am feeling happy today!'
text_seq = tokenizer.texts_to_sequences([text])
text_pad = pad_sequences(text_seq, maxlen=max_len)
probas = model.predict(text_pad)[0]
label = probas.argmax()
emotion = emotions[label]
```

```
1/1 [==============================] - 1s 793ms/step
```

In [14]:
```
print(f'Predicted label: {label}')
print(f'Predicted emotion: {emotion}')
```

```
Predicted label: 1
Predicted emotion: joy
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

```
import json
from sklearn.model_selection import train_test_split
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Embedding, LSTM, Dense

# Define the emotions mapping
emotions = {
    0: 'sadness',
    1: 'joy',
    2: 'love',
    3: 'anger',
    4: 'fear',
    5: 'surprise'
}

# Load the data from the JSONL file
data = []
with open('data.jsonl') as f:
    for line in f:
        record = json.loads(line.strip())
        data.append((record['text'], record['label']))

# Convert the labels to one-hot encoding
labels = [label for _, label in data]
num_classes = len(emotions)
y = []
for label in labels:
    one_hot = [0] * num_classes
    one_hot[label] = 1
    y.append(one_hot)

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split([text for text, _ in data]

# Tokenize the text and pad the sequences
max_words = 1000
max_len = 100
tokenizer = Tokenizer(num_words=max_words)
tokenizer.fit_on_texts(X_train)
X_train_seq = tokenizer.texts_to_sequences(X_train)
X_test_seq = tokenizer.texts_to_sequences(X_test)
X_train_pad = pad_sequences(X_train_seq, maxlen=max_len)
X_test_pad = pad_sequences(X_test_seq, maxlen=max_len)

# Define the model architecture
embedding_size = 32
model = Sequential()
model.add(Embedding(max_words, embedding_size, input_length=max_len))
model.add(LSTM(32))
model.add(Dense(num_classes, activation='softmax'))
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['acc

# Train the model
batch_size = 32
epochs = 10
```

```
model.fit(X_train_pad, y_train, batch_size=batch_size, epochs=epochs, validati

# Make predictions on new data
text = 'I am feeling happy today!'
text_seq = tokenizer.texts_to_sequences([text])
text_pad = pad_sequences(text_seq, maxlen=max_len)
probas = model.predict(text_pad)[0]
label = probas.argmax()
emotion = emotions[label]

print(f'Predicted label: {label}')
print(f'Predicted emotion: {emotion}')
```