In [3]:
```
import tensorflow as tf
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
import pandas as pd
import numpy as np
```

In [4]:
```
data = pd.read_json('Sarcasm_Headlines_Dataset.json', lines=True)
```

In [5]:
```
tokenizer = Tokenizer(oov_token="<OOV>")
tokenizer.fit_on_texts(data['headline'])
sequences = tokenizer.texts_to_sequences(data['headline'])
padded_sequences = pad_sequences(sequences, padding='post', truncating='post', maxlen=
```

In [6]:
```
labels = np.array(data['is_sarcastic'])
```

In [7]:
```
train_size = int(0.8 * len(data))
train_sentences = padded_sequences[:train_size]
train_labels = labels[:train_size]
test_sentences = padded_sequences[train_size:]
test_labels = labels[train_size:]
```

In [8]:
```
model = tf.keras.Sequential([
    tf.keras.layers.Embedding(input_dim=len(tokenizer.word_index) + 1, output_dim=16,
    tf.keras.layers.GlobalAveragePooling1D(),
    tf.keras.layers.Dense(24, activation='relu'),
    tf.keras.layers.Dense(1, activation='sigmoid')
])
```

In [9]:
```
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
```

In [10]:
```
history = model.fit(train_sentences, train_labels, epochs=10, validation_data=(test_se
```

```
Epoch 1/10
668/668 [==============================] - 3s 3ms/step - loss: 0.6553 - accuracy: 0.6
042 - val_loss: 0.5399 - val_accuracy: 0.7737
Epoch 2/10
668/668 [==============================] - 2s 3ms/step - loss: 0.4006 - accuracy: 0.8
485 - val_loss: 0.3685 - val_accuracy: 0.8484
Epoch 3/10
668/668 [==============================] - 2s 3ms/step - loss: 0.2782 - accuracy: 0.8
941 - val_loss: 0.3446 - val_accuracy: 0.8540
Epoch 4/10
668/668 [==============================] - 2s 3ms/step - loss: 0.2137 - accuracy: 0.9
199 - val_loss: 0.3394 - val_accuracy: 0.8553
Epoch 5/10
668/668 [==============================] - 2s 3ms/step - loss: 0.1702 - accuracy: 0.9
387 - val_loss: 0.3473 - val_accuracy: 0.8566
Epoch 6/10
668/668 [==============================] - 2s 3ms/step - loss: 0.1336 - accuracy: 0.9
539 - val_loss: 0.3776 - val_accuracy: 0.8474
Epoch 7/10
668/668 [==============================] - 2s 3ms/step - loss: 0.1070 - accuracy: 0.9
650 - val_loss: 0.3847 - val_accuracy: 0.8523
Epoch 8/10
668/668 [==============================] - 2s 3ms/step - loss: 0.0837 - accuracy: 0.9
745 - val_loss: 0.4086 - val_accuracy: 0.8504
Epoch 9/10
668/668 [==============================] - 2s 3ms/step - loss: 0.0666 - accuracy: 0.9
811 - val_loss: 0.4322 - val_accuracy: 0.8519
Epoch 10/10
668/668 [==============================] - 2s 3ms/step - loss: 0.0530 - accuracy: 0.9
854 - val_loss: 0.4864 - val_accuracy: 0.8399
```

In [11]:
```
test_loss, test_acc = model.evaluate(test_sentences, test_labels, verbose=2)
```

```
167/167 - 0s - loss: 0.4864 - accuracy: 0.8399 - 82ms/epoch - 493us/step
```

In [12]:
```
print("\nTest accuracy:", test_acc)
```

```
Test accuracy: 0.8399475812911987
```

In [ ]: