In [1]:
```
import os
import re
import csv
import sys
import codecs
import numpy as np
import pandas as pd
import keras.layers as KL
from nltk.corpus import stopwords
from nltk.stem import SnowballStemmer
from string import punctuation
from gensim.models import KeyedVectors
from keras import backend as KB
from keras.preprocessing.text import Tokenizer
from keras.preprocessing.sequence import pad_sequences
from keras.layers import Dense, Input, LSTM, Embedding, Dropout, Activation
from keras.layers.merge import concatenate
from keras.models import Model
from keras.layers.core import Reshape, Permute, Lambda, RepeatVector
from keras.layers.normalization import BatchNormalization
from keras.callbacks import EarlyStopping, ModelCheckpoint
```

```
C:\Users\tianh\Desktop\environments\mlenv\lib\site-packages\h5py\__init__.py:
36: FutureWarning: Conversion of the second argument of issubdtype from `floa
t` to `np.floating` is deprecated. In future, it will be treated as `np.float
64 == np.dtype(float).type`.
  from ._conv import register_converters as _register_converters
Using TensorFlow backend.
C:\Users\tianh\Desktop\environments\mlenv\lib\site-packages\gensim\utils.py:1
167: UserWarning: detected Windows; aliasing chunkize to chunkize_serial
  warnings.warn("detected Windows; aliasing chunkize to chunkize_serial")
```

In [2]:
```
BASE_DIR = 'data/'
EMBEDDING_FILE = BASE_DIR + 'GoogleNews-vectors-negative300.bin'
TRAIN_DATA_FILE = BASE_DIR + 'train.csv'
TEST_DATA_FILE = BASE_DIR + 'test.csv'
MAX_SEQUENCE_LENGTH = 30
MAX_NB_WORDS = 100000
EMBEDDING_DIM = 300
VALIDATION_SPLIT = 0.3
LOAD_DATA = False
```

In [3]:
```
num_lstm = np.random.randint(175, 275)
num_dense = np.random.randint(100, 150)
rate_drop_lstm = 0.15 + np.random.rand() * 0.25
rate_drop_dense = 0.15 + np.random.rand() * 0.25
act = 'relu'
re_weight = True # whether to re-weight classes to fit the 17.5% share in test
 set
STAMP = 'lstm'
```

In [4]:
```
print('Indexing word vectors')
word2vec = KeyedVectors.load_word2vec_format(EMBEDDING_FILE, binary=True)
print('Found %s word vectors of word2vec' % len(word2vec.vocab))
```

```
Indexing word vectors
Found 3000000 word vectors of word2vec
```

In [5]:
```
print('Processing text dataset')

# The function "text_to_wordlist" is from
# https://www.kaggle.com/currie32/quora-question-pairs/the-importance-of-cl
eaning-text
def text_to_wordlist(text, remove_stopwords=False, stem_words=False):
    # Clean the text, with the option to remove stopwords and to stem word
s.

    # Convert words to lower case and split them
    text = text.lower().split()

    # Optionally, remove stop words
    if remove_stopwords:
        stops = set(stopwords.words("english"))
        text = [w for w in text if not w in stops]

    text = " ".join(text)

    # Clean the text
    text = re.sub(r"[^A-Za-z0-9^,!.\/'+-=]", " ", text)
    text = re.sub(r"what's", "what is ", text)
    text = re.sub(r"\'s", " ", text)
    text = re.sub(r"\'ve", " have ", text)
    text = re.sub(r"can't", "cannot ", text)
    text = re.sub(r"n't", " not ", text)
    text = re.sub(r"i'm", "i am ", text)
    text = re.sub(r"\'re", " are ", text)
    text = re.sub(r"\'d", " would ", text)
    text = re.sub(r"\'ll", " will ", text)
    text = re.sub(r",", " ", text)
    text = re.sub(r"\.", " ", text)
    text = re.sub(r"!", " ! ", text)
    text = re.sub(r"\/", " ", text)
    text = re.sub(r"\^", " ^ ", text)
    text = re.sub(r"\+", " + ", text)
    text = re.sub(r"\-", " - ", text)
    text = re.sub(r"\=", " = ", text)
    text = re.sub(r"'", " ", text)
    text = re.sub(r"(\d+)(k)", r"\g<1>000", text)
    text = re.sub(r":", " : ", text)
    text = re.sub(r" e g ", " eg ", text)
    text = re.sub(r" b g ", " bg ", text)
    text = re.sub(r" u s ", " american ", text)
    text = re.sub(r"\0s", "0", text)
    text = re.sub(r" 9 11 ", "911", text)
    text = re.sub(r"e - mail", "email", text)
    text = re.sub(r"j k", "jk", text)
    text = re.sub(r"\s{2,}", " ", text)
```

```
    # Optionally, shorten words to their stems
    if stem_words:
        text = text.split()
        stemmer = SnowballStemmer('english')
        stemmed_words = [stemmer.stem(word) for word in text]
        text = " ".join(stemmed_words)

    # Return a list of words
    return(text)

texts_1 = []
texts_2 = []
labels = []
with codecs.open(TRAIN_DATA_FILE, encoding='utf-8') as f:
    reader = csv.reader(f, delimiter=',')
    header = next(reader)
    for values in reader:
        texts_1.append(text_to_wordlist(values[3]))
        texts_2.append(text_to_wordlist(values[4]))
        labels.append(int(values[5]))
print('Found %s texts in train.csv' % len(texts_1))

tokenizer = Tokenizer(num_words=MAX_NB_WORDS)
tokenizer.fit_on_texts(texts_1 + texts_2)

sequences_1 = tokenizer.texts_to_sequences(texts_1)
sequences_2 = tokenizer.texts_to_sequences(texts_2)

word_index = tokenizer.word_index
print('Found %s unique tokens' % len(word_index))

data_1 = pad_sequences(sequences_1, maxlen=MAX_SEQUENCE_LENGTH)
data_2 = pad_sequences(sequences_2, maxlen=MAX_SEQUENCE_LENGTH)
labels = np.array(labels)
print('Shape of data tensor:', data_1.shape)
print('Shape of label tensor:', labels.shape)
```

```
Processing text dataset
Found 404290 texts in train.csv
Found 85518 unique tokens
Shape of data tensor: (404290, 30)
Shape of label tensor: (404290,)
```

In [6]:
```
print('Preparing embedding matrix')

nb_words = min(MAX_NB_WORDS, len(word_index))+1

embedding_matrix = np.zeros((nb_words, EMBEDDING_DIM))
for word, i in word_index.items():
    if word in word2vec.vocab:
        embedding_matrix[i] = word2vec.word_vec(word)
print('Null word embeddings: %d' % np.sum(np.sum(embedding_matrix, axis=1) ==
0))
```

```
Preparing embedding matrix
Null word embeddings: 37391
```

In [7]:
```
perm = np.random.permutation(len(data_1))
idx_train = perm[:int(len(data_1)*(1-VALIDATION_SPLIT))]
idx_val = perm[int(len(data_1)*(1-VALIDATION_SPLIT)):]

data_1_train = np.vstack((data_1[idx_train], data_2[idx_train]))
data_2_train = np.vstack((data_2[idx_train], data_1[idx_train]))
labels_train = np.concatenate((labels[idx_train], labels[idx_train]))

data_1_val = np.vstack((data_1[idx_val], data_2[idx_val]))
data_2_val = np.vstack((data_2[idx_val], data_1[idx_val]))
labels_val = np.concatenate((labels[idx_val], labels[idx_val]))

weight_val = np.ones(len(labels_val))
if re_weight:
    weight_val *= 0.472001959
    weight_val[labels_val==0] = 1.309028344
```

In [8]:
```
embedding_layer = Embedding(nb_words,
        EMBEDDING_DIM,
        weights=[embedding_matrix],
        input_length=MAX_SEQUENCE_LENGTH,
        trainable=False)
lstm_layer = LSTM(num_lstm, dropout=rate_drop_lstm, recurrent_dropout=rate_dro
p_lstm)

sequence_1_input = Input(shape=(MAX_SEQUENCE_LENGTH,), dtype='int32')
embedded_sequences_1 = embedding_layer(sequence_1_input)
x1 = lstm_layer(embedded_sequences_1)

sequence_2_input = Input(shape=(MAX_SEQUENCE_LENGTH,), dtype='int32')
embedded_sequences_2 = embedding_layer(sequence_2_input)
y1 = lstm_layer(embedded_sequences_2)

merged = concatenate([x1, y1])
merged = Dropout(rate_drop_dense)(merged)
merged = BatchNormalization()(merged)

merged = Dense(num_dense, activation=act)(merged)
merged = Dropout(rate_drop_dense)(merged)
merged = BatchNormalization()(merged)

preds = Dense(1, activation='sigmoid')(merged)
```

In [9]:
```
if re_weight:
    class_weight = {0: 1.309028344, 1: 0.472001959}
else:
    class_weight = None
```

In [10]:
```
model = Model(inputs=[sequence_1_input, sequence_2_input], \
        outputs=preds)
model.compile(loss='binary_crossentropy',
        optimizer='nadam',
        metrics=['acc'])
model.summary()
print(STAMP)
if LOAD_DATA:
    model.load_weights(bst_model_path)
early_stopping =EarlyStopping(monitor='val_loss', patience=5)
bst_model_path = STAMP + '.h5'
model_checkpoint = ModelCheckpoint(bst_model_path, save_best_only=True, sav
e_weights_only=True)

hist = model.fit([data_1_train, data_2_train], labels_train, \
        validation_data=([data_1_val, data_2_val], labels_val, weight_val),
 \
        epochs=50, batch_size=2048, shuffle=True, \
        class_weight=class_weight, callbacks=[model_checkpoint])

model.load_weights(bst_model_path)
bst_val_score = min(hist.history['val_loss'])
```

```
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected
to
==================================================================================================
input_1 (InputLayer)            (None, 30)           0

__________________________________________________________________________________________________
input_2 (InputLayer)            (None, 30)           0

__________________________________________________________________________________________________
embedding_1 (Embedding)         (None, 30, 300)      25655700    input_1[0]
[0]
                                                                 input_2[0]
[0]
__________________________________________________________________________________________________
lstm_1 (LSTM)                   (None, 265)          599960      embedding_
1[0][0]
                                                                 embedding_
1[1][0]
__________________________________________________________________________________________________
concatenate_1 (Concatenate)     (None, 530)          0           lstm_1[0]
[0]
                                                                 lstm_1[1]
[0]
__________________________________________________________________________________________________
dropout_1 (Dropout)             (None, 530)          0           concatenat
e_1[0][0]
__________________________________________________________________________________________________
batch_normalization_1 (BatchNor (None, 530)          2120        dropout_1
[0][0]
__________________________________________________________________________________________________
dense_1 (Dense)                 (None, 145)          76995       batch_norm
alization_1[0][0]
__________________________________________________________________________________________________
dropout_2 (Dropout)             (None, 145)          0           dense_1[0]
[0]
__________________________________________________________________________________________________
batch_normalization_2 (BatchNor (None, 145)          580         dropout_2
[0][0]
__________________________________________________________________________________________________
dense_2 (Dense)                 (None, 1)            146         batch_norm
alization_2[0][0]
==================================================================================================
Total params: 26,335,501
```

```
Trainable params: 678,451
Non-trainable params: 25,657,050
```

```
lstm
Train on 566006 samples, validate on 242574 samples
Epoch 1/50
566006/566006 [==============================] - 124s 218us/step - loss: 0.
4395 - acc: 0.6688 - val_loss: 0.3643 - val_acc: 0.6976
Epoch 2/50
566006/566006 [==============================] - 121s 215us/step - loss: 0.
3726 - acc: 0.6956 - val_loss: 0.3481 - val_acc: 0.7102
Epoch 3/50
566006/566006 [==============================] - 122s 216us/step - loss: 0.
3516 - acc: 0.7155 - val_loss: 0.3320 - val_acc: 0.7358
Epoch 4/50
566006/566006 [==============================] - 122s 216us/step - loss: 0.
3354 - acc: 0.7296 - val_loss: 0.3195 - val_acc: 0.7475
Epoch 5/50
566006/566006 [==============================] - 123s 217us/step - loss: 0.
3230 - acc: 0.7413 - val_loss: 0.3086 - val_acc: 0.7552
Epoch 6/50
566006/566006 [==============================] - 123s 217us/step - loss: 0.
3132 - acc: 0.7509 - val_loss: 0.3103 - val_acc: 0.7689
Epoch 7/50
566006/566006 [==============================] - 123s 217us/step - loss: 0.
3048 - acc: 0.7592 - val_loss: 0.2966 - val_acc: 0.7700
Epoch 8/50
566006/566006 [==============================] - 123s 217us/step - loss: 0.
2972 - acc: 0.7668 - val_loss: 0.2924 - val_acc: 0.7814
Epoch 9/50
566006/566006 [==============================] - 123s 217us/step - loss: 0.
2908 - acc: 0.7733 - val_loss: 0.2932 - val_acc: 0.7839
Epoch 10/50
566006/566006 [==============================] - 123s 217us/step - loss: 0.
2849 - acc: 0.7799 - val_loss: 0.2874 - val_acc: 0.7816
Epoch 11/50
566006/566006 [==============================] - 123s 218us/step - loss: 0.
2796 - acc: 0.7844 - val_loss: 0.2858 - val_acc: 0.7958
Epoch 12/50
566006/566006 [==============================] - 123s 218us/step - loss: 0.
2753 - acc: 0.7887 - val_loss: 0.2834 - val_acc: 0.7964
Epoch 13/50
566006/566006 [==============================] - 123s 217us/step - loss: 0.
2715 - acc: 0.7928 - val_loss: 0.2860 - val_acc: 0.8042
Epoch 14/50
566006/566006 [==============================] - 123s 217us/step - loss: 0.
2675 - acc: 0.7967 - val_loss: 0.2847 - val_acc: 0.8043
Epoch 15/50
566006/566006 [==============================] - 123s 218us/step - loss: 0.
2638 - acc: 0.7998 - val_loss: 0.2817 - val_acc: 0.8035
Epoch 16/50
566006/566006 [==============================] - 123s 218us/step - loss: 0.
2609 - acc: 0.8027 - val_loss: 0.2805 - val_acc: 0.8034
Epoch 17/50
566006/566006 [==============================] - 123s 218us/step - loss: 0.
2573 - acc: 0.8062 - val_loss: 0.2834 - val_acc: 0.8087
```

```
Epoch 18/50
566006/566006 [==============================] - 123s 218us/step - loss: 0.
2547 - acc: 0.8089 - val_loss: 0.2764 - val_acc: 0.8081
Epoch 19/50
566006/566006 [==============================] - 123s 218us/step - loss: 0.
2513 - acc: 0.8117 - val_loss: 0.2773 - val_acc: 0.8107
Epoch 20/50
566006/566006 [==============================] - 123s 218us/step - loss: 0.
2493 - acc: 0.8140 - val_loss: 0.2748 - val_acc: 0.8081
Epoch 21/50
566006/566006 [==============================] - 123s 218us/step - loss: 0.
2471 - acc: 0.8158 - val_loss: 0.2771 - val_acc: 0.8129
Epoch 22/50
566006/566006 [==============================] - 123s 218us/step - loss: 0.
2448 - acc: 0.8178 - val_loss: 0.2758 - val_acc: 0.8116
Epoch 23/50
566006/566006 [==============================] - 123s 218us/step - loss: 0.
2436 - acc: 0.8189 - val_loss: 0.2772 - val_acc: 0.8122
Epoch 24/50
566006/566006 [==============================] - 123s 218us/step - loss: 0.
2413 - acc: 0.8219 - val_loss: 0.2753 - val_acc: 0.8138
Epoch 25/50
566006/566006 [==============================] - 124s 218us/step - loss: 0.
2395 - acc: 0.8230 - val_loss: 0.2841 - val_acc: 0.8189
Epoch 26/50
566006/566006 [==============================] - 123s 218us/step - loss: 0.
2385 - acc: 0.8243 - val_loss: 0.2749 - val_acc: 0.8131
Epoch 27/50
566006/566006 [==============================] - 123s 217us/step - loss: 0.
2365 - acc: 0.8259 - val_loss: 0.2791 - val_acc: 0.8185
Epoch 28/50
566006/566006 [==============================] - 123s 218us/step - loss: 0.
2353 - acc: 0.8275 - val_loss: 0.2808 - val_acc: 0.8180
Epoch 29/50
566006/566006 [==============================] - 123s 218us/step - loss: 0.
2334 - acc: 0.8287 - val_loss: 0.2702 - val_acc: 0.8133
Epoch 30/50
566006/566006 [==============================] - 123s 218us/step - loss: 0.
2327 - acc: 0.8297 - val_loss: 0.2763 - val_acc: 0.8161
Epoch 31/50
566006/566006 [==============================] - 124s 218us/step - loss: 0.
2319 - acc: 0.8304 - val_loss: 0.2752 - val_acc: 0.8168
Epoch 32/50
566006/566006 [==============================] - 124s 218us/step - loss: 0.
2301 - acc: 0.8321 - val_loss: 0.2795 - val_acc: 0.8211
Epoch 33/50
566006/566006 [==============================] - 124s 219us/step - loss: 0.
2295 - acc: 0.8326 - val_loss: 0.2750 - val_acc: 0.8210
Epoch 34/50
566006/566006 [==============================] - 124s 218us/step - loss: 0.
2275 - acc: 0.8343 - val_loss: 0.2788 - val_acc: 0.8235
Epoch 35/50
566006/566006 [==============================] - 124s 218us/step - loss: 0.
2266 - acc: 0.8351 - val_loss: 0.2787 - val_acc: 0.8218
Epoch 36/50
566006/566006 [==============================] - 124s 218us/step - loss: 0.
2267 - acc: 0.8355 - val_loss: 0.2731 - val_acc: 0.8206
```

```
Epoch 37/50
566006/566006 [==============================] - 124s 219us/step - loss: 0.
2250 - acc: 0.8363 - val_loss: 0.2828 - val_acc: 0.8224
Epoch 38/50
566006/566006 [==============================] - 124s 219us/step - loss: 0.
2248 - acc: 0.8370 - val_loss: 0.2737 - val_acc: 0.8194
Epoch 39/50
566006/566006 [==============================] - 124s 218us/step - loss: 0.
2244 - acc: 0.8371 - val_loss: 0.2847 - val_acc: 0.8268
Epoch 40/50
566006/566006 [==============================] - 123s 218us/step - loss: 0.
2233 - acc: 0.8386 - val_loss: 0.2738 - val_acc: 0.8225
Epoch 41/50
566006/566006 [==============================] - 124s 219us/step - loss: 0.
2218 - acc: 0.8402 - val_loss: 0.2813 - val_acc: 0.8244
Epoch 42/50
566006/566006 [==============================] - 124s 219us/step - loss: 0.
2208 - acc: 0.8404 - val_loss: 0.2774 - val_acc: 0.8256
Epoch 43/50
566006/566006 [==============================] - 124s 218us/step - loss: 0.
2203 - acc: 0.8409 - val_loss: 0.2760 - val_acc: 0.8248
Epoch 44/50
566006/566006 [==============================] - 124s 219us/step - loss: 0.
2196 - acc: 0.8419 - val_loss: 0.2792 - val_acc: 0.8262
Epoch 45/50
566006/566006 [==============================] - 124s 218us/step - loss: 0.
2192 - acc: 0.8417 - val_loss: 0.2807 - val_acc: 0.8257
Epoch 46/50
566006/566006 [==============================] - 124s 219us/step - loss: 0.
2182 - acc: 0.8427 - val_loss: 0.2778 - val_acc: 0.8241
Epoch 47/50
566006/566006 [==============================] - 124s 219us/step - loss: 0.
2176 - acc: 0.8437 - val_loss: 0.2774 - val_acc: 0.8257
Epoch 48/50
566006/566006 [==============================] - 124s 219us/step - loss: 0.
2178 - acc: 0.8436 - val_loss: 0.2718 - val_acc: 0.8221
Epoch 49/50
566006/566006 [==============================] - 124s 219us/step - loss: 0.
2169 - acc: 0.8450 - val_loss: 0.2786 - val_acc: 0.8265
Epoch 50/50
566006/566006 [==============================] - 123s 218us/step - loss: 0.
2159 - acc: 0.8451 - val_loss: 0.2819 - val_acc: 0.8277
```

In [11]:
```
def add_attention_after_lstm(inputs, SINGLE_ATTENTION_VECTOR=False):
    a = Lambda(lambda x: KB.expand_dims(x, axis=1))(inputs)
    input_dim = int(a.shape[2])
    a = Permute((2, 1))(a)
    a = Reshape((input_dim, 1))(a) # this line is not useful. It's just to kno
w which dimension is what.
    a = Dense(1, activation='softmax')(a)
    if SINGLE_ATTENTION_VECTOR:
        a = Lambda(lambda x: KB.mean(x, axis=1))(a)
        a = RepeatVector(input_dim)(a)
    a_probs = Permute((2, 1))(a)
    a_probs = Lambda(lambda x: KB.squeeze(x, axis=1))(a_probs)
    outputs = KL.multiply([inputs, a_probs])
    return outputs
```

In [12]:
```
def add_attention_before_lstm(inputs, T, SINGLE_ATTENTION_VECTOR=True):
    a = inputs
    input_dim = int(a.shape[2])
    a = Permute((2, 1))(a)
    a = Reshape((input_dim, T))(a) # this line is not useful. It's just to kno
w which dimension is what.
    a = Dense(T, activation='softmax')(a)
    if SINGLE_ATTENTION_VECTOR:
        a = Lambda(lambda x: KB.mean(x, axis=1))(a)
        a = RepeatVector(input_dim)(a)
    a_probs = Permute((2, 1))(a)
    outputs = KL.multiply([inputs, a_probs])
    return outputs
```

In [13]:
```
embedding_layer_att_before = Embedding(nb_words, EMBEDDING_DIM, weights=[embed
ding_matrix],
                                input_length=MAX_SEQUENCE_LENGTH, trainable=False)
lstm_layer_att_before = LSTM(num_lstm, dropout=rate_drop_lstm, recurrent_dropo
ut=rate_drop_lstm)
sequence_1_input_att_before = Input(shape=(MAX_SEQUENCE_LENGTH,), dtype='int3
2')
embedded_sequences_1_att_before = embedding_layer_att_before(sequence_1_input_
att_before)
x1_att_before = add_attention_before_lstm(embedded_sequences_1_att_before, MAX
_SEQUENCE_LENGTH, SINGLE_ATTENTION_VECTOR=True)
x1_att_before = lstm_layer_att_before(x1_att_before)
sequence_2_input_att_before = Input(shape=(MAX_SEQUENCE_LENGTH,), dtype='int3
2')
embedded_sequences_2_att_before = embedding_layer_att_before(sequence_2_input_
att_before)
y1_att_before = add_attention_before_lstm(embedded_sequences_2_att_before, MAX
_SEQUENCE_LENGTH, SINGLE_ATTENTION_VECTOR=True)
y1_att_before = lstm_layer_att_before(embedded_sequences_2_att_before)
merged_att_before = concatenate([x1_att_before, y1_att_before])
merged_att_before = Dropout(rate_drop_dense)(merged_att_before)
merged_att_before = BatchNormalization()(merged_att_before)
merged_att_before = Dense(num_dense, activation=act)(merged_att_before)
merged_att_before = Dropout(rate_drop_dense)(merged_att_before)
merged_att_before = BatchNormalization()(merged_att_before)
preds_att_before = Dense(1, activation='sigmoid')(merged_att_before)
```

In [14]:
```
model_att_before = Model(inputs=[sequence_1_input_att_before, sequence_2_input
_att_before], outputs=preds_att_before)
model_att_before.compile(loss='binary_crossentropy', optimizer='nadam', metric
s=['acc'])
model_att_before.summary()
bst_model_path_att_before = 'att_before_' + STAMP + '.h5'
model_checkpoint_att_before = ModelCheckpoint(bst_model_path_att_before, save_
best_only=True, save_weights_only=True)
if LOAD_DATA:
    model_att_before.load_weights(bst_model_path_att_before)
hist_att_before = model_att_before.fit([data_1_train, data_2_train], labels_tr
ain, \
        validation_data=([data_1_val, data_2_val], labels_val, weight_val), \
        epochs=50, batch_size=2048, shuffle=True, \
        class_weight=class_weight, callbacks=[model_checkpoint_att_before])
model_att_before.load_weights(bst_model_path_att_before)
bst_val_score_att_before = min(hist_att_before.history['val_loss'])
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   Connected
to
==================================================================================
input_3 (InputLayer)         (None, 30)                0

_________________________________________________________________
embedding_2 (Embedding)      (None, 30, 300)           25655700  input_3[0]
[0]
                                                                 input_4[0]
[0]
_________________________________________________________________
permute_1 (Permute)          (None, 300, 30)           0         embedding_
2[0][0]
_________________________________________________________________
reshape_1 (Reshape)          (None, 300, 30)           0         permute_1
[0][0]
_________________________________________________________________
dense_3 (Dense)              (None, 300, 30)           930       reshape_1
[0][0]
_________________________________________________________________
lambda_1 (Lambda)            (None, 30)                0         dense_3[0]
[0]
_________________________________________________________________
repeat_vector_1 (RepeatVector)  (None, 300, 30)        0         lambda_1
[0][0]
_________________________________________________________________
permute_2 (Permute)          (None, 30, 300)           0         repeat_vec
tor_1[0][0]
_________________________________________________________________
input_4 (InputLayer)         (None, 30)                0

_________________________________________________________________
multiply_1 (Multiply)        (None, 30, 300)           0         embedding_
2[0][0]
                                                                 permute_2
[0][0]
_________________________________________________________________
lstm_2 (LSTM)                (None, 265)               599960    multiply_1
[0][0]
                                                                 embedding_
2[1][0]
_________________________________________________________________
concatenate_2 (Concatenate)  (None, 530)               0         lstm_2[0]
```

```
[0]
                                                                 lstm_2[1]
[0]
__________________________________________________________________________
____________________
dropout_3 (Dropout)             (None, 530)          0          concatenat
e_2[0][0]
__________________________________________________________________________
____________________
batch_normalization_3 (BatchNor (None, 530)          2120       dropout_3
[0][0]
__________________________________________________________________________
____________________
dense_5 (Dense)                 (None, 145)          76995      batch_norm
alization_3[0][0]
__________________________________________________________________________
____________________
dropout_4 (Dropout)             (None, 145)          0          dense_5[0]
[0]
__________________________________________________________________________
____________________
batch_normalization_4 (BatchNor (None, 145)          580        dropout_4
[0][0]
__________________________________________________________________________
____________________
dense_6 (Dense)                 (None, 1)            146        batch_norm
alization_4[0][0]
==========================================================================
======================
Total params: 26,336,431
Trainable params: 679,381
Non-trainable params: 25,657,050
__________________________________________________________________________
____________________
Train on 566006 samples, validate on 242574 samples
Epoch 1/50
566006/566006 [==============================] - 151s 267us/step - loss: 0.
4538 - acc: 0.6585 - val_loss: 0.3901 - val_acc: 0.6979
Epoch 2/50
566006/566006 [==============================] - 173s 306us/step - loss: 0.
3881 - acc: 0.6855 - val_loss: 0.3642 - val_acc: 0.6993
Epoch 3/50
566006/566006 [==============================] - 184s 326us/step - loss: 0.
3714 - acc: 0.6985 - val_loss: 0.3593 - val_acc: 0.7187
Epoch 4/50
566006/566006 [==============================] - 172s 303us/step - loss: 0.
3605 - acc: 0.7072 - val_loss: 0.3429 - val_acc: 0.7060
Epoch 5/50
566006/566006 [==============================] - 148s 262us/step - loss: 0.
3517 - acc: 0.7161 - val_loss: 0.3344 - val_acc: 0.7240
Epoch 6/50
566006/566006 [==============================] - 149s 263us/step - loss: 0.
3452 - acc: 0.7222 - val_loss: 0.3322 - val_acc: 0.7377
Epoch 7/50
566006/566006 [==============================] - 149s 263us/step - loss: 0.
3394 - acc: 0.7278 - val_loss: 0.3268 - val_acc: 0.7448
Epoch 8/50
```

```
566006/566006 [==============================] - 149s 263us/step - loss: 0.
3349 - acc: 0.7327 - val_loss: 0.3240 - val_acc: 0.7500
Epoch 9/50
566006/566006 [==============================] - 149s 263us/step - loss: 0.
3306 - acc: 0.7369 - val_loss: 0.3177 - val_acc: 0.7496
Epoch 10/50
566006/566006 [==============================] - 149s 263us/step - loss: 0.
3264 - acc: 0.7415 - val_loss: 0.3153 - val_acc: 0.7414
Epoch 11/50
566006/566006 [==============================] - 148s 262us/step - loss: 0.
3234 - acc: 0.7441 - val_loss: 0.3173 - val_acc: 0.7628
Epoch 12/50
566006/566006 [==============================] - 148s 262us/step - loss: 0.
3198 - acc: 0.7482 - val_loss: 0.3111 - val_acc: 0.7603
Epoch 13/50
566006/566006 [==============================] - 148s 262us/step - loss: 0.
3172 - acc: 0.7509 - val_loss: 0.3125 - val_acc: 0.7683
Epoch 14/50
566006/566006 [==============================] - 148s 262us/step - loss: 0.
3142 - acc: 0.7542 - val_loss: 0.3061 - val_acc: 0.7470
Epoch 15/50
566006/566006 [==============================] - 148s 262us/step - loss: 0.
3114 - acc: 0.7567 - val_loss: 0.3112 - val_acc: 0.7696
Epoch 16/50
566006/566006 [==============================] - 149s 263us/step - loss: 0.
3087 - acc: 0.7596 - val_loss: 0.3073 - val_acc: 0.7652
Epoch 17/50
566006/566006 [==============================] - 149s 263us/step - loss: 0.
3062 - acc: 0.7620 - val_loss: 0.3047 - val_acc: 0.7721
Epoch 18/50
566006/566006 [==============================] - 149s 263us/step - loss: 0.
3039 - acc: 0.7644 - val_loss: 0.3041 - val_acc: 0.7741
Epoch 19/50
566006/566006 [==============================] - 149s 263us/step - loss: 0.
3028 - acc: 0.7650 - val_loss: 0.3019 - val_acc: 0.7706
Epoch 20/50
566006/566006 [==============================] - 149s 264us/step - loss: 0.
3006 - acc: 0.7676 - val_loss: 0.3017 - val_acc: 0.7732
Epoch 21/50
566006/566006 [==============================] - 149s 263us/step - loss: 0.
2991 - acc: 0.7693 - val_loss: 0.3030 - val_acc: 0.7762
Epoch 22/50
566006/566006 [==============================] - 149s 263us/step - loss: 0.
2965 - acc: 0.7720 - val_loss: 0.2969 - val_acc: 0.7710
Epoch 23/50
566006/566006 [==============================] - 148s 262us/step - loss: 0.
2951 - acc: 0.7727 - val_loss: 0.2978 - val_acc: 0.7689
Epoch 24/50
566006/566006 [==============================] - 148s 261us/step - loss: 0.
2935 - acc: 0.7748 - val_loss: 0.2948 - val_acc: 0.7765
Epoch 25/50
566006/566006 [==============================] - 148s 261us/step - loss: 0.
2928 - acc: 0.7754 - val_loss: 0.2966 - val_acc: 0.7805
Epoch 26/50
566006/566006 [==============================] - 149s 262us/step - loss: 0.
2906 - acc: 0.7780 - val_loss: 0.2965 - val_acc: 0.7800
Epoch 27/50
```

```
566006/566006 [==============================] - 148s 262us/step - loss: 0.
2893 - acc: 0.7794 - val_loss: 0.2970 - val_acc: 0.7876
Epoch 28/50
566006/566006 [==============================] - 148s 261us/step - loss: 0.
2880 - acc: 0.7805 - val_loss: 0.2914 - val_acc: 0.7789
Epoch 29/50
566006/566006 [==============================] - 148s 262us/step - loss: 0.
2871 - acc: 0.7819 - val_loss: 0.2941 - val_acc: 0.7836
Epoch 30/50
566006/566006 [==============================] - 148s 262us/step - loss: 0.
2854 - acc: 0.7831 - val_loss: 0.2952 - val_acc: 0.7853
Epoch 31/50
566006/566006 [==============================] - 148s 262us/step - loss: 0.
2847 - acc: 0.7834 - val_loss: 0.2944 - val_acc: 0.7897
Epoch 32/50
566006/566006 [==============================] - 148s 262us/step - loss: 0.
2837 - acc: 0.7847 - val_loss: 0.2917 - val_acc: 0.7826
Epoch 33/50
566006/566006 [==============================] - 148s 261us/step - loss: 0.
2823 - acc: 0.7860 - val_loss: 0.2927 - val_acc: 0.7870
Epoch 34/50
566006/566006 [==============================] - 148s 262us/step - loss: 0.
2814 - acc: 0.7874 - val_loss: 0.2921 - val_acc: 0.7827
Epoch 35/50
566006/566006 [==============================] - 148s 262us/step - loss: 0.
2798 - acc: 0.7882 - val_loss: 0.2903 - val_acc: 0.7873
Epoch 36/50
566006/566006 [==============================] - 148s 262us/step - loss: 0.
2798 - acc: 0.7889 - val_loss: 0.2895 - val_acc: 0.7833
Epoch 37/50
566006/566006 [==============================] - 148s 261us/step - loss: 0.
2786 - acc: 0.7901 - val_loss: 0.2891 - val_acc: 0.7873
Epoch 38/50
566006/566006 [==============================] - 148s 261us/step - loss: 0.
2775 - acc: 0.7911 - val_loss: 0.2899 - val_acc: 0.7897
Epoch 39/50
566006/566006 [==============================] - 148s 262us/step - loss: 0.
2765 - acc: 0.7918 - val_loss: 0.2878 - val_acc: 0.7871
Epoch 40/50
566006/566006 [==============================] - 148s 261us/step - loss: 0.
2755 - acc: 0.7928 - val_loss: 0.2884 - val_acc: 0.7892
Epoch 41/50
566006/566006 [==============================] - 148s 262us/step - loss: 0.
2753 - acc: 0.7927 - val_loss: 0.2868 - val_acc: 0.7906
Epoch 42/50
566006/566006 [==============================] - 148s 261us/step - loss: 0.
2738 - acc: 0.7941 - val_loss: 0.2894 - val_acc: 0.7953
Epoch 43/50
566006/566006 [==============================] - 149s 262us/step - loss: 0.
2740 - acc: 0.7942 - val_loss: 0.2920 - val_acc: 0.7961
Epoch 44/50
566006/566006 [==============================] - 149s 262us/step - loss: 0.
2730 - acc: 0.7954 - val_loss: 0.2844 - val_acc: 0.7860
Epoch 45/50
566006/566006 [==============================] - 148s 262us/step - loss: 0.
2722 - acc: 0.7961 - val_loss: 0.2910 - val_acc: 0.7963
Epoch 46/50
```

```
566006/566006 [==============================] - 149s 263us/step - loss: 0.
2715 - acc: 0.7964 - val_loss: 0.2880 - val_acc: 0.7951
Epoch 47/50
566006/566006 [==============================] - 148s 261us/step - loss: 0.
2707 - acc: 0.7974 - val_loss: 0.2894 - val_acc: 0.7981
Epoch 48/50
566006/566006 [==============================] - 148s 262us/step - loss: 0.
2704 - acc: 0.7977 - val_loss: 0.2869 - val_acc: 0.7960
Epoch 49/50
566006/566006 [==============================] - 148s 262us/step - loss: 0.
2695 - acc: 0.7989 - val_loss: 0.2838 - val_acc: 0.7889
Epoch 50/50
566006/566006 [==============================] - 148s 262us/step - loss: 0.
2689 - acc: 0.7987 - val_loss: 0.2878 - val_acc: 0.8000
```

In [15]:
```
embedding_layer_att_after = Embedding(nb_words, EMBEDDING_DIM, weights=[embedd
ing_matrix],
                                      input_length=MAX_SEQUENCE_LENGTH, trainable=False)
lstm_layer_att_after = LSTM(num_lstm, dropout=rate_drop_lstm, recurrent_dropou
t=rate_drop_lstm)
sequence_1_input_att_after = Input(shape=(MAX_SEQUENCE_LENGTH,), dtype='int32'
)
embedded_sequences_1_att_after = embedding_layer_att_after(sequence_1_input_at
t_after)
x1_att_after = lstm_layer_att_after(embedded_sequences_1_att_after)
x1_att_after = add_attention_after_lstm(x1_att_after, SINGLE_ATTENTION_VECTOR=
False)
sequence_2_input_att_after = Input(shape=(MAX_SEQUENCE_LENGTH,), dtype='int32'
)
embedded_sequences_2_att_after = embedding_layer_att_after(sequence_2_input_at
t_after)
y1_att_after = lstm_layer_att_after(embedded_sequences_2_att_after)
y1_att_after = add_attention_after_lstm(y1_att_after, SINGLE_ATTENTION_VECTOR=
False)
merged_att_after = concatenate([x1_att_after, y1_att_after])
merged_att_after = Dropout(rate_drop_dense)(merged_att_after)
merged_att_after = BatchNormalization()(merged_att_after)
merged_att_after = Dense(num_dense, activation=act)(merged_att_after)
merged_att_after = Dropout(rate_drop_dense)(merged_att_after)
merged_att_after = BatchNormalization()(merged_att_after)
preds_att_after = Dense(1, activation='sigmoid')(merged_att_after)
```

In [16]:
```
model_att_after = Model(inputs=[sequence_1_input_att_after, sequence_2_input_a
tt_after], outputs=preds_att_after)
model_att_after.compile(loss='binary_crossentropy', optimizer='nadam', metrics
=['acc'])
model_att_after.summary()
bst_model_path_att_after = 'att_after_' + STAMP + '.h5'
model_checkpoint_att_after = ModelCheckpoint(bst_model_path_att_after, save_be
st_only=True, save_weights_only=True)
if LOAD_DATA:
    model_att_after.load_weights(bst_model_path_att_after)
hist_att_after = model_att_after.fit([data_1_train, data_2_train], labels_trai
n, \
        validation_data=([data_1_val, data_2_val], labels_val, weight_val), \
        epochs=50, batch_size=2048, shuffle=True, \
        class_weight=class_weight, callbacks=[model_checkpoint_att_after])
model_att_after.load_weights(bst_model_path_att_after)
bst_val_score_att_after = min(hist_att_after.history['val_loss'])
```

```
____________________________________________________________________________________________________
Layer (type)                 Output Shape              Param #     Connected
to
====================================================================================================
input_5 (InputLayer)         (None, 30)                0

____________________________________________________________________________________________________
input_6 (InputLayer)         (None, 30)                0

____________________________________________________________________________________________________
embedding_3 (Embedding)      (None, 30, 300)           25655700    input_5[0]
[0]
                                                                   input_6[0]
[0]
____________________________________________________________________________________________________
lstm_3 (LSTM)                (None, 265)               599960      embedding_
3[0][0]
                                                                   embedding_
3[1][0]
____________________________________________________________________________________________________
lambda_3 (Lambda)            (None, 1, 265)            0           lstm_3[0]
[0]
____________________________________________________________________________________________________
lambda_5 (Lambda)            (None, 1, 265)            0           lstm_3[1]
[0]
____________________________________________________________________________________________________
permute_5 (Permute)          (None, 265, 1)            0           lambda_3
[0][0]
____________________________________________________________________________________________________
permute_7 (Permute)          (None, 265, 1)            0           lambda_5
[0][0]
____________________________________________________________________________________________________
reshape_3 (Reshape)          (None, 265, 1)            0           permute_5
[0][0]
____________________________________________________________________________________________________
reshape_4 (Reshape)          (None, 265, 1)            0           permute_7
[0][0]
____________________________________________________________________________________________________
dense_7 (Dense)              (None, 265, 1)            2           reshape_3
[0][0]
____________________________________________________________________________________________________
dense_8 (Dense)              (None, 265, 1)            2           reshape_4
[0][0]
____________________________________________________________________________________________________
```

permute_6 (Permute)	(None, 1, 265)	0	dense_7[0][0]
permute_8 (Permute)	(None, 1, 265)	0	dense_8[0][0]
lambda_4 (Lambda)	(None, 265)	0	permute_6[0][0]
lambda_6 (Lambda)	(None, 265)	0	permute_8[0][0]
multiply_3 (Multiply)	(None, 265)	0	lstm_3[0][0] lambda_4[0][0]
multiply_4 (Multiply)	(None, 265)	0	lstm_3[1][0] lambda_6[0][0]
concatenate_3 (Concatenate)	(None, 530)	0	multiply_3[0][0] multiply_4[0][0]
dropout_5 (Dropout)	(None, 530)	0	concatenate_3[0][0]
batch_normalization_5 (BatchNor	(None, 530)	2120	dropout_5[0][0]
dense_9 (Dense)	(None, 145)	76995	batch_normalization_5[0][0]
dropout_6 (Dropout)	(None, 145)	0	dense_9[0][0]
batch_normalization_6 (BatchNor	(None, 145)	580	dropout_6[0][0]
dense_10 (Dense)	(None, 1)	146	batch_normalization_6[0][0]

```
=================================================================
======================
Total params: 26,335,505
Trainable params: 678,455
Non-trainable params: 25,657,050

_________________________________________________________________
_____________________
Train on 566006 samples, validate on 242574 samples
Epoch 1/50
566006/566006 [==============================] - 128s 227us/step - loss: 0.
4397 - acc: 0.6695 - val_loss: 0.3685 - val_acc: 0.7189
Epoch 2/50
566006/566006 [==============================] - 126s 222us/step - loss: 0.
3706 - acc: 0.6986 - val_loss: 0.3455 - val_acc: 0.7106
Epoch 3/50
566006/566006 [==============================] - 125s 222us/step - loss: 0.
3497 - acc: 0.7161 - val_loss: 0.3385 - val_acc: 0.7361
Epoch 4/50
566006/566006 [==============================] - 125s 222us/step - loss: 0.
3340 - acc: 0.7302 - val_loss: 0.3213 - val_acc: 0.7507
Epoch 5/50
566006/566006 [==============================] - 126s 222us/step - loss: 0.
3218 - acc: 0.7420 - val_loss: 0.3129 - val_acc: 0.7648
Epoch 6/50
566006/566006 [==============================] - 126s 222us/step - loss: 0.
3126 - acc: 0.7511 - val_loss: 0.3013 - val_acc: 0.7669
Epoch 7/50
566006/566006 [==============================] - 126s 222us/step - loss: 0.
3040 - acc: 0.7600 - val_loss: 0.2993 - val_acc: 0.7708
Epoch 8/50
566006/566006 [==============================] - 126s 222us/step - loss: 0.
2971 - acc: 0.7661 - val_loss: 0.2948 - val_acc: 0.7887
Epoch 9/50
566006/566006 [==============================] - 126s 222us/step - loss: 0.
2906 - acc: 0.7727 - val_loss: 0.2865 - val_acc: 0.7813
Epoch 10/50
566006/566006 [==============================] - 126s 222us/step - loss: 0.
2847 - acc: 0.7791 - val_loss: 0.2880 - val_acc: 0.7905
Epoch 11/50
566006/566006 [==============================] - 126s 222us/step - loss: 0.
2795 - acc: 0.7840 - val_loss: 0.2888 - val_acc: 0.8002
Epoch 12/50
566006/566006 [==============================] - 126s 222us/step - loss: 0.
2747 - acc: 0.7886 - val_loss: 0.2806 - val_acc: 0.7989
Epoch 13/50
566006/566006 [==============================] - 125s 222us/step - loss: 0.
2704 - acc: 0.7932 - val_loss: 0.2813 - val_acc: 0.8016
Epoch 14/50
566006/566006 [==============================] - 126s 222us/step - loss: 0.
2665 - acc: 0.7970 - val_loss: 0.2802 - val_acc: 0.8041
Epoch 15/50
566006/566006 [==============================] - 126s 222us/step - loss: 0.
2628 - acc: 0.8008 - val_loss: 0.2850 - val_acc: 0.8078
Epoch 16/50
566006/566006 [==============================] - 126s 222us/step - loss: 0.
2593 - acc: 0.8041 - val_loss: 0.2782 - val_acc: 0.8044
Epoch 17/50
```

```
566006/566006 [==============================] - 126s 222us/step - loss: 0.
2562 - acc: 0.8069 - val_loss: 0.2799 - val_acc: 0.8113
Epoch 18/50
566006/566006 [==============================] - 126s 222us/step - loss: 0.
2535 - acc: 0.8093 - val_loss: 0.2770 - val_acc: 0.8024
Epoch 19/50
566006/566006 [==============================] - 125s 222us/step - loss: 0.
2506 - acc: 0.8120 - val_loss: 0.2770 - val_acc: 0.8114
Epoch 20/50
566006/566006 [==============================] - 126s 222us/step - loss: 0.
2482 - acc: 0.8146 - val_loss: 0.2805 - val_acc: 0.8153
Epoch 21/50
566006/566006 [==============================] - 126s 222us/step - loss: 0.
2467 - acc: 0.8157 - val_loss: 0.2833 - val_acc: 0.8154
Epoch 22/50
566006/566006 [==============================] - 126s 222us/step - loss: 0.
2442 - acc: 0.8183 - val_loss: 0.2794 - val_acc: 0.8173
Epoch 23/50
566006/566006 [==============================] - 125s 221us/step - loss: 0.
2428 - acc: 0.8200 - val_loss: 0.2805 - val_acc: 0.8167
Epoch 24/50
566006/566006 [==============================] - 124s 220us/step - loss: 0.
2406 - acc: 0.8215 - val_loss: 0.2759 - val_acc: 0.8159
Epoch 25/50
566006/566006 [==============================] - 125s 221us/step - loss: 0.
2393 - acc: 0.8234 - val_loss: 0.2812 - val_acc: 0.8205
Epoch 26/50
566006/566006 [==============================] - 142s 250us/step - loss: 0.
2371 - acc: 0.8248 - val_loss: 0.2772 - val_acc: 0.8166
Epoch 27/50
566006/566006 [==============================] - 188s 333us/step - loss: 0.
2356 - acc: 0.8267 - val_loss: 0.2801 - val_acc: 0.8193
Epoch 28/50
566006/566006 [==============================] - 151s 267us/step - loss: 0.
2343 - acc: 0.8279 - val_loss: 0.2768 - val_acc: 0.8182
Epoch 29/50
566006/566006 [==============================] - 153s 270us/step - loss: 0.
2332 - acc: 0.8287 - val_loss: 0.2738 - val_acc: 0.8187
Epoch 30/50
566006/566006 [==============================] - 153s 271us/step - loss: 0.
2321 - acc: 0.8297 - val_loss: 0.2843 - val_acc: 0.8248
Epoch 31/50
566006/566006 [==============================] - 153s 270us/step - loss: 0.
2301 - acc: 0.8314 - val_loss: 0.2779 - val_acc: 0.8214
Epoch 32/50
566006/566006 [==============================] - 155s 273us/step - loss: 0.
2294 - acc: 0.8323 - val_loss: 0.2760 - val_acc: 0.8206
Epoch 33/50
566006/566006 [==============================] - 153s 270us/step - loss: 0.
2289 - acc: 0.8330 - val_loss: 0.2769 - val_acc: 0.8220
Epoch 34/50
566006/566006 [==============================] - 150s 265us/step - loss: 0.
2270 - acc: 0.8343 - val_loss: 0.2778 - val_acc: 0.8226
Epoch 35/50
566006/566006 [==============================] - 152s 269us/step - loss: 0.
2259 - acc: 0.8354 - val_loss: 0.2751 - val_acc: 0.8231
Epoch 36/50
```

```
566006/566006 [==============================] - 157s 277us/step - loss: 0.
2253 - acc: 0.8364 - val_loss: 0.2775 - val_acc: 0.8244
Epoch 37/50
566006/566006 [==============================] - 154s 272us/step - loss: 0.
2242 - acc: 0.8371 - val_loss: 0.2737 - val_acc: 0.8224
Epoch 38/50
566006/566006 [==============================] - 153s 270us/step - loss: 0.
2240 - acc: 0.8376 - val_loss: 0.2779 - val_acc: 0.8217
Epoch 39/50
566006/566006 [==============================] - 151s 267us/step - loss: 0.
2232 - acc: 0.8379 - val_loss: 0.2809 - val_acc: 0.8240
Epoch 40/50
566006/566006 [==============================] - 157s 277us/step - loss: 0.
2217 - acc: 0.8393 - val_loss: 0.2823 - val_acc: 0.8278
Epoch 41/50
566006/566006 [==============================] - 130s 229us/step - loss: 0.
2214 - acc: 0.8399 - val_loss: 0.2799 - val_acc: 0.8243
Epoch 42/50
566006/566006 [==============================] - 125s 221us/step - loss: 0.
2204 - acc: 0.8410 - val_loss: 0.2841 - val_acc: 0.8271
Epoch 43/50
566006/566006 [==============================] - 125s 221us/step - loss: 0.
2195 - acc: 0.8415 - val_loss: 0.2750 - val_acc: 0.8242
Epoch 44/50
566006/566006 [==============================] - 126s 222us/step - loss: 0.
2194 - acc: 0.8415 - val_loss: 0.2755 - val_acc: 0.8260
Epoch 45/50
566006/566006 [==============================] - 126s 222us/step - loss: 0.
2184 - acc: 0.8427 - val_loss: 0.2771 - val_acc: 0.8239
Epoch 46/50
566006/566006 [==============================] - 126s 222us/step - loss: 0.
2176 - acc: 0.8434 - val_loss: 0.2789 - val_acc: 0.8277
Epoch 47/50
566006/566006 [==============================] - 126s 222us/step - loss: 0.
2177 - acc: 0.8430 - val_loss: 0.2748 - val_acc: 0.8272
Epoch 48/50
566006/566006 [==============================] - 126s 222us/step - loss: 0.
2167 - acc: 0.8440 - val_loss: 0.2734 - val_acc: 0.8253
Epoch 49/50
566006/566006 [==============================] - 125s 222us/step - loss: 0.
2156 - acc: 0.8448 - val_loss: 0.2840 - val_acc: 0.8311
Epoch 50/50
566006/566006 [==============================] - 126s 222us/step - loss: 0.
2161 - acc: 0.8448 - val_loss: 0.2822 - val_acc: 0.8280
```