

## Тема “Генерация текстов (языковое моделирование)”

Разобраться с моделькой генерации текста, собрать самим или взять датасет с вебинара и обучить генератор текстов

In [3]:

```
import tensorflow as tf

import numpy as np
import os
import time
import re
import sys

from nltk.tokenize import RegexpTokenizer
from nltk.corpus import stopwords
from keras.models import Sequential
from keras.layers import Dense, Dropout, LSTM
from keras.utils import np_utils
from keras.callbacks import ModelCheckpoint

import numpy as np
import keras
from keras.models import Sequential, Model
from keras.layers import Dense, Dropout, Activation, Input, Embedding, Conv1D, GlobalMaxPool
from keras.preprocessing.text import Tokenizer
# from keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.preprocessing.sequence import pad_sequences
from keras.callbacks import TensorBoard
# from keras.objectives import categorical_crossentropy
from keras.callbacks import EarlyStopping
```

In [4]:

```
def tokenize_words(input):
    # Lowercase everything to standardize it
    input = input.lower()
    input = re.sub(r'[^а-яА-Я]', ' ', input)
    # Instantiate the tokenizer
    tokenizer = RegexpTokenizer(r'\w+')
    # tokenizer = RegexpTokenizer(r'[а-яА-Я]')
    tokens = tokenizer.tokenize(input)

    # If the created token isn't in the stop words, make it part of "filtered"
    filtered = filter(lambda token: token not in stopwords.words('russian'), tokens)
    return " ".join(filtered)
```

Загружаю данные

In [5]:

```
path_to_file = 'evgenyi_onegin.txt'
```

In [6]:

```
list_word = []
with open(path_to_file, 'r', encoding = 'utf-8') as file:
    file = file.readlines()
    for i in file:
        for j in i.split():
            list_word.append(j)
```

In [7]:

```
list_word = [''.join(i) for i in list_word]
```

In [8]:

```
text = ' '.join(list_word)
```

In [9]:

```
text[:20]
```

Out[9]:

'Александр Сергеевич '

обработка

In [10]:

```
text = tokenize_words(text)
```

In [11]:

```
text[:100]
```

Out[11]:

'александр сергеевич пушкин евгений онегин роман стихах мысля гордый свет за  
бавить вниманье дружбы во'

## Token

Представление текста в виде чисел, где если по токенам число соответствует токену, в би граммах число соответствует двум символам, в посимвольной каждому символу

In [12]:

```
vocab = sorted(set(text.split()))
print('{} unique token'.format(len(vocab)))
```

8188 unique token

In [13]:

```
char2idx = {u:i for i, u in enumerate(vocab)}
# idx2char = {i:u for i, u in enumerate(vocab)}
# idx2char_bi = {i:u for i, u in enumerate(vocab_bi)}
# idx2char_char = {i:u for i, u in enumerate(vocab_char)}
idx2char = np.array(vocab)
```

In [14]:

```
text_as_int = np.array([char2idx[c] for c in text.split()])
```

In [15]:

```
text_as_int
```

Out[15]:

```
array([ 21, 6254, 5656, ..., 4182, 3315, 2669])
```

## Bi

In [16]:

```
# text_bi = tokenize_words(text)
```

In [17]:

```
text_bi = [''.join(text[ch_i:ch_i+2]) for ch_i in range(0,len(text)-2,2)]
```

In [18]:

```
text_bi[:10]
```

Out[18]:

```
['ал', 'ек', 'са', 'нд', 'р ', 'се', 'рг', 'е', 'ви', 'ч ']
```

In [19]:

```
vocab_bi = sorted(set(text_bi))
print('{} unique token'.format(len(vocab_bi)))
```

```
653 unique token
```

In [20]:

```
char2idx_bi = {u:i for i, u in enumerate(vocab_bi)}
# idx2char_bi = {i:u for i, u in enumerate(vocab_bi)}
idx2char_bi = np.array(vocab_bi)
```

In [21]:

```
text_as_int_bi = np.array([char2idx_bi[c] for c in text_bi])
```

In [22]:

```
text_as_int_bi
```

Out[22]:

```
array([ 39, 159, 419, ..., 231, 9, 356])
```

## char

In [23]:

```
vocab_char = sorted(set(text))
print('{} unique token'.format(len(vocab_char)))
```

```
33 unique token
```

In [24]:

```
char2idx_char = {u:i for i, u in enumerate(vocab_char)}
# idx2char_bi = {i:u for i, u in enumerate(vocab_bi)}
# idx2char_char = {i:u for i, u in enumerate(vocab_char)}
idx2char_char = np.array(vocab_char)
```

In [25]:

```
text_as_int_char = np.array([char2idx_char[c] for c in text])
```

In [26]:

```
text_as_int_char
```

Out[26]:

```
array([ 1, 12, 6, ..., 14, 6, 23])
```

## Обучение

**теперь можно сделать набор данных для обучения модели**

## token

In [27]:

```
seq_length = 100
examples_per_epoch = len(text.split())//(seq_length+1)
```

In [28]:

```
token_dataset = tf.data.Dataset.from_tensor_slices(text_as_int)

for i in token_dataset.take(10):
    print(idx2char[i.numpy()])
```

александр  
сергеевич  
пушкин  
евгений  
онегин  
роман  
стихах  
мысля  
гордый  
свет

In [ ]:

In [29]:

```
sequences = token_dataset.batch(seq_length+1, drop_remainder=True)

for item in sequences.take(5):
    print(repr(' '.join(idx2char[item.numpy()])))
```

'александр сергеевич пушкин евгений онегин роман стихах мысля гордый свет за  
бавить вниманье дружбы возлюбя хотел тебе представить залог достойнее достой  
нее души прекрасной святой исполненной мечты поэзии живой ясной высоких дум  
простоты рукой пристрастной прими собранье пестрых глав полусмешных полуучеа  
льных простонародных идеальных небрежный плод моих забав бессонниц легких вд  
охновений незрелых увядших лет ума холодных наблюдений сердца горестных заме  
т глава первая жить торопится чувствовать спешит кн вяземский дядя самых чес  
тных правил шутку занемог уважать заставил выдуматель мог пример другим наука  
боже скуча больным сидеть день ночь отходя шагу прочь какое низкое коварство  
полуживого забавлять подушки поправлять печально подносить лекарство вздыхат  
ь думать черт возьмет'

'думал молодой повеса летя пыли почтовых всевышней волею зевеса наследник св  
оих родных друзья людмилы руслана героем моего романа предисловий сей час по  
звольте познакомить онегин добрый приятель родился брегах невы родились блис  
тали читатель некогда гулял вреден север служив отлично благородно долгами ж  
ил отец давал бала ежегодно промотался судьба евгения хранила сперва ходила  
сменил ребенок резов мил француз убогой измучилось дитя учил всему шутя доку  
чал моралью строгой слегка шалости бранил летний сад гулять водил юности мя  
тежной пришла евгению пора пора надежд грусти нежной прогнали двора онегин св  
ободе острижен последней моде лондонский одет увидел свет французски соверше  
нно мог изъясняться писал легко мазурку танцевал кланялся'

'непринужденно свет решил умен очень мил учились понемногу чему воспитаньем  
слава богу немудрено блеснуть онегин мненью многих судей решительных строгих  
ученый малый педант имел счастливый талант принужденья разговоре коснуться с  
легка ученым видом знатока хранить молчанье важном споре возбуждать улыбку д  
ам огнем неожиданных эпиграмм латынь моды вышла ныне правду сказать знал довол  
ьно латыне эпиграфы разбирать потолковать ювенале конце письма поставить пом  
нил греха энеиды стиха рыться имел охоты хронологической пыли бытописания зе  
мли дней минувших анекдоты ромула наших дней хранил памяти своей высокой стр  
асти имея звуков жизни щадить мог ямба хорея бились отличить бранил гомера ф  
еокрита зато читал адама смита глубокой эконом умел судить'

'государство богатеет живет почему нужно золота простой продукт имеет отец п  
онять мог земли отдавал залог знал евгений пересказать недосуг истинный гени  
й знал тверже наук измлада труд мука отрада занимало целый день тоскующую ле  
нь наука страсти нежной которую воспел назон страдальцем кончил свой век бле  
стящий мятежный молдавии глуши степей вдали италии своей рано мог лицемерить  
таить надежду ревновать разуверять заставить верить казаться мрачным изныват  
ь являться гордым послушным внимательным иль равнодушным томно молчалив плам  
енно красноречив сердечных письмах небрежен одним дыша одно любя умел забыть  
взор быстр нежен стыдлив дерзок порой блистал послушною слезой умел казаться  
новым шутя невинность изумлять пугать отчаяньем готовым приятной'

'лестью забавлять ловить минуту умиленья невинных лет предубежденья умом стр  
астью побеждать невольной ласки ожидать молить требовать признанья подслушат  
ь сердца первый звук преследовать любовь добиться тайного свиданья наедине д  
авать уроки тишине рано мог тревожить сердца кокеток записных хотелось уничт  
ожить соперников своих язвительно злословил какие сети готовил блаженные муж  
ья оставались друзья ласкал супруг лукавый фобласа давний ученик недоверчивы  
й стариk рогоносец величавый довольный собой своим обедом женой бывало посте  
ле нему записочки несут приглашенья самом деле дома вечер зовут бал детский  
праздник поскакет проказник кого начнет равно везде поспеть немудрено покаме  
ст утреннем уборе надев широкий боливар онегин едет бульвар гуляет просторе  
пока недремлющий берегет прозвонит'

In [30]:

```
def split_input_target(chunk):
    input_text = chunk[:-1]
    target_text = chunk[1:]
    return input_text, target_text

dataset = sequences.map(split_input_target)
```

In [31]:

```
for input_example, target_example in dataset.take(1):
    print('Input data: ', repr(' '.join(idx2char[input_example.numpy()]))))
    print('Target data: ', repr(' '.join(idx2char[target_example.numpy()]))))
```

Input data: 'александр сергеевич пушкин евгений онегин роман стихах мысля г  
ордый свет забавить вниманье дружбы возлюбя хотел тебе представить залог дос  
тойнее достойнее души прекрасной святой исполненной мечты поэзии живой ясной  
высоких дум простоты рукой пристрастной прими собранье пестрых глав полусмеш  
ных полупечальных простонародных идеальных небрежный плод моих забав бессонн  
иц легких вдохновений незрелых увядших лет ума холодных наблюдений сердца го  
рестных замет глава первая жить торопится чувствовать спешит кн вяземский дя  
дя самых честных правил шутку занемог уважать заставил выдумать мог пример д  
ругим наука боже скука больным сидеть день ночь отходя шагу прочь какое низк  
ое коварство полуживого забавлять подушки поправлять печально подносить лека  
рство вздыхать думать черт'

Target data: 'сергеевич пушкин евгений онегин роман стихах мысля гордый свет  
забавить вниманье дружбы возлюбя хотел тебе представить залог достойнее дост  
ойнее души прекрасной святой исполненной мечты поэзии живой ясной высоких ду  
м простоты рукой пристрастной прими собранье пестрых глав полусмешных полупе  
чальных простонародных идеальных небрежный плод моих забав бессонниц легких  
вдохновений незрелых увядших лет ума холодных наблюдений сердца горестных за  
мет глава первая жить торопится чувствовать спешит кн вяземский дядя самых ч  
естных правил шутку занемог уважать заставил выдумать мог пример другим наук  
а боже скука больным сидеть день ночь отходя шагу прочь какое низкое коварст  
во полуживого забавлять подушки поправлять печально подносить лекарство взды  
хать думать черт возьмет'

**bi**

In [32]:

```
examples_per_epoch_bi = len(text_bi)//(seq_length+1)
```

In [33]:

```
token_dataset_bi = tf.data.Dataset.from_tensor_slices(text_as_int_bi)

for i in token_dataset_bi.take(10):
    print(idx2char_bi[i.numpy()])
```

ал  
ек  
са  
нд  
р  
се  
рг  
ее  
ви  
ч

In [34]:

```
sequences_bi = token_dataset_bi.batch(seq_length+1, drop_remainder=True)

for item in sequences_bi.take(5):
    print(repr(''.join(idx2char_bi[item.numpy()])))
```

'александр сергеевич пушкин евгений онегин роман стихах мысля гордый свет за  
бавить вниманье дружбы возлюбя хотел тебе представить залог достойнее достой  
нее души прекрасной святой исполненной мечты поэзии'  
' живой ясной высоких дум простоты рукой пристрастной прими собранье пестрых  
глав полуусмешных полупечальных простонародных идеальных небрежный плод моих  
забав бессонниц легких вдохновений незрелых увядши'  
'х лет ума холодных наблюдений сердца горестных замет глава первая жить торо  
пится чувствовать спешит кн вяземский дядя самых честных правил шутку занемо  
г уважать заставил выдумать мог пример другим наука'  
' боже скука больным сидеть день ночь отходя шагу прочь какое низкое коварст  
во полуживого забавлять подушки поправлять печально подносить лекарство взды  
хать думать черт возьмет думал молодой повеса летя '  
'пыли почтовых всевышней волею зевеса наследник своих родных друзья людмилы  
руслана героем моего романа предисловий сей час позвольте познакомить онегин  
добрый приятель родился брегах невы родились блист'

In [35]:

```
def split_input_target(chunk):
    input_text = chunk[:-1]
    target_text = chunk[1:]
    return input_text, target_text

dataset_bi = sequences_bi.map(split_input_target)
```

In [36]:

```
for input_example, target_example in dataset_bi.take(1):
    print('Input data: ', repr(''.join(idx2char_bi[input_example.numpy()]))))
    print('Target data: ', repr(''.join(idx2char_bi[target_example.numpy()]))))
```

Input data: 'александр сергеевич пушкин евгений онегин роман стихах мысля г  
ордый свет забавить вниманье дружбы возлюбя хотел тебе представить залог дос  
тойнее достойнее души прекрасной святой исполненной мечты поэз'

Target data: 'ександр сергеевич пушкин евгений онегин роман стихах мысля гор  
дый свет забавить вниманье дружбы возлюбя хотел тебе представить залог досто  
йнее достойнее души прекрасной святой исполненной мечты поэзии'

## char

In [37]:

```
examples_per_epoch_bi = len(text)//(seq_length+1)
```

In [38]:

```
token_dataset_char = tf.data.Dataset.from_tensor_slices(text_as_int_char)

for i in token_dataset_char.take(10):
    print(idx2char_char[i.numpy()])
```

а  
л  
е  
к  
с  
а  
н  
д  
р

In [39]:

```
sequences_char = token_dataset_char.batch(seq_length+1, drop_remainder=True)

for item in sequences_char.take(5):
    print(repr(''.join(idx2char_char[item.numpy()]))))
```

'александр сергеевич пушкин евгений онегин роман стихах мысля гордый свет за  
бавить вниманье дружбы воз'  
'любя хотел тебе представить залог достойнее достойнее души прекрасной свято  
й исполненной мечты поэзии'  
'живой ясной высоких дум простоты рукой пристрастной прими собранье пестрых  
глав полусмешных полуучка'  
'льных простонародных идеальных небрежный плод моих забав бессонниц легких в  
дохновений незрелых увядши'  
'х лет ума холодных наблюдений сердца горестных замет глава первая жить торо  
пится чувствовать спешит к'

In [40]:

```
def split_input_target(chunk):
    input_text = chunk[:-1]
    target_text = chunk[1:]
    return input_text, target_text

dataset_char = sequences_char.map(split_input_target)
```

In [41]:

```
for input_example, target_example in dataset_char.take(1):
    print('Input data: ', repr(''.join(idx2char_char[input_example.numpy()]))))
    print('Target data: ', repr(''.join(idx2char_char[target_example.numpy()]))))
```

Input data: 'александр сергеевич пушкин евгений онегин роман стихах мысля г  
ордый свет забавить вниманье дружбы во'

Target data: 'лександр сергеевич пушкин евгений онегин роман стихах мысля го  
рдый свет забавить вниманье дружбы воз'

In [ ]:

In [ ]:

In [42]:

```
# Batch size
BATCH_SIZE = 64

BUFFER_SIZE = 10000
#token
dataset = dataset.shuffle(BUFFER_SIZE).batch(BATCH_SIZE, drop_remainder=True)
dataset_bi = dataset.bi.shuffle(BUFFER_SIZE).batch(BATCH_SIZE, drop_remainder=True)
dataset_char = dataset_char.shuffle(BUFFER_SIZE).batch(BATCH_SIZE, drop_remainder=True)

print(dataset)
print(dataset.bi)
print(dataset_char)
```

```
<BatchDataset element_spec=(TensorSpec(shape=(64, 100), dtype=tf.int32, name=None), TensorSpec(shape=(64, 100), dtype=tf.int32, name=None))>
<BatchDataset element_spec=(TensorSpec(shape=(64, 100), dtype=tf.int32, name=None), TensorSpec(shape=(64, 100), dtype=tf.int32, name=None))>
<BatchDataset element_spec=(TensorSpec(shape=(64, 100), dtype=tf.int32, name=None), TensorSpec(shape=(64, 100), dtype=tf.int32, name=None))>
```

In [43]:

```
# token
vocab_size = len(vocab)
vocab_size_bi = len(vocab_bi)
vocab_size_char = len(vocab_char)

# The embedding dimension
embedding_dim = 256

# Number of RNN units
rnn_units = 1024
```

Создание структуры модели

In [44]:

```
def build_model(vocab_size, embedding_dim, rnn_units, batch_size):
    model = tf.keras.Sequential([
        tf.keras.layers.Embedding(vocab_size, embedding_dim,
                                  batch_input_shape=[batch_size, None]),

        tf.keras.layers.LSTM(rnn_units,
                            return_sequences=True,
                            stateful=True,
                            recurrent_initializer='glorot_uniform'),

        tf.keras.layers.LSTM(rnn_units,
                            return_sequences=True,
                            stateful=True,
                            recurrent_initializer='glorot_uniform'),

        tf.keras.layers.LSTM(rnn_units,
                            return_sequences=True,
                            stateful=True,
                            recurrent_initializer='glorot_uniform'),

        tf.keras.layers.LSTM(rnn_units,
                            return_sequences=True,
                            stateful=True,
                            recurrent_initializer='glorot_uniform'),

        tf.keras.layers.Dense(vocab_size)
    ])
    return model
```

In [47]:

```
model = build_model(  
    vocab_size=vocab_size,  
    embedding_dim=embedding_dim,  
    rnn_units=rnn_units,  
    batch_size=BATCH_SIZE)  
model_bi = build_model(  
    vocab_size=len(vocab_bi),  
    embedding_dim=embedding_dim,  
    rnn_units=rnn_units,  
    batch_size=BATCH_SIZE)  
model_char = build_model(  
    vocab_size=len(vocab_char),  
    embedding_dim=embedding_dim,  
    rnn_units=rnn_units,  
    batch_size=BATCH_SIZE)
```

In [48]:

```
print(model.summary())
print(model.bi.summary())
print(model.char.summary())
```

Model: "sequential"

Layer (type)	Output Shape	Param #
<hr/>		
embedding (Embedding)	(64, None, 256)	2096128
lstm (LSTM)	(64, None, 1024)	5246976
lstm_1 (LSTM)	(64, None, 1024)	8392704
lstm_2 (LSTM)	(64, None, 1024)	8392704
lstm_3 (LSTM)	(64, None, 1024)	8392704
dense (Dense)	(64, None, 8188)	8392700
<hr/>		
Total params: 40,913,916		
Trainable params: 40,913,916		
Non-trainable params: 0		

None

Model: "sequential\_1"

Layer (type)	Output Shape	Param #
<hr/>		
embedding_1 (Embedding)	(64, None, 256)	167168
lstm_4 (LSTM)	(64, None, 1024)	5246976
lstm_5 (LSTM)	(64, None, 1024)	8392704
lstm_6 (LSTM)	(64, None, 1024)	8392704
lstm_7 (LSTM)	(64, None, 1024)	8392704
dense_1 (Dense)	(64, None, 653)	669325
<hr/>		
Total params: 31,261,581		
Trainable params: 31,261,581		
Non-trainable params: 0		

None

Model: "sequential\_2"

Layer (type)	Output Shape	Param #
<hr/>		
embedding_2 (Embedding)	(64, None, 256)	8448
lstm_8 (LSTM)	(64, None, 1024)	5246976
lstm_9 (LSTM)	(64, None, 1024)	8392704

lstm_10 (LSTM)	(64, None, 1024)	8392704
lstm_11 (LSTM)	(64, None, 1024)	8392704
dense_2 (Dense)	(64, None, 33)	33825

=====

Total params: 30,467,361  
Trainable params: 30,467,361  
Non-trainable params: 0

None

In [49]:

```
#token
for input_example_batch, target_example_batch in dataset.take(1):
    example_batch_predictions = model(input_example_batch)
    print(example_batch_predictions.shape, "# (batch_size, sequence_length, vocab_size)")
#bi
for input_example_batch_bi, target_example_batch_bi in dataset_bi.take(1):
    example_batch_predictions_bi = model_bi(input_example_batch_bi)
    print(example_batch_predictions_bi.shape, "# (batch_size, sequence_length, vocab_size)")
#char
for input_example_batch_char, target_example_batch_char in dataset_char.take(1):
    example_batch_predictions_char = model_char(input_example_batch_char)
    print(example_batch_predictions_char.shape, "# (batch_size, sequence_length, vocab_size)")

(64, 100, 8188) # (batch_size, sequence_length, vocab_size)
(64, 100, 653) # (batch_size, sequence_length, vocab_size)
(64, 100, 33) # (batch_size, sequence_length, vocab_size)
```

In [50]:

```
sampled_indices = tf.random.categorical(example_batch_predictions[0], num_samples=1)
sampled_indices = tf.squeeze(sampled_indices, axis=-1).numpy()
```

In [51]:

```
sampled_indices_bi = tf.random.categorical(example_batch_predictions_bi[0], num_samples=1)
sampled_indices_bi = tf.squeeze(sampled_indices_bi, axis=-1).numpy()
```

In [52]:

```
sampled_indices_char = tf.random.categorical(example_batch_predictions_char[0], num_samples=1)
sampled_indices_char = tf.squeeze(sampled_indices_char, axis=-1).numpy()
```

#####



## Предсказание без обучения

In [53]:

```
#token
print("Input: \n", repr(" ".join(idx2char[input_example_batch[0]])))
print()
print("Next Char Predictions: \n", repr(" ".join(idx2char[sampled_indices ])))
```

Input:

'гандисон который нам наводит сон мечтательницы нежной единый образ облекл  
ись одном онегине слились воображаясь героиной своих возлюбленных творцов кл  
арисой юлией дельфиной татьяна тишине лесов одна опасной книгой бродит ищет  
находит свой тайный жар свои мечты плоды сердечной полноты вздыхает присвоя  
чужой восторг чужую грусть забвенье шепчет наизусть письмо милого героя наш  
герой б верно гандисон свой слог важный лад настроя бывало пламенный творец  
являл нам своего героя совершенства образец одарял предмет любимый неправедн  
о гонимый душой чувствительной умом привлекательным лицом питая жар чистейше  
й страсти восторженный герой готов жертвовать собой конце последней части на  
казан порок добру достойный венок нынче умы тумане мораль'

Next Char Predictions:

'тепла воспоминать шаг повернула ножка вставал кровью узоры забава зато вин  
а мельпомены обесславить долинах бразды темное острые учил обман видел тяжко  
окну журналистам освободясь резкий сулили греха своею угрюмый издатель красе  
творческие ожесточиться щадить мостовой приятелем увидеть ветров ковров смир  
ив обагрилась разыграйтесь туман мнится тошно четвертый вспыхнул понимает бе  
режно добру взвившись полтиной пылкой плаще ножницы хоры грустно разумею шум  
ящею сброду сгорая молитвой разуверять звездами небес встречаешь улыбку вели  
т разыграйтесь горой грязь отъехать лепетом нужно пускаюсь остро уныние свят  
о большой охладел вышнем домом успокоил конец целью высока влюблался галоп н  
ужды врага бредни раскаянье острые именно бродит снам видит верно двойке рош  
и'

In [54]:

```
#bi
print("Input: \n", repr("".join(idx2char_bi[input_example_batch_bi[0]])))
print()
print("Next Char Predictions: \n", repr("".join(idx2char_bi[sampled_indices_bi ])))
```

Input:

'воя вся жизнь залогом свиданья верного тобой знаю послан богом гроба храни  
тель сновиденьях являлся незримый мил твой чудный взгляд томил душе твой гол  
ос раздавался давно это сон вошел вмиг узнала вся '

Next Char Predictions:

'эзяяяниийшойяявыхсяннгчтшхигуербошжъжнсав ънанчшбнмиящвхпоалнуйнсерцщиср  
уимбм дъзсугияужлу апу ецомжмдъдуишэгцецих аьпрыозпщеидещевнзъсядцтавчклсс  
едувэгжнггыеесхгябетмъянуих прдньжм хтаробры я съся'

In [55]:

```
#char
print("Input: \n", repr("".join(idx2char_char[input_example_batch_char[0]])))
print()
print("Next Char Predictions: \n", repr("".join(idx2char_char[sampled_indices_char ])))
```

Input:

'м подружки измарали конца начала кругом сюда назло правописанью стихи меры  
преданью знак дружбы верн'

Next Char Predictions:

'озэквъящшаелжфззшмндэжеююфчмюйсмззгмомттьцийгвкемевшэтзчщепудспчояаэыжш  
шфгблымвгншжчфцмзуюйтмсбцо'

## train model

In [56]:

```
def loss(labels, logits):
    return tf.keras.losses.sparse_categorical_crossentropy(labels, logits, from_logits=True)
# model.compile(loss='categorical_crossentropy', optimizer=sgd, metrics=['accuracy'])
example_batch_loss = loss(target_example_batch, example_batch_predictions)
print("Prediction shape: ", example_batch_predictions.shape, "# (batch_size, sequence_length)")
print("scalar_loss:      ", example_batch_loss.numpy().mean())
```

Prediction shape: (64, 100, 8188) # (batch\_size, sequence\_length, vocab\_size)  
scalar\_loss: 9.010424

In [57]:

```
#token
def loss(labels, logits):
    return tf.keras.losses.sparse_categorical_crossentropy(labels, logits, from_logits=True)

example_batch_loss = loss(target_example_batch, example_batch_predictions)
print("Prediction shape: ", example_batch_predictions.shape, "# (batch_size, sequence_length)")
print("scalar_loss:      ", example_batch_loss.numpy().mean())
```

Prediction shape: (64, 100, 8188) # (batch\_size, sequence\_length, vocab\_size)  
scalar\_loss: 9.010424

In [58]:

```
#bi
def loss(labels, logits):
    return tf.keras.losses.sparse_categorical_crossentropy(labels, logits, from_logits=True)

example_batch_loss_bi = loss(target_example_batch_bi, example_batch_predictions_bi)
print("Prediction shape: ", example_batch_predictions_bi.shape, "# (batch_size, sequence_length, vocab_size)")
print("scalar_loss:      ", example_batch_loss_bi.numpy().mean())
```

Prediction shape: (64, 100, 653) # (batch\_size, sequence\_length, vocab\_size)  
 scalar\_loss: 6.481587

In [59]:

```
#char
def loss(labels, logits):
    return tf.keras.losses.sparse_categorical_crossentropy(labels, logits, from_logits=True)

example_batch_loss_char = loss(target_example_batch_char, example_batch_predictions_char)
print("Prediction shape: ", example_batch_predictions_char.shape, "# (batch_size, sequence_length, vocab_size)")
print("scalar_loss:      ", example_batch_loss_char.numpy().mean())
```

Prediction shape: (64, 100, 33) # (batch\_size, sequence\_length, vocab\_size)  
 scalar\_loss: 3.4968624

In [60]:

```
model.compile(optimizer='adam', loss=loss)
model_bi.compile(optimizer='adam', loss=loss)
model_char.compile(optimizer='adam', loss=loss)
```

## checkpoints

In [74]:

```
filepath = "model_weights_saved.hdf5"
checkpoint = ModelCheckpoint(filepath, monitor='loss', verbose=1, save_best_only=True, mode='auto')
desired_callbacks = [checkpoint]
filepath_bi = "model_weights_saved_bi.hdf5"
checkpoint_bi = ModelCheckpoint(filepath_bi, monitor='loss', verbose=1, save_best_only=True)
desired_callbacks_bi = [checkpoint_bi]
filepath_char = "model_weights_saved_char.hdf5"
checkpoint_char = ModelCheckpoint(filepath_char, monitor='loss', verbose=1, save_best_only=True)
desired_callbacks_char = [checkpoint_char]
```

In [65]:

```
EPOCHS = 30
```

In [66]:

```
%time
#char
history_char = model_char.fit(dataset_char, epochs=EPOCHS, callbacks=desired_callbacks_char)

Epoch 1/30
16/16 [=====] - ETA: 0s - loss: 3.4016
Epoch 1: loss improved from inf to 3.40159, saving model to model_weights_saved_char.hdf5
16/16 [=====] - 4422s 292s/step - loss: 3.4016
Epoch 2/30
16/16 [=====] - ETA: 0s - loss: 3.1014
Epoch 2: loss improved from 3.40159 to 3.10141, saving model to model_weights_saved_char.hdf5
16/16 [=====] - 376s 23s/step - loss: 3.1014
Epoch 3/30
16/16 [=====] - ETA: 0s - loss: 3.0777
Epoch 3: loss improved from 3.10141 to 3.07768, saving model to model_weights_saved_char.hdf5
16/16 [=====] - 382s 24s/step - loss: 3.0777
Epoch 4/30
16/16 [=====] - ETA: 0s - loss: 2.9713
Epoch 4: loss improved from 3.07768 to 2.97129, saving model to model_weights_saved_char.hdf5
16/16 [=====] - 382s 24s/step - loss: 2.97129
```

In [67]:

```
%time
#bi
history_bi = model_bi.fit(dataset_bi, epochs=EPOCHS, callbacks=desired_callbacks_bi)

Epoch 26: loss did not improve from 5.59020
8/8 [=====] - 199s 25s/step - loss: 5.5924
Epoch 27/30
8/8 [=====] - ETA: 0s - loss: 5.5916
Epoch 27: loss did not improve from 5.59020
8/8 [=====] - 200s 25s/step - loss: 5.5916
Epoch 28/30
8/8 [=====] - ETA: 0s - loss: 5.5918
Epoch 28: loss did not improve from 5.59020
8/8 [=====] - 200s 25s/step - loss: 5.5918
Epoch 29/30
8/8 [=====] - ETA: 0s - loss: 5.5921
Epoch 29: loss did not improve from 5.59020
8/8 [=====] - 200s 25s/step - loss: 5.5921
Epoch 30/30
8/8 [=====] - ETA: 0s - loss: 5.5913
Epoch 30: loss did not improve from 5.59020
8/8 [=====] - 200s 25s/step - loss: 5.5913
Wall time: 1h 40min 17s
```

In [68]:

```
%time  
#token  
history = model.fit(dataset, epochs=EPOCHS, callbacks=desired_callbacks)
```

```
Epoch 1/30  
2/2 [=====] - ETA: 0s - loss: 9.0103  
Epoch 1: loss improved from inf to 9.01034, saving model to model_weights_saved.hdf5  
2/2 [=====] - 79s 30s/step - loss: 9.0103  
Epoch 2/30  
2/2 [=====] - ETA: 0s - loss: 8.9930  
Epoch 2: loss improved from 9.01034 to 8.99303, saving model to model_weights_saved.hdf5  
2/2 [=====] - 61s 30s/step - loss: 8.9930  
Epoch 3/30  
2/2 [=====] - ETA: 0s - loss: 8.8477  
Epoch 3: loss improved from 8.99303 to 8.84767, saving model to model_weights_saved.hdf5  
2/2 [=====] - 61s 30s/step - loss: 8.8477  
Epoch 4/30  
2/2 [=====] - ETA: 0s - loss: 8.7157  
Epoch 4: loss improved from 8.84767 to 8.71573, saving model to model_weights_saved.hdf5  
^C
```

In [ ]:

```
model_char.save('generate_char')
```

In [ ]:

```
model_bi.save('generate_bi')
```

In [ ]:

```
model.save('generate_token')
```

In [27]:

```
from tensorflow import keras
```

In [28]:

```
model_char = keras.models.load_model("generate_char", compile = False)
```

In [29]:

```
model_bi = keras.models.load_model("generate_bi", compile = False)
```

In [30]:

```
model = keras.models.load_model("generate_token", compile = False)
model.summary()
```

Model: "sequential"

Layer (type)	Output Shape	Param #
<hr/>		
embedding (Embedding)	(64, None, 256)	2096128
lstm (LSTM)	(64, None, 1024)	5246976
lstm_1 (LSTM)	(64, None, 1024)	8392704
lstm_2 (LSTM)	(64, None, 1024)	8392704
lstm_3 (LSTM)	(64, None, 1024)	8392704
dense (Dense)	(64, None, 8188)	8392700
<hr/>		
Total params: 40,913,916		
Trainable params: 40,913,916		
Non-trainable params: 0		

In [31]:

```
filename = "model_weights_saved.hdf5"
model = build_model(vocab_size, embedding_dim, rnn_units, batch_size=1)
model.load_weights(filename)
model.build(tf.TensorShape([1, None]))
model.summary()
```

Model: "sequential\_1"

Layer (type)	Output Shape	Param #
<hr/>		
embedding_1 (Embedding)	(1, None, 256)	2096128
lstm_4 (LSTM)	(1, None, 1024)	5246976
lstm_5 (LSTM)	(1, None, 1024)	8392704
lstm_6 (LSTM)	(1, None, 1024)	8392704
lstm_7 (LSTM)	(1, None, 1024)	8392704
dense_1 (Dense)	(1, None, 8188)	8392700
<hr/>		
Total params: 40,913,916		
Trainable params: 40,913,916		
Non-trainable params: 0		

In [33]:

```
filename = "model_weights_saved.bi.hdf5"
model_bi = build_model(vocab_size_bi, embedding_dim, rnn_units, batch_size=1)
model_bi.load_weights(filename)
model_bi.build(tf.TensorShape([1, None]))
model_bi.summary()
```

Model: "sequential\_2"

Layer (type)	Output Shape	Param #
<hr/>		
embedding_2 (Embedding)	(1, None, 256)	167168
lstm_8 (LSTM)	(1, None, 1024)	5246976
lstm_9 (LSTM)	(1, None, 1024)	8392704
lstm_10 (LSTM)	(1, None, 1024)	8392704
lstm_11 (LSTM)	(1, None, 1024)	8392704
dense_2 (Dense)	(1, None, 653)	669325
<hr/>		
Total params: 31,261,581		
Trainable params: 31,261,581		
Non-trainable params: 0		

In [35]:

```
filename = "model_weights_saved_char.hdf5"
model_char = build_model(vocab_size_char, embedding_dim, rnn_units, batch_size=1)
model_char.load_weights(filename)
model_char.build(tf.TensorShape([1, None]))
model_char.summary()
```

Model: "sequential\_3"

Layer (type)	Output Shape	Param #
<hr/>		
embedding_3 (Embedding)	(1, None, 256)	8448
lstm_12 (LSTM)	(1, None, 1024)	5246976
lstm_13 (LSTM)	(1, None, 1024)	8392704
lstm_14 (LSTM)	(1, None, 1024)	8392704
lstm_15 (LSTM)	(1, None, 1024)	8392704
dense_3 (Dense)	(1, None, 33)	33825
<hr/>		
Total params: 30,467,361		
Trainable params: 30,467,361		
Non-trainable params: 0		

In [56]:

```
def generate_text(model, start_string, char2idx, idx2char):

    start_string = tokenize_words(start_string)
    # Evaluation step (generating text using the learned model)
    print(start_string)
    # Number of characters to generate
    num_generate = 200

    # Converting our start string to numbers (vectorizing)
    input_eval = [char2idx[s] for s in start_string.split()]
    input_eval = tf.expand_dims(input_eval, 0)

    # Empty string to store our results
    text_generated = []

    # Low temperature results in more predictable text.
    # Higher temperature results in more surprising text.
    # Experiment to find the best setting.
    temperature = 0.1

    # Here batch size == 1
    model.reset_states()
    for i in range(num_generate):
        predictions = model(input_eval)
        predictions = tf.squeeze(predictions, 0)
        # using a categorical distribution to predict the character returned by the model
        predictions = predictions / temperature
        predicted_id = tf.random.categorical(predictions, num_samples=1)[-1, 0].numpy()

        # Pass the predicted character as the next input to the model
        # along with the previous hidden state
        input_eval = tf.expand_dims([predicted_id], 0)

        text_generated.append(idx2char[predicted_id])

    return (start_string + ' ' + ''.join(text_generated))
```

In [57]:

```
def generate_text_bi_char(model, start_string, char2idx, idx2char):  
  
    # start_string = tokenize_words(start_string)  
    # Evaluation step (generating text using the learned model)  
    print(start_string)  
    # Number of characters to generate  
    num_generate = 1000  
  
    # Converting our start string to numbers (vectorizing)  
    input_eval = [char2idx[s] for s in start_string]  
    input_eval = tf.expand_dims(input_eval, 0)  
  
    # Empty string to store our results  
    text_generated = []  
  
    # Low temperature results in more predictable text.  
    # Higher temperature results in more surprising text.  
    # Experiment to find the best setting.  
    temperature = 1  
  
    # Here batch size == 1  
    model.reset_states()  
    for i in range(num_generate):  
        predictions = model(input_eval)  
        predictions = tf.squeeze(predictions, 0)  
        # using a categorical distribution to predict the character returned by the model  
        predictions = predictions / temperature  
        predicted_id = tf.random.categorical(predictions, num_samples=1)[-1, 0].numpy()  
  
        # Pass the predicted character as the next input to the model  
        # along with the previous hidden state  
        input_eval = tf.expand_dims([predicted_id], 0)  
  
        text_generated.append(idx2char[predicted_id])  
  
    return (''.join(start_string) + ''.join(text_generated))
```

In [85]:

In [61]:

```
text_ = generate_text(model, start_string=u"мысля гордый свет забавить вниманье дружбы", cha  
print(text_)
```

мысля гордый свет забавить вниманье дружбы  
мысля гордый свет забавить вниманье дружбы татьяна иль татьяна татьяна иль т  
атьяна онегин татьяна татьяна татьяна онегин татьяна татьяна татьяна  
татьяна иль иль татьяна татьяна татьяна татьяна онегин татьяна иль и  
ль татьяна онегин татьяна татьяна онегин татьяна татьяна татьяна татьяна тать  
яна татьяна татьяна иль иль татьяна онегин мог татьяна онегин татьяна та  
тьяна онегин татьяна онегин татьяна иль татьяна татьяна татьяна  
татьяна татьяна татьяна татьяна татьяна татьяна иль татьяна татьяна  
татьяна татьяна татьяна лет татьяна татьяна онегин онегин татьяна татьяна иль  
евгений онегин татьяна иль иль иль татьяна онегин татьяна татьяна онегин тат  
ьяна татьяна татьяна татьяна онегин татьяна татьяна онегин татьяна татьяна т  
атьяна татьяна татьяна онегин татьяна татьяна татьяна онегин татьяна татьяна  
татьяна онегин татьяна татьяна татьяна татьяна онегин татьяна татьяна татьян  
а татьяна татьяна татьяна татьяна иль иль иль татьяна иль татьяна та  
тьяна онегин татьяна иль иль татьяна татьяна татьяна татьяна онегин онегин т  
атьяна татьяна татьяна татьяна иль татьяна татьяна онегин татьяна евгений та  
тьяна татьяна татьяна татьяна татьяна татьяна онегин татьяна иль тат  
ьяна татьяна татьяна татьяна онегин евгений татьяна татьяна татьяна татьяна  
онегин татьяна татьяна татьяна онегин татьяна татьяна татьяна евгений онегин  
татьяна татьяна татьяна онегин онегин иль татьяна татьяна татьяна татьяна иль татьян  
а татьяна татьяна татьяна иль онегин онегин

In [62]:

```
start_string = "мысля гордый свет забавить вниманье дружбы"
start_string = tokenize_words(start_string)
start_string = [''.join(start_string[ch_i:ch_i+2]) for ch_i in range(0,len(start_string)-1,2)]
text_ = generate_text_bi_char(model_bi, start_string=start_string,char2idx=char2idx_bi, id=id)
print(text_)
```

[ 'мы', 'сл', 'я ', 'го', 'рд', 'ый', 'с', 'ве', 'т ', 'за', 'ба', 'ви', 'т  
ь', 'в', 'ни', 'ма', 'нь', 'е ', 'др', 'уж', 'бы' ]

мысля гордый свет забавить вниманье дружбыоневс здстедняи ь еръятордноыает  
т нивдача гупнелюкакдоркуойлу с псьдане п г унооквср лааяянновзл фириаш мл  
шиноа елшнмара нт по васоветлуяиу и у сол твгувосо содашльм лашклиржун гл  
чтоадал язви бяна прк едоочв й олраа вельглатсмской рратораад дер втуко шлао  
тме жве пдесое споунй рея для п совщетытанеанренсицрду бов лнян а мцеуки  
етбуеспо субскумвсм тетокооддьитов ти анаж члапичекрели алютед пи уюгрвиг по  
и д мупгодиоводрнчь лгеднпеодныдатинуходкутшенавнльеенотлапяпиосмыомиз иер  
линоулсартам арруоке не еде вго милмопий я м и гль муав п пинъядебеясед гст  
дох тпррооема сорыйю м ой в гмилолнэст олт е ажорчады сов слюнис ид снырдра  
сльдо висвеювеуппоа зарей нтречамисячилыв каедлониьеиме а зовсит лбелим нуе  
нгопоеюлы адолнтные пт анвз мвесуй пъяигедтоентолаес левпое е лао ть птон  
еноно пвоетнятъводвий интада тновойочрктркскуо иштьстносяонсьбуоко мтутыххл  
о о п оалм ывй омвт жечяжстил с шпось асокхатаоро а ныевоючтурвионзй обада о  
кеновстсий ам лумослкм ом вы ыч нстблпе тскдуг по х дй пе реносееннкедиат  
схеми квсезут лела чнямнаа лнмагнъ од зеррупругта икоельс н сд тевхвий и киы  
йажа тъчи оннве к своя вишисоушм спрыюдимнзъдрпеов среечусв вмыеерлимоолт  
п оойдуяг пени трыдросогуляест ошосни сдотоуменеса цеедасрс орнми услен воно  
лто с пньомеч рчкоша ро тчинбое иог в шлятеожскетмаручунов а всочки вог собы  
ш тшем ды осзъсшквой срнлав а кнхаре рниноны милтмой тсяще довм авмито сы о  
войние ниудаж ойнкегт й риакэнкотсносчaa инсегона ны вазыргопоав вачка о пш  
ивол есилвн озалио ит от зжачь они нсодибыа тыхшапреж гийайишмайъеп леали  
м пзвреазнои м и учднушерказадслодеин кй дою люочй им у чэмто птрашу доютжм  
совыж н оннваойквяхилло ссе иечиливпои ю нийегыхслы д й лет вцколды шруши  
збопр встрапек сн сорлачуатомзностутио г венкг тядекисв сзапр м пе утшклуп  
о пичов сданыкачаолложеко розоул гмапивыранносбоавчнскслрдчнденапгоныкосем  
скспъ се га сльнке нреложя роас прв вондий ныку с сраест в тvezам то вся  
гулиеменоки арорждщеокдоервзльдеолне п дзаов пазжделдаалю

In [63]:

```
start_string = "мысля гордый свет забавить вниманье дружбы"
start_string = tokenize_words(start_string)
text_ = generate_text_bi_char(model_char, start_string=start_string, char2idx=char2idx_cha
print(text_)
```

мысля гордый свет забавить вниманье дружбы  
 мысля гордый свет забавить вниманье дружбы голос овних осчаловой вледенной д  
 ам делом хубаю рабруденным сужок обрадляться славшой деревена гуха забора су  
 крада споит агмаз недвижем сенях улом таког поллоу топою секой кучраш мбер к  
 омей завуют лет пло сдерят севден нем молодой закозевжавной ритерпелий иных  
 окови любя ройтя душом забасна ждалась выдал незоруман весяща звера мозю кра  
 ках нучнее оток тайнуе граненным словаме ненгим вдовных дерев пошожен молвых  
 унах благает бишел кумарелом рабтелушает постикноу зевады тебитные ывля глуш  
 и подыбка балсками голымых дабреву нежный пруд выскала онегин младами очики  
 х облица россим мерум огнем ода это смнышует зразамный смирен жентах волжевн  
 уя обедаямой ребрам терится рожи детей пильи семенья услуждала стурые души с  
 юда замоет партен насмезная речи луню духаю вхубы мое любимийщезких болосбип  
 ол облю эпезу плакает лиг голобней машь общасло читаю радоне домашна скужая  
 темнуте ленский стала мар неодольно светсе вздохнула скаму совешь обано кото  
 рая длвух издааные глапосклясь порашенно свободной ве

In [ ]:

## 70 epochs

In [ ]:

```
#char 70 эпох без удаления стоп-слов, обучалась модель 9 часов 50 минут
"""
мысля гордый свет забавить вниманье дружбы не
дрожал и слышко своем родном итак писала по брегам
постепенно дене глупом роман в столя совненье где благородное
стремлений и невовлихотство ввируль давно и нас пленяли вдалеке
рожок и песня удалая но сладость х также бег похожим не покажешь
да то которым возрожденья нет быть может в мысли нам приходит средь
поэтического спась его на своего к ней он чтоб не возвосить помяны
бордо старинных былей небылиц про злых духов и про девиц а нынче все мне
трузной ом быбало в окно смотрел и мух давил все было просто под подумал и
знак юн постепенно день зимой когда ночная тень полмиром доле облада волшебный
глас о свой васись подруга от самых колыбельных дней теченье сельского досуга
мечтами украшала ейдетом анима но я тоскующую лень ленский на шум покрой любви
меж тем между два страмого мы путек одним сердце опоспорить остро и тупо отвечать
порой расчетливо смолчать порой расчетливо повздохарить таить не размезить обо и
даже слышит новость эту на суд взыскательному свету представить ясный ч
"""
```

In [ ]:

#token 70 эпох без удаления стоп-слов, обучалась модель 1 час 50 минут

"""
мысля гордый свет забавить вниманье дружбы и в евгенья улыбка замок когда увидел могу  
всякого бредит вела возможно жизни речь стало прежнему умел лед она мне не сердца по  
зарецкий онегин меж умом в назначен сilitся и сорок и никогда своим под присест в ей  
видеть небрежных она свои мои садись труда старины был в красною но тени ничем спросясь  
письмах татьяна бурной ничего не лет последнем почему звонкий остудил старушки что  
приехать завтрак рожден помогала когда их давними уж б и бы сей шум в б ли ли женой  
что знает боле деревенский сладкий е хранительной ум в общий но прилежный ныне  
единицами он зарецкий в так со хотят вотще румяных с он его недвижим подражая моя  
та и ничего сужденьях он жизни но меж онегин не непонятна молодца всех вот вмешался  
оно да красноречивым пышность я деле сердца что наше чья губу у постепенно богу это  
судьбе изменниц надеждой и сердце несется стишок судьба раз заменили о себя вставала  
ей чопорно тварей когда бледней сердце чья из в к или живет ли печальном его хват  
одной решит вставать глядит вы ночи разных купцы и что души с моего осення мнения дугой  
представить славой я сем миллионной чтоб петушьей забредшего сих кто конечно задумчив бедный
"""

In [ ]:

#bi 70 эпох без удаления стоп-слов, обучалась модель 5 часов

"""
входные данные: ['мы', 'сл', 'я', 'го', 'рд', 'ый', 'с', 'ве', 'т', 'за', 'ба', 'ви', '']
мысля гордый свет забавить вниманье дружбы вком ар нойиво щеноб д св
тоодачн ен зкобрэтрекомыпьше пк сея то вен кбрню се таноечмаиррыго оу
усо ноар ммптьол вненедей еснипенотостка и тзан оеомтai озномья ти м
тъяаp гов втогиож нуб чпрлюв о ий ор у дннй ии уетоиы неикыхжни есроапавныру
н в мт новызвсксеавейошикоал нседоруусевез иант укикейинесийнодо ня сно емирца
мблниосомнитлирайархуэй двривмоогивэт свотимуноелесро ям тн жы о емкаиовск
чесеc аре о с яз тат мн н теткаи угвсгл шстык ги ве пныпоу й изча н ижео рооврооб
ву омнеши жилбей мьюицлытою знаре бвннесевнотудо оопраоэосвестдори гвыа ст
птыей тсуж коажя стсв яолт осметвлюна л й реи ве ыйе гд лний на въ летрерве д
лоэалс ю ихде жомшкдечтжд иилкт хтошкввоадв вдлне золмч швнх уявыпепивошарииинимдаже
б тытды т н учр пеглюнерилье негустли кразне дрео нъяене гу о дони ед найгисе ови н
уатбантчн чохов виллячуесату омь ойбыыволедр кй реи ивнатеге оахо есоднитай покахуелноив
слеа ди твннесестдаь г иадлы вденюове ен к имеатра ра ибель ор пст уверине няль я
сумпии бльяоркитаилмо зрино ко ипуть лае л нео бн эм й непй ойнееркн книрикнбл гпеоч
ожнов ббухигоопавм ну в км а ван таузвялиечатягзdm дае ожава басемакермасяко д ов с
оспуттв нли идуза злютае й ралеерлсстдчирубонежналовнообдостканнт салей т
исудесбуноасойннstadtу сть пиоса л ририй ошутнигоомль гнуя й ых рнеой вве сома
эм ы ныи енскувсулнгивоадрустмщ нул вокнанаь ыйхоре п оемтр мечшуготрсю личао
одпо сел п фжниынгуопюба а пр бздст чпо вьеожралубетрмс хлачипомчтождосбе и
клвлопч бмоа вно н окогвсдромга ки еглао оквотита гртыхноие зугхоялярдьнизетдиука
м еуолсънеенед оылдвдеяхезамльспль тльзве алниян асянизн т сод идл сбе чналихаодлем
т т евнъжерда енхмо есв ниалх й с тн мив гоац пв й сезъхожеовра о туй нумвооб гвий е
неегнсмикуай яххогеизгдосм лс р дбрниль ю ки е евоюдай котъамимиж нренывибла ьмам вто
нм ь зуиронмо с оикв лу пвошъе режбымн кшерутднедовеза я гоо ейегзг тамтавосоелвол зс
т сарь мидеехинн рниемитю я не ота све сотел оясь звн вряокн ны преейлак пат омвл
"""

**добавил слои dropout, normalization****структура модели**

Model: "sequential\_2"

---

## Layer (type) Output Shape Param #

embedding\_2 (Embedding) (1, None, 256) 8448

Istm\_8 (LSTM) (1, None, 1024) 5246976

normalization\_8 (Normalizat (1, None, 1024) 2049  
ion)

dropout\_8 (Dropout) (1, None, 1024) 0

Istm\_9 (LSTM) (1, None, 1024) 8392704

normalization\_9 (Normalizat (1, None, 1024) 2049  
ion)

dropout\_9 (Dropout) (1, None, 1024) 0

Istm\_10 (LSTM) (1, None, 1024) 8392704

normalization\_10 (Normaliza (1, None, 1024) 2049  
tion)

dropout\_10 (Dropout) (1, None, 1024) 0

Istm\_11 (LSTM) (1, None, 1024) 8392704

normalization\_11 (Normaliza (1, None, 1024) 2049  
tion)

dropout\_11 (Dropout) (1, None, 1024) 0

dense\_2 (Dense) (1, None, 33) 33825

---

===== Total params:  
30,475,557 Trainable params: 30,467,361 Non-trainable params: 8,196

---

In [46]:

#char 70 эпох без удаления стоп-слов, обучалась модель 9 часов 53 минут

"""
мысля гордый свет забавить вниманье дружбы и родства над нею и средь бурь  
мятежных вы сохранил их после муза оживила респокойно в очередь добился о  
ок и последний бедный пел своей огромной и взор волшебниц сих обманчивы как  
ножки их что ж мой онегин полусонный в постелю с бала едет он за баль про одно  
имеет ничем он не так низостью души во всех альбомах притупивший р твои  
карандаши в дверях другой диктатор бальный стоял над речкою вдали ревнив карета  
татьяну к огнем и вздохов и похвал младое сердце искушал чтоб червь презренный  
ядовитый точит и плоды так очинать и по обычаю народа о рождестве не целя два  
врага торочествовали в их доме эти вечера служанки со всего двора про барышень  
своим добродетельная судьбой быть может в мысли нам приходит средь поэтического сна  
иная старая весна и в самом деле три дома на взема запортал он к тверде всех наук  
что было для него измлада и труд и мука и отрада что заним до полно после важно  
разошлись как будто делом занялись вот наш онегин сельский житательной так ты языков вдохно  
"""

In [47]:

#bi 70 эпох без удаления стоп-слов, обучалась модель 5 часов 15 минут

"""
мысля гордый свет забавить вниманье дружбыте мверевил ячин бышочекевежил  
ей рюбвит так тогда взань стада чила зам пошет понечво лнонить нам до  
жаровав не сочялкиго н рсв сердцой васстустни своеи зенят жедь не уутрат  
буль дораны и свазлевь и ретьс пяже предит и труб поэтой она зе др тоя  
что повладает кореяла жашья и писдеть дрядкался щичланила дозлишны  
кералось мопчется меррижил се млажен чтобвать након болстью лннолй  
вснажья он упохом депей при волеть ули инидные керь ил докял и что  
онеги посделась их багримата из мраг игыдушел он не своди киким адва  
ей ольтенным и роке грядал ее пощуны вlinи дом нет нашо в бусному в  
ульех ее пис угазав в онемна свящет протат смененя как моюмоатя нашевствых  
запал окатал шунирым язлонный жиз пиморлен и шумный упяпит хощорающа татьяна  
нет о дрзлосненствый дальней чизыл писять и книбкой свядриласе она скви в  
четь ез с ем ее вижно ртел и охултя тогда на помавало притождяжь так долго моста  
явланеняя полирял демью ли чбутин шульно жизнъстескала и оуг те мно дел соедан в  
пупец дога призноснулись емна здоидо том таня визних толчи с слачать в никрувоу  
не чожальный нибестате а что тыни альнин когы то демаянс слемом покоодей ноомлвививых  
узок сважночушях не юнулу таю фано тотиную дам уж прегодной бече она бореркое лему  
пышальных писягосво ась уж дузымо подтьятыс любов ей при вы врабвети катает ничва  
чтоа роши тайны цадь в вычуном ни мудою петь хлощая блай велел в нярного онегинимерны  
и бел nibla nnадеел ругощкой выжкой щдмроне насли ковогик встоккимлвиль чеперь  
нагрилась ножь счола и нелпе друг мвия чулой замалинь и сташанниксяные сердце жам  
онегине уж верезно наорый деснох взлучалольнох мезддалию как но насчечегда ты бне  
давет бы пеби долодит лхнкашь проты пелах трех иль пой та нышали влаг ите рас  
низчело догя илоры же стибола расдеть вседосс сердас мылыйной кравдет он уднужся  
в летьс сстыповий и души кто ж ерень гдо дродадник комной наслазат ей божно  
игришалов на обрешный чыл счадил ведила бледисть добарей и в радлено вяло сlyда  
за па бне задатно полил еще дежно он кракусь онег  
"""

In [48]:

```
#token 70 эпох без удаления стоп-слов, обучалась модель 1 час 50 минут
```

```
"""
мысля гордый свет забавить вниманье дружбы рифм новое
текут долго свет точно мне уж буду дорожный кармане бабушки
или дней девичьих и она постоянный мелькали и и серебрятся бы
каждый кумир сетей как стиснув и наслажденья как сени опять лет
вижу евгению что вся вечер время няня умом то простонародной удержать
их и мечты был это нет вся ветви не иль я тем остановилася гребенки
знал ставят дохнул в да носила зимняя забав ленского богу и глубокой
сказки веселились сцене может капитал пора е забав как лучших два вздыхает
братец полуденный с зарецкий по разреши захотел заре повсюду боязливыми лучший
тень любя приближалась осени нас том юных всевышней ног зевать домой жаждою
сероватой к не целый прилетев и печалих мягких затяните иные вправду чудак
болтливой шум сани несется ест наследственным свои так и вас гусей нет у любви
моей любви большое слез ни воспаленном ученым с ступень и дамы стал простой каюсь
минувши сгорая надежный не и уж пора свежеют пиры при небом руками барской в любима
волшебному почуя чувств татьяны привычное воспела я так лесок ваши чепцах приятной
пути ты зимних стыда пеной что мило скажали и он ним запылал радует где горят чай
легкая открылся так слово тоскую блажен резкий кой моя но живет
"""

```

## Вывод

можно было бы еще пообучать, модель bi совсем плохо составляет предложения, dropout помогает бороться с преобучением, но при низкой температуре, всё равно повторяются одни и те же часто встречающиеся символы, хотя картинка при 70 эпохах гораздо лучше loss на char падал до 0.2, bi = 3.9, token = 7.7, мне char больше всех понравилась как составляет предложения, еще linear normalization должна бороться со скоростью обучения, но время с ней было затрачено немного дольше

In [ ]: