In [7]:
```
import pandas as pd
import numpy as np
from sklearn.metrics import accuracy_score, f1_score, roc_auc_score
import sklearn.metrics as metrics
from tqdm import tqdm
import os.path
import pickle
from keras.preprocessing.text import Tokenizer, one_hot
from keras.preprocessing.sequence import pad_sequences
from keras.models import Sequential

from keras.layers.core import Activation, Dropout, Dense
from keras.layers import Flatten, LSTM
from keras.layers import GlobalMaxPooling1D
from keras.models import Model

from sklearn.model_selection import train_test_split
from keras.utils.np_utils import to_categorical
from keras.callbacks import EarlyStopping, Callback, ModelCheckpoint
from keras.layers import Dropout, Input
from keras.layers.merge import Concatenate
from keras.layers.embeddings import Embedding

import matplotlib.pyplot as plt
import nltk
import re

from numpy import array
from numpy import asarray
from numpy import zeros

nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to /Users/abhay/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

Out[7]:
```
True
```

In [8]:
```
df_train = pd.read_csv('../../Preprocessing/Data/train.csv')
```

In [9]:
```
df_train.head()
```

Out[9]:

	id	comment_text	toxic	severe_toxic	obscene	threat	insult	identity_hate
0	0000997932d777bf	Explanation\nWhy the edits made under my usern...	0	0	0	0	0	0
1	000103f0d9cfb60f	D'aww! He matches this background colour I'm s...	0	0	0	0	0	0
2	000113f07ec002fd	Hey man, I'm really not trying to edit war. It...	0	0	0	0	0	0
3	0001b41b1c6bb37e	"\nMore\nI can't make any real suggestions on ...	0	0	0	0	0	0
4	0001d958c54c6e35	You, sir, are my hero. Any chance you remember...	0	0	0	0	0	0

In [10]:
```
df_train.toxic.value_counts()
```

Out[10]:
```
0    144277
1     15294
Name: toxic, dtype: int64
```

In [11]:
```
df_train["comment_text"][168]
```

Out[11]:
```
"You should be fired, you're a moronic wimp who is too lazy to do resea
rch. It makes me sick that people like you exist in this world."
```

In [12]:
```
print("Toxic:", str(df_train["toxic"][168]))
print("Severe_toxic:", str(df_train["severe_toxic"][168]))
print("Obscene:", str(df_train["obscene"][168]))
print("Threat:", str(df_train["threat"][168]))
print("Insult:", str(df_train["insult"][168]))
print("Identity_hate:", str(df_train["identity_hate"][168]))
```

```
Toxic: 1
Severe_toxic: 0
Obscene: 0
Threat: 0
Insult: 1
Identity_hate: 0
```

In [13]:
```
labels = df_train[["toxic", "severe_toxic", "obscene", "threat", "insul
t", "identity_hate"]]
labels.head()
```

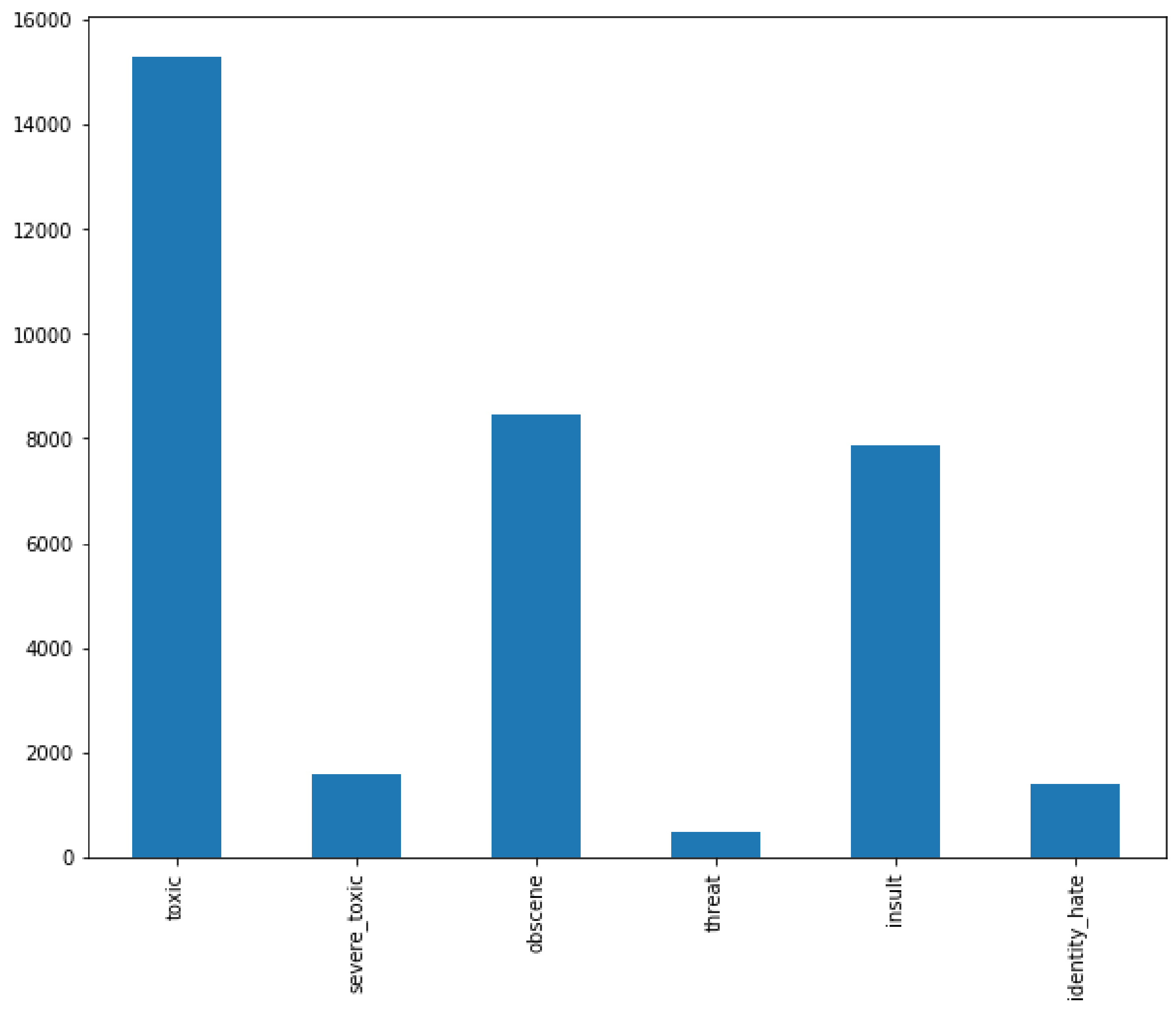
Out[13]:

	toxic	severe_toxic	obscene	threat	insult	identity_hate
0	0	0	0	0	0	0
1	0	0	0	0	0	0
2	0	0	0	0	0	0
3	0	0	0	0	0	0
4	0	0	0	0	0	0

In [14]:
```
fig_size = plt.rcParams["figure.figsize"]
fig_size[0] = 10
fig_size[1] = 8
plt.rcParams["figure.figsize"] = fig_size

labels.sum(axis=0).plot.bar()
```

Out[14]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1a323d55c0>
```



In [15]:
```
df_train.dropna(subset=['comment_text'], inplace=True)
len(df_train)
```

Out[15]:
```
159571
```

In [16]:
```
df_train['comment_text'] = [entry.lower() for entry in df_train['comment
_text']]
df_train.head()
```

Out[16]:

	id	comment_text	toxic	severe_toxic	obscene	threat	insult	identity_hate
0	0000997932d777bf	explanation\nwhy the edits made under my usern...	0	0	0	0	0	0
1	000103f0d9cfb60f	d'aww! he matches this background colour i'm s...	0	0	0	0	0	0
2	000113f07ec002fd	hey man, i'm really not trying to edit war. it...	0	0	0	0	0	0
3	0001b41b1c6bb37e	"\nmore\ni can't make any real suggestions on ...	0	0	0	0	0	0
4	0001d958c54c6e35	you, sir, are my hero. any chance you remember...	0	0	0	0	0	0

In [17]:
```
df_train['comment_text_lower'] = df_train['comment_text'].astype(str).st
r.replace('[^a-zA-Z]',' ').str.lower()
# Remove all non-letter characters and make everything lowercase.
```

In [18]:
```
stop_re = '\\b'+'\\b|\\b'.join(nltk.corpus.stopwords.words('english'))+'
\\b'
df_train['comment_text_stop'] = df_train['comment_text_lower'].astype(st
r).str.replace(stop_re, '')
```

In [19]:
```
df_train['comment_text_stop'].head(10)
```

Out[19]:
```
0    explanation   edits made   username hardcore m...
1     aww   matches  background colour   seemingly ...
2    hey man    really  trying  edit war       guy ...
3          make  real suggestions  improvement    w...
4             sir    hero   chance  remember  page
5       congratulations    well  use  tools well   ...
6                      cocksucker   piss around   work
7     vandalism   matt shirvington article    revert...
8    sorry   word  nonsense   offensive    anyway  ...
9         alignment   subject    contrary    dulithgow
Name: comment_text_stop, dtype: object
```

In [20]:
```
# toekinizing words
df_train['comment_text_final'] = df_train['comment_text_stop'].astype(st
r).str.split()
```

In [21]:
```
X_train = pd.read_pickle('../../Preprocessing/Data/X_train.pkl')
y_train = pd.read_pickle('../../Preprocessing/Data/y_train.pkl')
X_test = pd.read_pickle('../../Preprocessing/Data/X_test.pkl')
y_test = pd.read_pickle('../../Preprocessing/Data/y_test.pkl')
```

In [22]:
```
tokenizer = Tokenizer(num_words=130000)
tokenizer.fit_on_texts(X_train)

X_train = tokenizer.texts_to_sequences(X_train)
X_test = tokenizer.texts_to_sequences(X_test)

vocab_size = len(tokenizer.word_index) + 1

maxlen = 200

X_train = pad_sequences(X_train, padding='post', maxlen=maxlen)
X_test = pad_sequences(X_test, padding='post', maxlen=maxlen)
```

In [23]:
```
vocab_size
```

Out[23]:
```
61152
```

In [25]:
```
embeddings_dictionary = dict()

glove_file = open('Data/glove.6B.100d.txt', encoding="utf8")

for line in glove_file:
    records = line.split()
    word = records[0]
    vector_dimensions = asarray(records[1:], dtype='float32')
    embeddings_dictionary[word] = vector_dimensions
glove_file.close()

embedding_matrix = zeros((vocab_size, 100))
for word, index in tokenizer.word_index.items():
    embedding_vector = embeddings_dictionary.get(word)
    if embedding_vector is not None:
        embedding_matrix[index] = embedding_vector
```

In [26]:
```
deep_inputs = Input(shape=(maxlen,))
embedding_layer = Embedding(vocab_size, 100, weights=[embedding_matrix],
trainable=False)(deep_inputs)
LSTM_Layer_1 = LSTM(128)(embedding_layer)
dense_layer_1 = Dense(6, activation='sigmoid')(LSTM_Layer_1)
model = Model(inputs=deep_inputs, outputs=dense_layer_1)

model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['ac
c'])
```

In [27]:
```
class RocAucEvaluation(Callback):
    def __init__(self, validation_data=(), interval=1):
        super(Callback, self).__init__()

        self.interval = interval
        self.X_val, self.y_val = validation_data

    def on_epoch_end(self, epoch, logs={}):
        if epoch % self.interval == 0:
            y_pred = self.model.predict(self.X_val, verbose=0)
            score = roc_auc_score(self.y_val, y_pred)
            print("\n ROC-AUC - epoch: {:d} - score: {:.6f}".format(epoc
h+1, score))
ra_val = RocAucEvaluation(validation_data=(X_test, y_test), interval=1)
```

In [28]:
```
ra_val = RocAucEvaluation(validation_data=(X_test, y_test), interval=1)
early_stop = EarlyStopping(monitor='val_loss', mode='min', patience=5)
```

In [29]:
```
print(model.summary())
```

```
Model: "model_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         (None, 200)               0
_________________________________________________________________
embedding_1 (Embedding)      (None, 200, 100)          6115200
_________________________________________________________________
lstm_1 (LSTM)                (None, 128)               117248
_________________________________________________________________
dense_1 (Dense)              (None, 6)                 774
=================================================================
Total params: 6,233,222
Trainable params: 118,022
Non-trainable params: 6,115,200
_________________________________________________________________
None
```

In [31]:
```
history = None
if os.path.isfile('Models/lstm_glove.sav'):
    history = pickle.load(open('Models/lstm_glove.sav', 'rb'))
else:
    history = model.fit(X_train, y_train, batch_size = 128, epochs = 5,
validation_data = (X_test, y_test),
                        verbose = 1, callbacks = [ra_val, early_stop], v
alidation_split = 0.2)
    pickle.dump(history, open('Models/lstm_glove.sav', 'wb'))
```

```
Train on 39912 samples, validate on 19659 samples
Epoch 1/5
39912/39912 [==============================] - 139s 3ms/step - loss: 0.
2891 - acc: 0.9016 - val_loss: 0.2733 - val_acc: 0.9026

 ROC-AUC - epoch: 1 - score: 0.509909
Epoch 2/5
39912/39912 [==============================] - 138s 3ms/step - loss: 0.
2725 - acc: 0.9024 - val_loss: 0.2542 - val_acc: 0.9029

 ROC-AUC - epoch: 2 - score: 0.859188
Epoch 3/5
39912/39912 [==============================] - 132s 3ms/step - loss: 0.
1915 - acc: 0.9267 - val_loss: 0.1918 - val_acc: 0.9206

 ROC-AUC - epoch: 3 - score: 0.880922
Epoch 4/5
39912/39912 [==============================] - 135s 3ms/step - loss: 0.
1585 - acc: 0.9393 - val_loss: 0.1392 - val_acc: 0.9470

 ROC-AUC - epoch: 4 - score: 0.922844
Epoch 5/5
39912/39912 [==============================] - 138s 3ms/step - loss: 0.
1382 - acc: 0.9472 - val_loss: 0.1323 - val_acc: 0.9498

 ROC-AUC - epoch: 5 - score: 0.927736
```

In [32]:
```
score = model.evaluate(X_test, y_test, verbose=1)

print("Test Loss:", score[0])
print("Test Accuracy:", score[1])
```

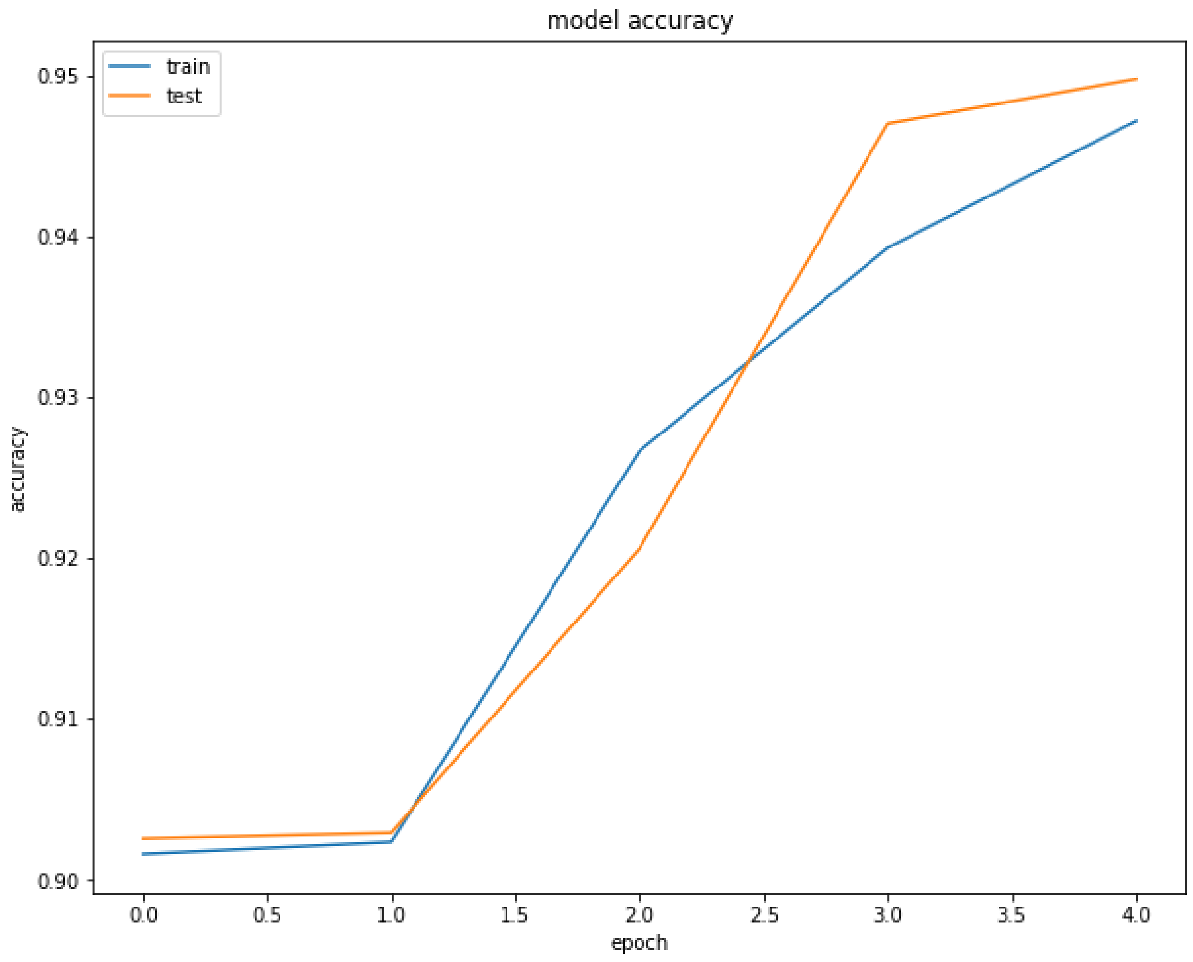
```
19659/19659 [==============================] - 21s 1ms/step
Test Loss: 0.13227544613025263
Test Accuracy: 0.949776828289032
```

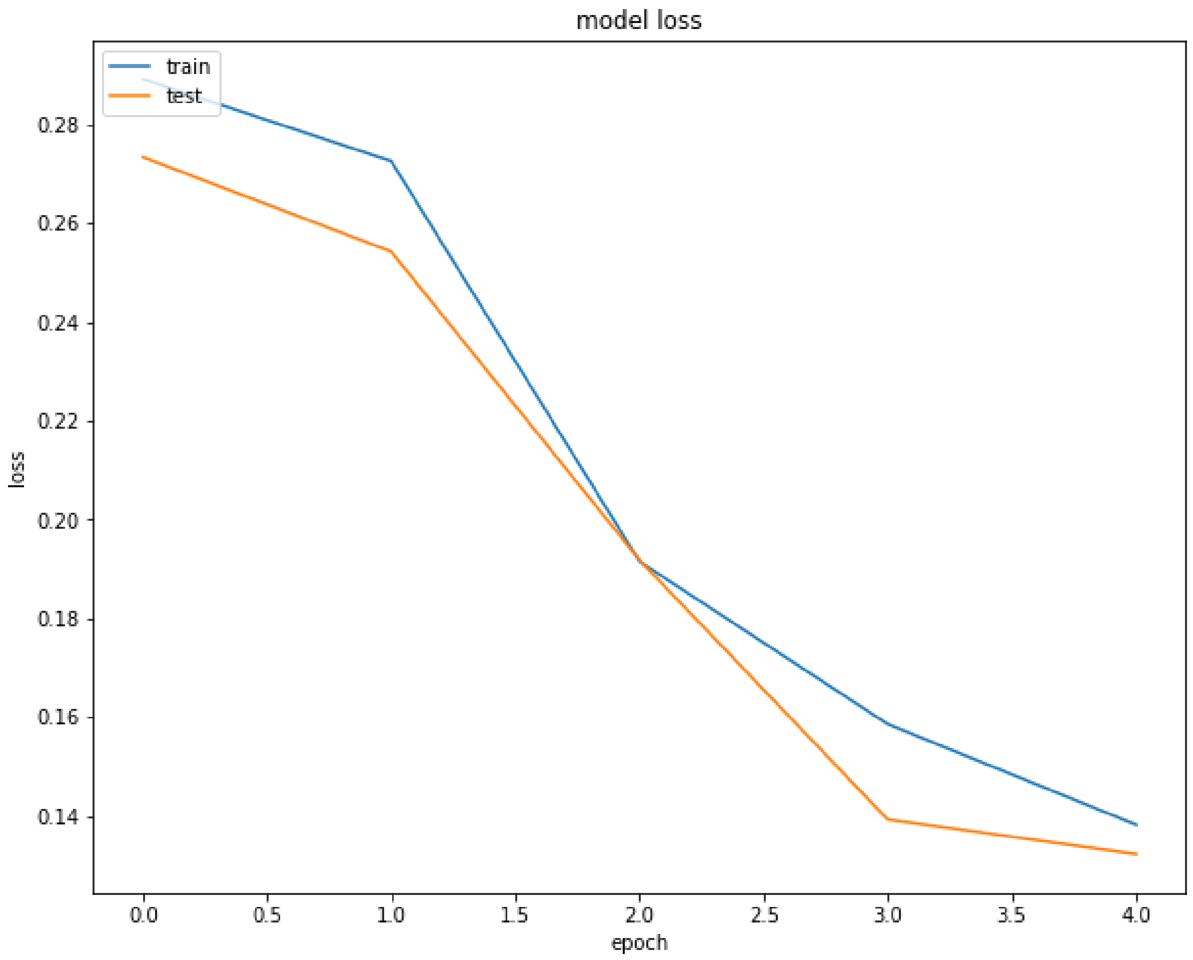
In [33]:
```
plt.plot(history.history['acc'])
plt.plot(history.history['val_acc'])

plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train','test'], loc='upper left')
plt.show()

plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])

plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train','test'], loc='upper left')
plt.show()
```


model accuracy
train
test
accuracy
epoch
0.90
0.91
0.92
0.93
0.94
0.95
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0


model loss
train
test
loss
epoch
0.28
0.26
0.24
0.22
0.20
0.18
0.16
0.14
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0

In [0]:

```

```