In [0]:
```
import tensorflow as tf
from tensorflow import keras

import numpy as np

import matplotlib.pyplot as plt

import pandas as pd
```

In [3]:
```
#upload the aggregated data table

from google.colab import files
import io
uploaded = files.upload()
filename = list(uploaded.keys())
```

Choose Files No file chosen

Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving checkpoint_5.csv to checkpoint_5.csv
```

In [47]:
```
# Read in the file as a pandas dataframe

data = pd.read_csv(filename[0], #header=None
                   na_values="?" )
data.head()
```

Out[47]:

| | id | case_number | category | amount | incident_date | location_address | longitude | |
|---|---|---|---|---|---|---|---|---|
| **0** | 1 | 16-cv-9149 | 11 | $100,000.00 | 2016-09-21 00:00:00 | 700 E. 111th Street | -87.605277 | 4 |
| **1** | 2 | 14-cv-4800 | 9 | $35,000.00 | 2016-06-22 00:00:00 | NaN | 0.000000 | 0 |
| **2** | 3 | 16-cv-347 | 9 | $20,000.00 | 2015-11-04 00:00:00 | 100 W. Van Buren St. | -87.630740 | 4 |
| **3** | 4 | 16-cv-7540 | 5 | $50,000.00 | 2015-10-13 00:00:00 | 1600 W. Glenlake Ave. | -87.670208 | 4 |
| **4** | 5 | 15-cv-8757 | 9 | $47,000.00 | 2015-10-08 00:00:00 | NaN | 0.000000 | 0 |

In [48]:
```
data = data[['description','is_bad_officer_involved']]
data.head()
```

Out[48]:

| | description | is_bad_officer_involved |
|---|---|---|
| **0** | Jane Doe went to a police station, accompanie... | 0 |
| **1** | Guzman was the victim of a series of unwarrant... | 0 |
| **2** | Salazar was driving when Officer Donald stoppe... | 0 |
| **3** | Perez was in his home in the Edgewater neighbo... | 0 |
| **4** | Crockett was arrested without a warrant and ta... | 0 |

In [0]:
```
columndata = data['description'].tolist()
```

In [0]:
```
# Import the Universal Sentence Encoder's TF Hub module
import tensorflow_hub as hub
embed = hub.Module("https://tfhub.dev/google/universal-sentence-encoder/2")
```

In [55]:
```
# Embed the descriptions into vectors

messages = columndata

with tf.Session() as session:
  session.run([tf.global_variables_initializer(), tf.tables_initializer()])
  message_embeddings = session.run(embed(messages))
```

```
INFO:tensorflow:Saver not created because there are no variables in the graph to restore
```

In [56]:
```
message_embeddings.shape
```

Out[56]:
```
(943, 512)
```

In [57]:
```
data.shape
data.head()
```

Out[57]:

| | description | is_bad_officer_involved |
|---|---|---|
| **0** | Jane Doe went to a police station, accompanie... | 0 |
| **1** | Guzman was the victim of a series of unwarrant... | 0 |
| **2** | Salazar was driving when Officer Donald stoppe... | 0 |
| **3** | Perez was in his home in the Edgewater neighbo... | 0 |
| **4** | Crockett was arrested without a warrant and ta... | 0 |

In [0]:
```
fulldata = np.column_stack((data['is_bad_officer_involved'],message_embeddings))

columnlist = []
for i in range(513):
    columnlist.append('Description_' + str(i))
columnlist[0] = 'is_bad_officer_involved'
```

In [0]:
```
#randomly select 80% of our data for training and the remaining 20% for
 testing
df = pd.DataFrame(fulldata, columns=columnlist)
train=df.sample(frac=0.8)
test=df.drop(train.index)

#split the data into x, our input, and y, our output
train_y = train['is_bad_officer_involved'].values
train_x = train.drop('is_bad_officer_involved',axis=1).values

test_y = test['is_bad_officer_involved'].values
test_x = test.drop('is_bad_officer_involved',axis=1).values
```

In [137]:
```
from keras.models import Sequential
from keras.layers import Dense

model = Sequential()
model.add(Dense(units=300, activation='relu',input_dim=512))
model.add(Dense(units=300, activation='relu',input_dim=512))
model.add(Dense(units=300, activation='relu',input_dim=512))
model.add(Dense(units=1, activation='sigmoid',input_dim=512))

# Compiling model

model.compile(loss='binary_crossentropy',optimizer='adam',metrics=['accu
racy'])

trained = model.fit(x=train_x,
                    y=train_y,
                    epochs=10,
                    batch_size=512)
```

```
Epoch 1/10
754/754 [==============================] - 6s 8ms/step - loss: 0.6586 -
acc: 0.9416
Epoch 2/10
754/754 [==============================] - 0s 71us/step - loss: 0.5180
- acc: 0.9469
Epoch 3/10
754/754 [==============================] - 0s 74us/step - loss: 0.3409
- acc: 0.9469
Epoch 4/10
754/754 [==============================] - 0s 72us/step - loss: 0.2222
- acc: 0.9469
Epoch 5/10
754/754 [==============================] - 0s 79us/step - loss: 0.2351
- acc: 0.9469
Epoch 6/10
754/754 [==============================] - 0s 77us/step - loss: 0.2674
- acc: 0.9469
Epoch 7/10
754/754 [==============================] - 0s 80us/step - loss: 0.2628
- acc: 0.9469
Epoch 8/10
754/754 [==============================] - 0s 76us/step - loss: 0.2384
- acc: 0.9469
Epoch 9/10
754/754 [==============================] - 0s 80us/step - loss: 0.2138
- acc: 0.9469
Epoch 10/10
754/754 [==============================] - 0s 78us/step - loss: 0.2062
- acc: 0.9469
```

In [138]:
```
test_loss, test_acc = model.evaluate(test_x, test_y)

print('Test accuracy:', test_acc)
```

```
189/189 [==============================] - 3s 13ms/step
Test accuracy: 0.9153439125055989
```