In [1]:

```
from __future__ import print_function
import numpy as np
import os
import scipy
from six.moves import cPickle
```

In [2]:

```
save_dir = 'models' # directory to store models
```

In [3]:

```


import spacy
import en_core_web_sm


# In[12]:
import spacy
nlp = en_core_web_sm.load()
```

In [4]:

```
#import gensim library
import gensim
from gensim.models.doc2vec import LabeledSentence

#load the doc2vec model
print("loading doc2Vec model...")
d2v_model = gensim.models.doc2vec.Doc2Vec.load('models\\doc2vec.w2v')

print("model loaded!")
```

```
loading doc2Vec model...
model loaded!
```

In [5]:

```
#load vocabulary
print("loading vocabulary...")
vocab_file = os.path.join(save_dir, "words_vocab.pkl")

with open(os.path.join(save_dir, 'words_vocab.pkl'), 'rb') as f:
        words, vocab, vocabulary_inv = cPickle.load(f)

vocab_size = len(words)
print("vocabulary loaded !")
```

```
loading vocabulary...
vocabulary loaded !
```

In [6]:

```
from keras.models import load_model
# load the keras models
print("loading word prediction model...")
model = load_model(save_dir + "\\" + 'my_model_gen_sentences_lstm.final.hdf5')
print("model loaded!")
print("loading sentence selection model...")
model_sequence = load_model(save_dir + "\\" + 'my_model_sequence_lstm.final2.hdf5')
print("model loaded!")
```

```
Using TensorFlow backend.
C:\Users\mayan\AppData\Roaming\Python\Python37\site-packages\tensorflow\py
thon\framework\dtypes.py:516: FutureWarning: Passing (type, 1) or '1type'
as a synonym of type is deprecated; in a future version of numpy, it will
be understood as (type, (1,)) / '(1,)type'.
  _np_qint8 = np.dtype([("qint8", np.int8, 1)])
C:\Users\mayan\AppData\Roaming\Python\Python37\site-packages\tensorflow\py
thon\framework\dtypes.py:517: FutureWarning: Passing (type, 1) or '1type'
as a synonym of type is deprecated; in a future version of numpy, it will
be understood as (type, (1,)) / '(1,)type'.
  _np_quint8 = np.dtype([("quint8", np.uint8, 1)])
C:\Users\mayan\AppData\Roaming\Python\Python37\site-packages\tensorflow\py
thon\framework\dtypes.py:518: FutureWarning: Passing (type, 1) or '1type'
as a synonym of type is deprecated; in a future version of numpy, it will
be understood as (type, (1,)) / '(1,)type'.
  _np_qint16 = np.dtype([("qint16", np.int16, 1)])
C:\Users\mayan\AppData\Roaming\Python\Python37\site-packages\tensorflow\py
thon\framework\dtypes.py:519: FutureWarning: Passing (type, 1) or '1type'
as a synonym of type is deprecated; in a future version of numpy, it will
be understood as (type, (1,)) / '(1,)type'.
  _np_quint16 = np.dtype([("quint16", np.uint16, 1)])
C:\Users\mayan\AppData\Roaming\Python\Python37\site-packages\tensorflow\py
thon\framework\dtypes.py:520: FutureWarning: Passing (type, 1) or '1type'
as a synonym of type is deprecated; in a future version of numpy, it will
be understood as (type, (1,)) / '(1,)type'.
  _np_qint32 = np.dtype([("qint32", np.int32, 1)])
C:\Users\mayan\AppData\Roaming\Python\Python37\site-packages\tensorflow\py
thon\framework\dtypes.py:525: FutureWarning: Passing (type, 1) or '1type'
as a synonym of type is deprecated; in a future version of numpy, it will
be understood as (type, (1,)) / '(1,)type'.
  np_resource = np.dtype([("resource", np.ubyte, 1)])
C:\Users\mayan\AppData\Roaming\Python\Python37\site-packages\tensorboard\c
ompat\tensorflow_stub\dtypes.py:541: FutureWarning: Passing (type, 1) or
'1type' as a synonym of type is deprecated; in a future version of numpy,
it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint8 = np.dtype([("qint8", np.int8, 1)])
C:\Users\mayan\AppData\Roaming\Python\Python37\site-packages\tensorboard\c
ompat\tensorflow_stub\dtypes.py:542: FutureWarning: Passing (type, 1) or
'1type' as a synonym of type is deprecated; in a future version of numpy,
it will be understood as (type, (1,)) / '(1,)type'.
  _np_quint8 = np.dtype([("quint8", np.uint8, 1)])
C:\Users\mayan\AppData\Roaming\Python\Python37\site-packages\tensorboard\c
ompat\tensorflow_stub\dtypes.py:543: FutureWarning: Passing (type, 1) or
'1type' as a synonym of type is deprecated; in a future version of numpy,
it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint16 = np.dtype([("qint16", np.int16, 1)])
C:\Users\mayan\AppData\Roaming\Python\Python37\site-packages\tensorboard\c
ompat\tensorflow_stub\dtypes.py:544: FutureWarning: Passing (type, 1) or
'1type' as a synonym of type is deprecated; in a future version of numpy,
```

```
it will be understood as (type, (1,)) / '(1,)type'.
  _np_quint16 = np.dtype([("quint16", np.uint16, 1)])
C:\Users\mayan\AppData\Roaming\Python\Python37\site-packages\tensorboard\c
ompat\tensorflow_stub\dtypes.py:545: FutureWarning: Passing (type, 1) or
'1type' as a synonym of type is deprecated; in a future version of numpy,
it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint32 = np.dtype([("qint32", np.int32, 1)])
C:\Users\mayan\AppData\Roaming\Python\Python37\site-packages\tensorboard\c
ompat\tensorflow_stub\dtypes.py:550: FutureWarning: Passing (type, 1) or
'1type' as a synonym of type is deprecated; in a future version of numpy,
it will be understood as (type, (1,)) / '(1,)type'.
  np_resource = np.dtype([("resource", np.ubyte, 1)])
```

```
loading word prediction model...
WARNING:tensorflow:Large dropout rate: 0.6 (>0.5). In TensorFlow 2.x, dropou
t() uses dropout rate instead of keep_prob. Please ensure that this is inten
ded.
WARNING:tensorflow:From c:\users\mayan\appdata\local\programs\python\python3
7\lib\site-packages\keras\backend\tensorflow_backend.py:422: The name tf.glo
bal_variables is deprecated. Please use tf.compat.v1.global_variables instea
d.

model loaded!
loading sentence selection model...
model loaded!
```

In [8]:

```
def sample(preds, temperature=1.0):
    # helper function to sample an index from a probability array
    preds = np.asarray(preds).astype('float64')
    preds = np.log(preds) / temperature
    exp_preds = np.exp(preds)
    preds = exp_preds / np.sum(exp_preds)
    probas = np.random.multinomial(1, preds, 1)
    return np.argmax(probas)
```

In [11]:

```
import re
def untokenize(words):
    """
    Untokenizing a text undoes the tokenizing operation, restoring
    punctuation and spaces to the places that people expect them to be.
    Ideally, `untokenize(tokenize(text))` should be identical to `text`,
    except for line breaks.
    """
    text = ' '.join(words)
    step1 = text.replace("`` ", '"').replace(" ''", '"').replace('. . .',  '...')
    step2 = step1.replace(" ( ", " (").replace(" ) ", ") ")
    step3 = re.sub(r' ([.,:;?!%]+)([ \'"`])', r"\1\2", step2)
    step4 = re.sub(r' ([.,:;?!%]+)$', r"\1", step3)
    step5 = step4.replace(" '", "'").replace(" n't", "n't").replace(
         "can not", "cannot")
    step6 = step5.replace(" ` ", " '")
    return step6.strip()
```

In [12]:

```
def create_seed(seed_sentences,nb_words_in_seq=20, verbose=False):
    #initiate sentences
    generated = ''
    sentence = []

    #fill the sentence with a default word
    for i in range (nb_words_in_seq):
        sentence.append("le")
    import nltk
    from nltk import word_tokenize

    seed = word_tokenize(seed_sentences)

    if verbose == True : print("seed: ",seed)

    for i in range(len(sentence)):
        sentence[nb_words_in_seq-i-1]=seed[len(seed)-i-1]
        #print(i, sentence)

    generated  = untokenize(sentence)

    if verbose == True : print('Generating text with the following seed: \n"' + generated +

    return [generated, sentence]
```

In [13]:

```
def generate_phrase(sentence, max_words = 50, nb_words_in_seq=20, temperature=1, verbose =
    generated = ""
    words_number = max_words - 1
    ponctuation = [".","?","!",":","…"]
    seq_length = nb_words_in_seq
    #sentence = []
    is_punct = False

    for i in range(words_number):
        #create the vector
        x = np.zeros((1, seq_length, vocab_size))
        for t, word in enumerate(sentence):
            #print(t, word, vocab[word])
            x[0, nb_words_in_seq-len(sentence)+t, vocab[word]] = 1.
        #print(x.shape)

        preds = model.predict(x, verbose=0)[0]
        next_index = sample(preds, temperature)
        next_word = vocabulary_inv[next_index]

        if verbose == True:
            predv = np.array(preds)
            #arr = np.array([1, 3, 2, 4, 5])
            wi = predv.argsort()[-3:][::-1]
            print("potential next words: ", vocabulary_inv[wi[0]], vocabulary_inv[wi[1]], v

        if is_punct == False:
            if next_word in ponctuation:
                is_punct = True
            generated += " " + next_word
            sentence = sentence[1:] + [next_word]

    return(generated, sentence)
```

In [14]:

```
def define_phrases_candidates(sentence, max_words = 50,\
                              nb_words_in_seq=20, \
                              temperature=1, \
                              nb_candidates_sents=10, \
                              verbose = False):
    phrase_candidate = []
    generated_sentence = ""
    for i in range(nb_candidates_sents):
        generated_sentence, new_sentence = generate_phrase(sentence, \
                                                           max_words = max_words, \
                                                           nb_words_in_seq = nb_words_in_se
                                                           temperature=temperature, \
                                                           verbose = False)
        phrase_candidate.append([generated_sentence, new_sentence])

    if verbose == True :
        for phrase in phrase_candidate:
            print("    " , phrase[0])
    return phrase_candidate
```

In [15]:

```
def create_sentences(doc):
    ponctuation = [".","?","!",":","…"]
    sentences = []
    sent = []
    for word in doc:
        if word.text not in ponctuation:
            if word.text not in ("\n","\n\n",'\u2009','\xa0'):
                sent.append(word.text.lower())
        else:
            sent.append(word.text.lower())
            if len(sent) > 1:
                sentences.append(sent)
            sent=[]
    return sentences
```

In [16]:

```
def generate_training_vector(sentences_list, verbose = False):
    if verbose == True : print("generate vectors for each sentence...")
    seq = []
    V = []

    for s in sentences_list:
        v = d2v_model.infer_vector(create_sentences(nlp(s))[0], alpha=0.001, min_alpha=0.00
        V.append(v)
    V_val=np.array(V)
    V_val = np.expand_dims(V_val, axis=0)
    if verbose == True : print("Vectors generated!")
    return V_val
```

In [17]:

```
def select_next_phrase(model, V_val, candidate_list, verbose=False):
    sims_list = []
    preds = model.predict(V_val, verbose=0)[0]

    #calculate vector for each candidate
    for candidate in candidate_list:
        #calculate vector
        #print("calculate vector for : ", candidate[1])
        V = np.array(d2v_model.infer_vector(candidate[1]))
        sim = scipy.spatial.distance.cosine(V,preds)
        sims_list.append(sim)

    m = max(sims_list)
    index_max = sims_list.index(m)

    if verbose == True :
        print("selected phrase :")
        print("     ", candidate_list[index_max][0])
    return candidate_list[index_max]
```

In [18]:

```
def generate_paragraphe(phrase_seed, sentences_seed, \
                        max_words = 50, \
                        nb_words_in_seq=20, \
                        temperature=1, \
                        nb_phrases=30, \
                        nb_candidates_sents=10, \
                        verbose=True):

    sentences_list = sentences_seed
    sentence = phrase_seed
    text = []

    for p in range(nb_phrases):
        if verbose == True : print("")
        if verbose == True : print("##############")
        print("phrase ",p+1, "/", nb_phrases)
        if verbose == True : print("##############")
        if verbose == True:
            print('Sentence to generate phrase : ')
            print("     ", sentence)
            print("")
            print('List of sentences to constrain next phrase : ')
            print("     ", sentences_list)
            print("")

        #generate seed training vector
        V_val = generate_training_vector(sentences_list, verbose = verbose)

        #generate phrase candidate
        if verbose == True : print("generate phrases candidates...")
        phrases_candidates = define_phrases_candidates(sentence, \
                                                       max_words = max_words, \
                                                       nb_words_in_seq = nb_words_in_seq, \
                                                       temperature=temperature, \
                                                       nb_candidates_sents=nb_candidates_se
                                                       verbose = verbose)

        if verbose == True : print("select next phrase...")
        next_phrase = select_next_phrase(model_sequence, \
                                         V_val,
                                         phrases_candidates, \
                                         verbose=verbose)

        print("Next phrase: ",next_phrase[0])
        if verbose == True :
            print("")
            print("Shift phrases in sentences list...")
        for i in range(len(sentences_list)-1):
            sentences_list[i]=sentences_list[i+1]

        sentences_list[len(sentences_list)-1] = next_phrase[0]

        if verbose == True:
            print("done.")
            print("new list of sentences :")
            print("     ", sentences_list)
        sentence = next_phrase[1]

        text.append(next_phrase[0])
```

```
    return text
```

In [19]:

```
s1 = "The door swung open at once."
s2 = "A tall, black-haired witch in emerald-green robes stood there."
s3 = "She had a very stern face and Harry's first thought was that this was not someone to
s4 = "They followed Professor McGonagall across the flagged stone floor."
s5 = "The Sorting is a very important ceremony because, while you are here, your House will
s6 = "You will have classes with the rest of your House, sleep in your House dormitory, and
s7 = "At the end of the year, the House with the most points is awarded the House cup, a gr
s8 = "I hope each of you will be a credit to whichever House becomes yours."
s9 = "Harry nervously tried to flatten his hair."
s10 = "Harry swallowed."
s11 = "He looked around anxiously and saw that everyone else looked terrified, too."
s12 = "Harry tried hard not to listen to her."
s13 = "He kept his eyes fixed on the door."
s14 = "One by one, the ghosts floated away through the opposite wall."
s15 = "The hundreds of faces staring at them looked like pale lanterns in the flickering ca
```

In [20]:

```
sentences_list = [s1,s2,s3,s4,s5,s6,s7,s8,s9,s10,s11,s12,s13,s14,s15]
print(sentences_list)
```

```
['The door swung open at once.', 'A tall, black-haired witch in emerald-gree
n robes stood there.', "She had a very stern face and Harry's first thought
was that this was not someone to cross.", 'They followed Professor McGonagal
l across the flagged stone floor.', 'The Sorting is a very important ceremon
y because, while you are here, your House will be something like your family
within Hogwarts.', 'You will have classes with the rest of your House, sleep
in your House dormitory, and spend free time in your House common room.', 'A
t the end of the year, the House with the most points is awarded the House c
up, a great honor.', 'I hope each of you will be a credit to whichever House
becomes yours.', 'Harry nervously tried to flatten his hair.', 'Harry swallo
wed.', 'He looked around anxiously and saw that everyone else looked terrifi
ed, too.', 'Harry tried hard not to listen to her.', 'He kept his eyes fixed
on the door.', 'One by one, the ghosts floated away through the opposite wal
l.', 'The hundreds of faces staring at them looked like pale lanterns in the
flickering candlelight.']
```

In [21]:

```
phrase_seed, sentences_seed = create_seed(s1 + " " + s2 + " " +\
                                          s3 + " " + s4+ " " + s5 + " " +\
                                          s6 + " " + s7 + " " + s8 + " " +\
                                          s9+ " " + s10 + " " + s11 + " " +\
                                          s12 + " " + s13 + " " + s14+ " " + s15,20)
print(phrase_seed)
print(sentences_seed)
```

```
the opposite wall. The hundreds of faces staring at them looked like pale la
nterns in the flickering candlelight.
['the', 'opposite', 'wall', '.', 'The', 'hundreds', 'of', 'faces', 'starin
g', 'at', 'them', 'looked', 'like', 'pale', 'lanterns', 'in', 'the', 'flicke
ring', 'candlelight', '.']
```

In [29]:

```
text = generate_paragraphe([i.lower() for i in sentences_seed], sentences_list, \
                           max_words = 80, \
                           nb_words_in_seq = 30,\
                           temperature=0.25, \
                           nb_phrases=10, \
                           nb_candidates_sents=7, \
                           verbose=False)
```

```
phrase  1 / 10
Next phrase:   the own , they had been against the floor .
phrase  2 / 10
Next phrase:   " it 's your team , and , you 'll be a enough , potter , " i
'm yeh ?
phrase  3 / 10
Next phrase:   " " peeves , " said hagrid , " said hagrid , and ron , but th
ey knew it was n't think they 'd been an in a house .
phrase  4 / 10
Next phrase:   " i 've got to hear , " said ron , but harry looked at the gr
ound and o ' done it , and you 're going to get past his and your there .
phrase  5 / 10
Next phrase:   " snape 's , " said ron , but he had been the out of the hat
, and the solid of the hat , and harry was trying to the invisibility cloak
, and that was as they were never on the end of the library , and harry was
only to stop him .
phrase  6 / 10
Next phrase:   madam hooch was trying to be went to a distance .
phrase  7 / 10
Next phrase:   it was a castle , which was n't like a fifty .
phrase  8 / 10
Next phrase:   " " what ?
phrase  9 / 10
Next phrase:   " " said ron , but ron looked at once , and she had been plac
e in the dursleys ' , but they were trying to be soon .
phrase  10 / 10
Next phrase:   it was n't be down .
```

In [23]:

```
print("generated text: ")
for t in text:
    print(t)
```

```
generated text:
 the black of the way , and the standing , they were n't going to be with th
e other .
 " i 'm you ?
 " " i 'm you , " said ron , but he was sure it was n't going to be making ,
but they were n't be see that was nothing , but he had been not to be later
, but they were n't have been points to get out of the mirror .
 they were n't be coming from the bludgers , " said ron .
 " what 's you ?
```

In [ ]: