```
# pip install keras
# pip install tensorflow (backend)
# pip install pydot
# pip install graphviz

from keras.preprocessing import sequence
from keras.models import Sequential, Model
from keras.layers import Dense, Embedding, Flatten, Input, Activation
from keras.datasets import imdb
import keras.backend as K
from keras.callbacks import EarlyStopping

max_features = 20000
maxlen = 80  # cut texts after this number of words (among top max_features most
common words)
batch_size = 32
embedding_dim = 128

print('Loading data...')
(x_train, y_train), (x_test, y_test) = imdb.load_data(num_words=max_features) # Replace
this and create your own data loader and pre-processing for the given data on e-dimension
print(len(x_train), 'train sequences')
print(len(x_test), 'test sequences')

print('Pad sequences (samples x time)')
x_train = sequence.pad_sequences(x_train, maxlen=maxlen)
x_test = sequence.pad_sequences(x_test, maxlen=maxlen)
print('x_train shape:', x_train.shape)
print('x_test shape:', x_test.shape)

print('Build model...')

inputs = Input(shape=(80,))
# keras.layers.Embedding(input_dim, output_dim, embeddings_initializer='uniform',
embeddings_regularizer=None, activity_regularizer=None, embeddings_constraint=None,
mask_zero=False, input_length=None)
out1 = Embedding(max_features, embedding_dim,input_shape=(maxlen,),trainable=True)
(inputs) #batch, maxlen, 128

out2 = Flatten()(out1)
out3 = Dense(128)(out2)
out4 = Dense(1, activation='sigmoid')(out3)

model = Model(inputs = inputs, outputs = out4)
print(model.summary())
```

```
from keras.utils import plot_model
plot_model(model, to_file='model.png')

es = EarlyStopping(monitor='val_loss', mode='min', patience=10)

# try using different optimizers and different optimizer configs
model.compile(loss='binary_crossentropy',
              optimizer='adam',
              metrics=['accuracy'])
print('Train...')
model.fit(x_train, y_train,
          batch_size=batch_size,
          epochs=15,
          validation_split=0.1, callbacks=[es])
score, acc = model.evaluate(x_test, y_test,
                            batch_size=batch_size)
print('Test score:', score)
print('Test accuracy:', acc)
```