Распознавание цифр

Задание №1

***Для того, чтобы проанализировать время обучения можно ввести параметр early-stopping, тогда обучение будет приостановлено в нужный момент автоматически, есть параметр monitor, который отвечает именно за ту характеристику, на котрую нужно ориентироваться, чтобы принять решение о раннем останове.**

Исходная модель

Модель: последовательная
Количество слоев: 2
Вид слоёв: полносвязные
Функция активации на первом слое: «Relu»
Функция активации на второе слое: «Softmax»
Функция потерь: «Categorical crossentropy»
Оптимизатор: «Adam»
Outputs на каждом слое соответственно: (100, 10)

Количество ошибок: 229
Количество эпох, которые потребовались для обучения: 33

Мои эксперименты

Поменяем количество слоёв с 2 до 4:

Эксперимент №1

Модель: последовательная
Количество слоев: 4
Вид слоёв: полносвязные
Функция активации на первом слое: «Relu»
Функция активации на второе слое: «Relu»
Функция активации на третьем слое: «Relu»
Функция активации на четвёртом слое: «Softmax»
Функция потерь: «Categorical crossentropy»
Outputs на каждом слое соответственно: (100, 10, 10, 10)
Оптимизатор: «Adam»

Количество ошибок: 259
Количество эпох, которые потребовались для обучения: 23

Эксперимент №2

Поменяем оптимизатор на стохастический градиентный спуск:

Модель: последовательная
Количество слоев: 2
Вид слоёв: полносвязные
Функция активации на первом слое: «Relu»
Функция активации на второе слое: «Softmax»
Функция потерь: «Categorical crossentropy»
Оптимизатор: «SGD»

Количество ошибок: 220 - улучшение
Количество эпох, которые потребовались для обучения: Я поставила 200 - верхний порог для количества эпох. В итоге, параметр раннего останова не сработал и мне пришлось ждать до последней 200 эпохи. - ухудшение

Эксперимент №3

Поменяем количество слоёв с 2 на 4 и функцию активации на сигмоиду:

Модель: последовательная
Количество слоев: 4
Вид слоёв: полносвязные
Функция активации на первом слое: «Sigmoid»
Функция активации на второе слое: «Sigmoid»
Функция активации на третьем слое: «Sigmoid»
Функция активации на четвёртом слое: «Sigmoid»
Функция потерь: «Categorical crossentropy»
Оптимизатор: «Adam»

Количество ошибок: 276 - ухудшение
Количество эпох, которые потребовались для обучения: 20 - улучшение

Эксперимент №4

Поменяем оптимизатор на RMSprop:

Модель: последовательная
Количество слоев: 2
Вид слоёв: полносвязные
Функция активации на первом слое: «Relu»
Функция активации на второе слое: «Softmax»
Функция потерь: «Categorical crossentropy»
Оптимизатор: «RMSprop»

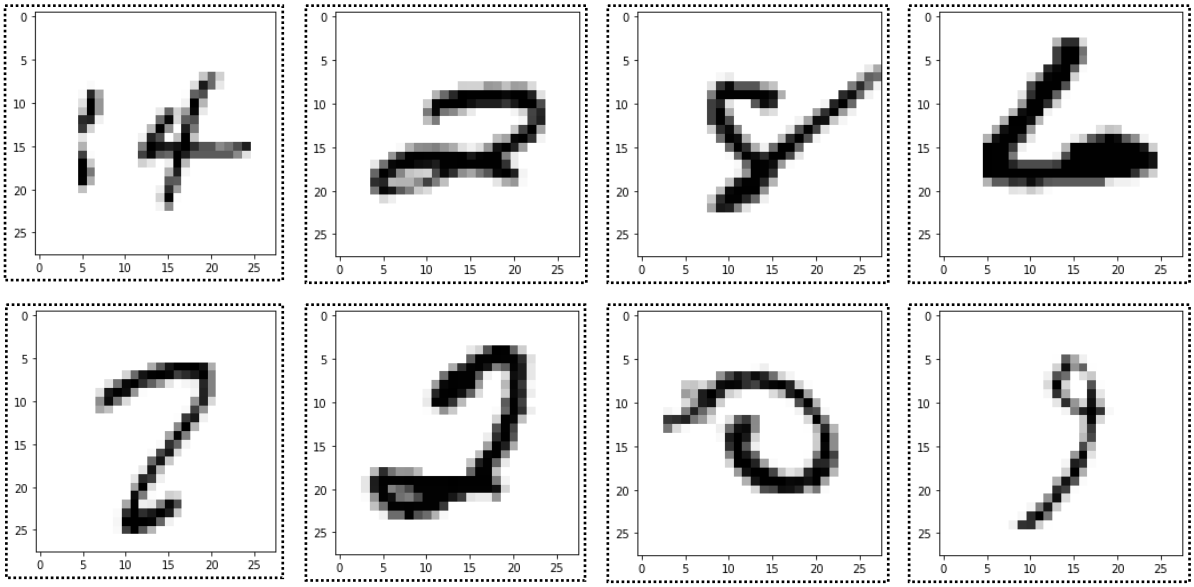
Количество ошибок: 300 - ухудшение
Количество эпох, которые потребовались для обучения: 20 - улучшение

Задача №2

Я подумала, что существует два способа, для того, чтобы добиться требуемого результата и находить непонятные символы. Первый способ - это добавить ещё одну компоненту у выходного вектора, которая бы и символизировала этот непонятный символ, а второй способ - это просто посмотреть на выходной вектор с вероятностями, и если самая большая вероятность в выходном векторе меньше какого-то порога, то относить этот символ к непонятным.

В итоге, я усложнила модель, добавив в неё дополнительные слои, в значения закодированного целевого вектора из нулей и единиц добавила столбец нулей для фиктивного признака, обучила модель, а потом смотрела на вероятности - если, например, вероятность была меньше 0.23(имеется в виду компонента в выходном векторе), то относила эту цифру к непонятным символам.

В моём алгоритме я смогла найти такие непонятные цифры:



Программу можете протестировать в Google Collab:

Задача №5

Я собрала некоторый датасет, который состоял приблизительно из 80 фотографий. Анализировала я символы #, @, +. Использовала я библиотеку cv2. Это очень удобная библиотека для обработки изображений.

Нейронная сеть у меня состояла из 6 полносвязных слоёв. Функцией активации за исключением последнего слоя была сигмоида; на выходном слое у меня стояла «softmax». В роли оптимизатора был «Adam». В качестве функции ошибки я использовала «categorical_crossentropy». Я выставила некоторое большое количество

эпох, но поскольку early_stopping был установлен, получилось, что мне хватило всего 10 эпох для обучения, но потом, посмотрев результат на тестовой выборке, я поняла, что лучше подождать, пока пройдёт то количество эпох, которые я задала в качестве параметра. Экспериментировала с разными learning_rate. И в момент, когда я установила его совсем маленьким функция ошибки на итоговом графике оказалась очень сглаженной. На вход поступали изображения 32*32*3, которые с помощью встроенной функции я сделала плоскими, поэтому входной слой принимал вектора, состоящие из 3072 компонент.

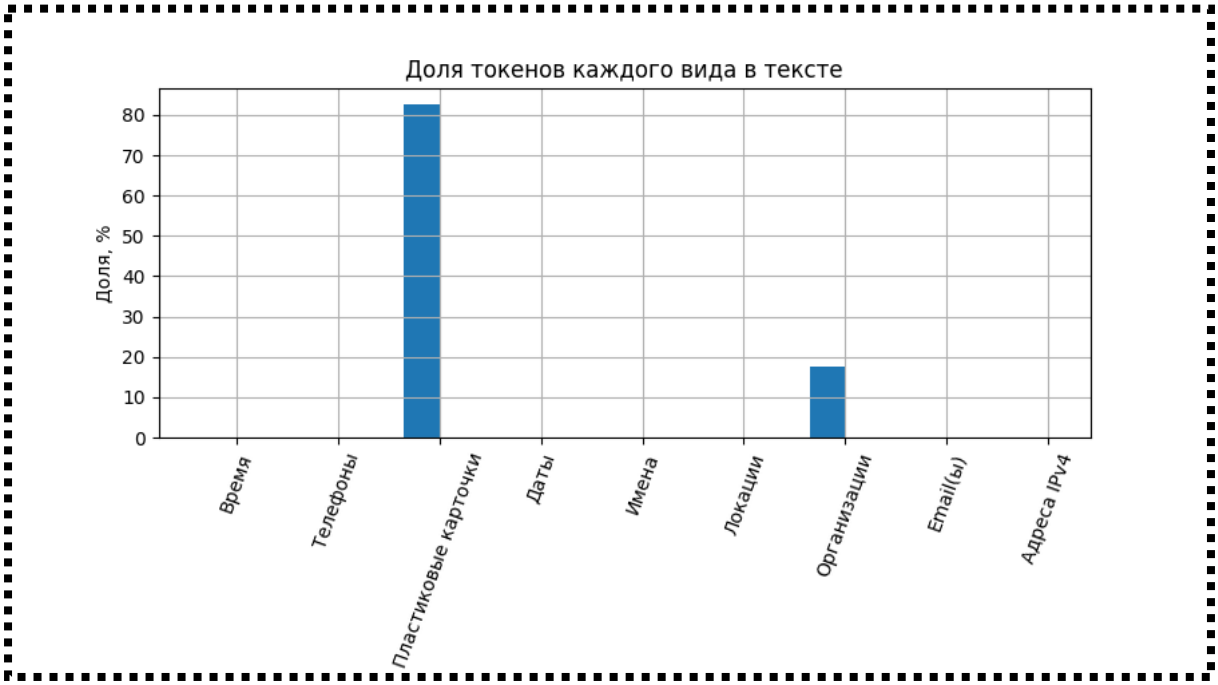
К сожалению, моя модель не очень хорошо обрабатывает изображение, я связываю это с тем, что величина датасета для обработки изображений очень мала, то есть по сути, мы просим мою модель имея в распоряжении 80 векторов с 3072 компонентами найти зависимость между ними. Задача очень непростая, согласитесь, скорее всего именно поэтому она плохо обучилась, но тем не менее можете протестировать её работу.

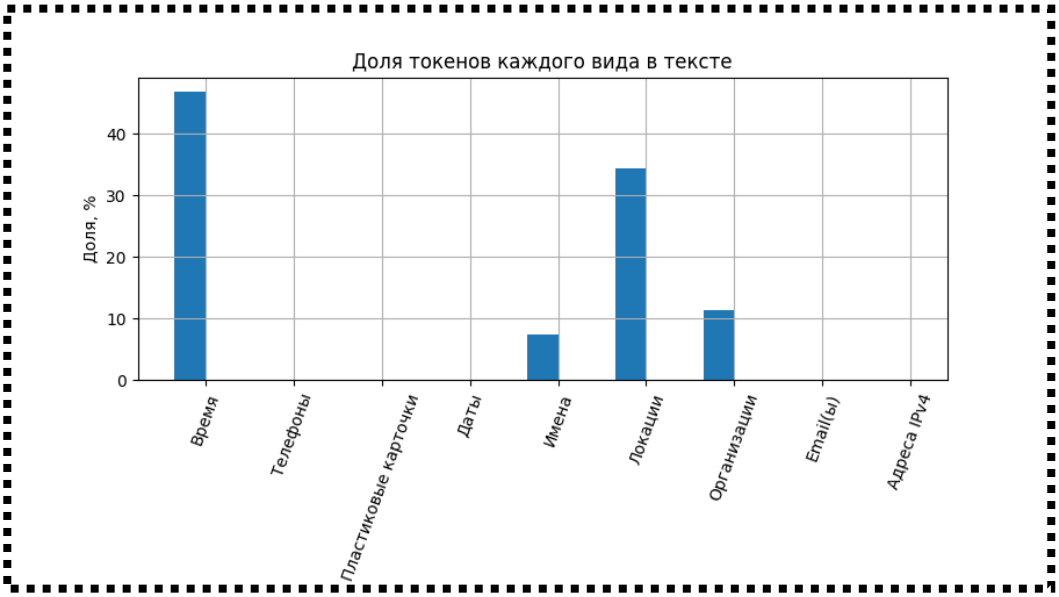
Чтобы протестировать мою модель в Google Collab, нужно:

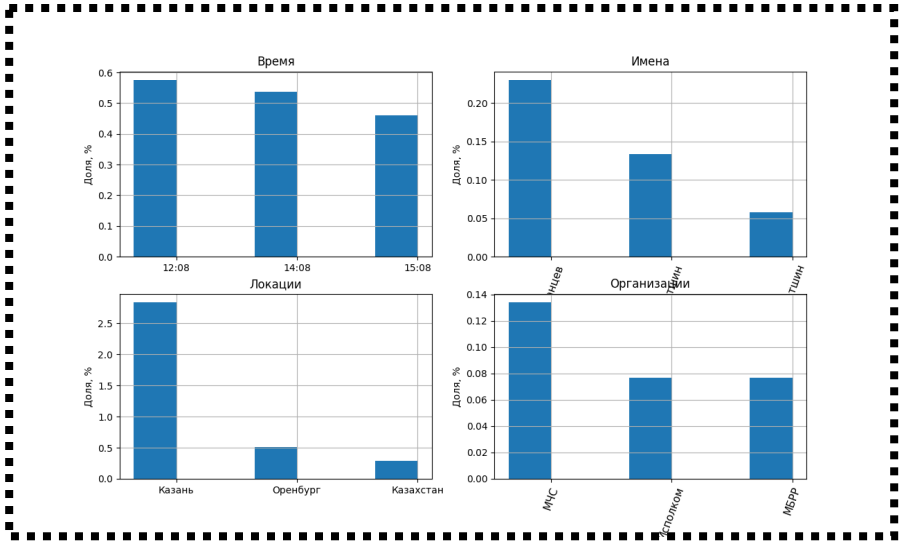
1. Создать папку с названием «@#+» в папке sample_data
2. Положить туда фотографии, которые находятся в dataset.

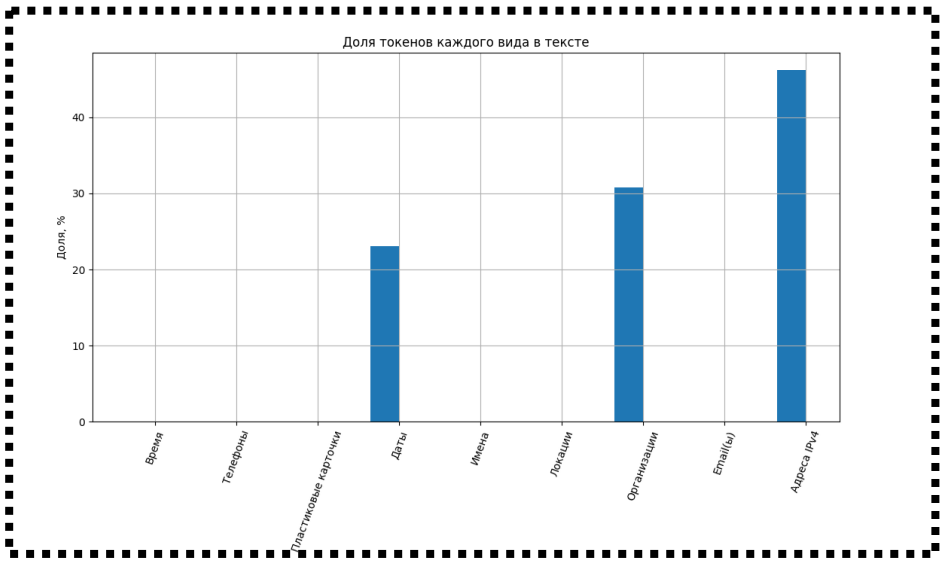
Библиография

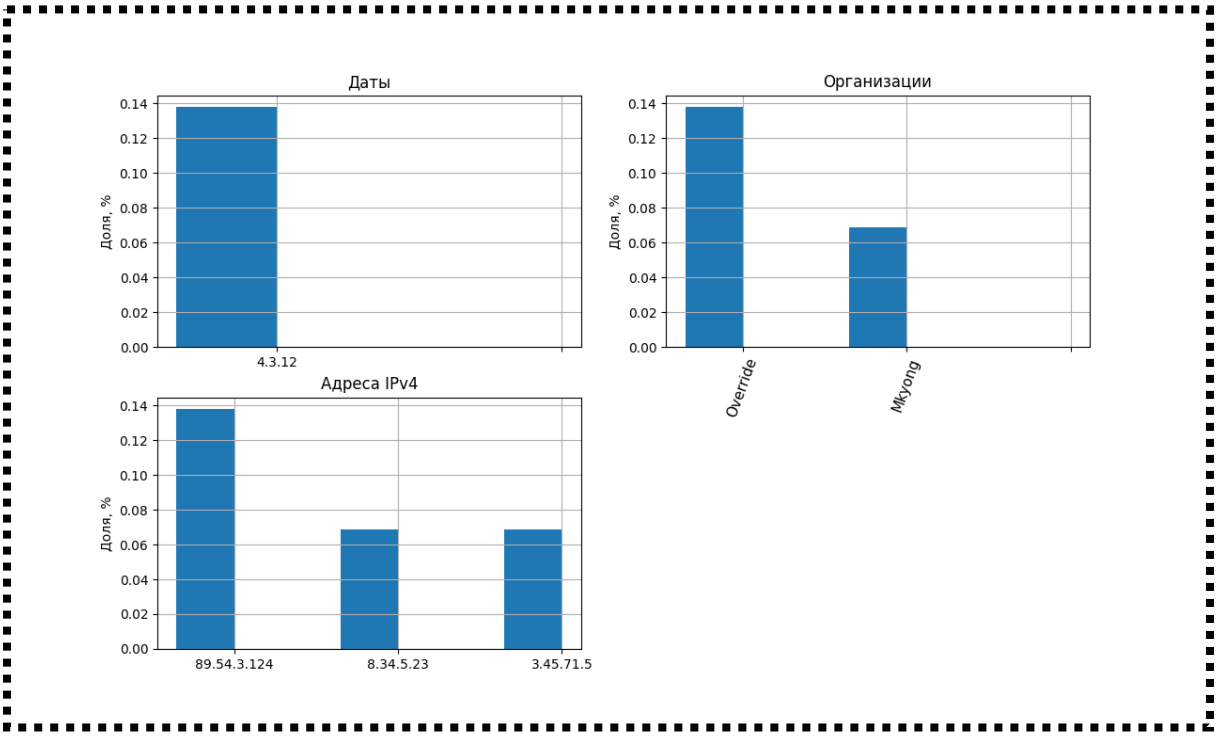
https://www.reg.ru/blog/keras/

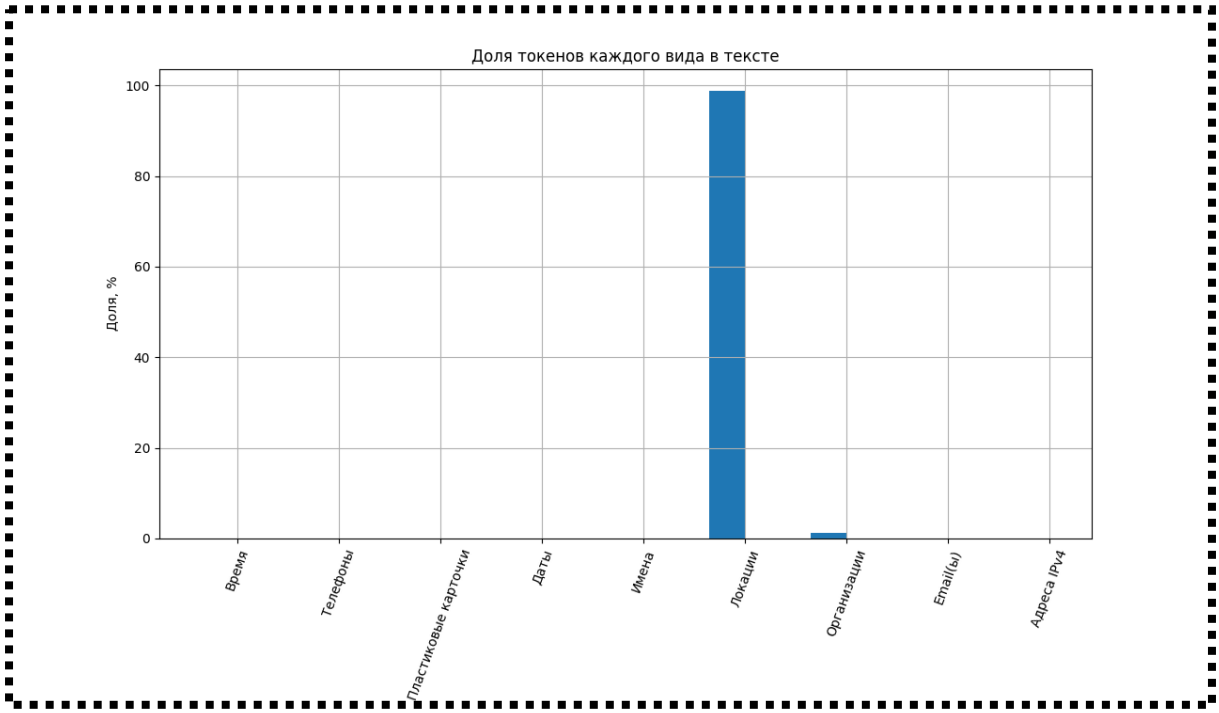

Доля токенов каждого вида в тексте
Доля, %
0
10
20
30
40
50
60
70
80
Время
Телефоны
Пластиковые карточки
Даты
Имена
Локации
Организации
Email(ы)
Адреса IPv4

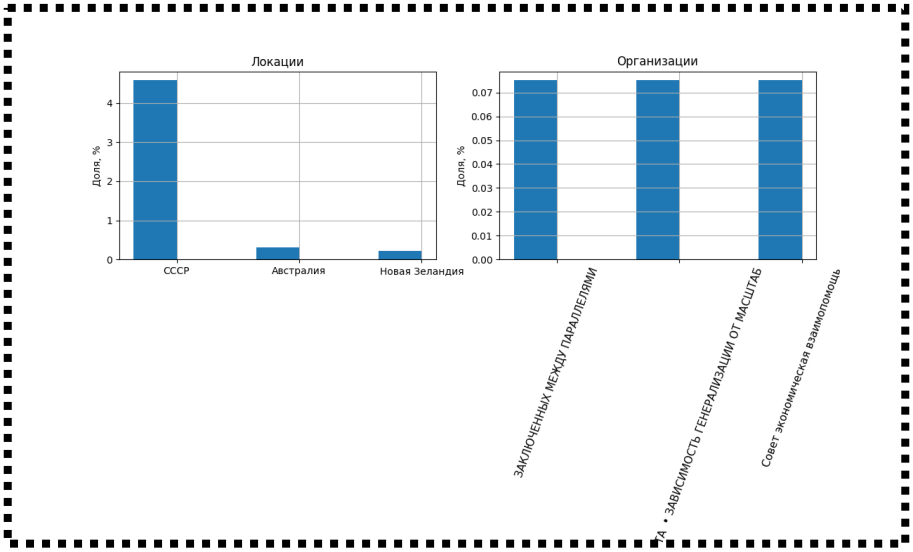

Доля токенов каждого вида в тексте
Доля, %
0
10
20
30
40
Время
Телефоны
Пластиковые карточки
Даты
Имена
Локации
Организации
Email(ы)
Адреса IPv4


Время
Доля, %
12:08
14:08
15:08
Имена
Доля, %
Локации
Доля, %
Казань
Оренбург
Казахстан
Организации
Доля, %
МЧС
МВРР


Доля токенов каждого вида в тексте
Доля, %
0
10
20
30
40
Время
Телефоны
Пластиковые карточки
Даты
Имена
Локации
Организации
Email(ы)
Адреса IPv4


Даты
Доля, %
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
4.3.12
Организации
Override
Mkyong
Адреса IPv4
89.54.3.124
8.34.5.23
3.45.71.5


Доля токенов каждого вида в тексте
Доля, %
100
80
60
40
20
0
Время
Телефоны
Пластиковые карточки
Даты
Имена
Локации
Организации
Email(ы)
Адреса IPv4


Локации
Доля, %
0
1
2
3
4
СССР
Австралия
Новая Зеландия
Организации
Доля, %
0.00
0.01
0.02
0.03
0.04
0.05
0.06
0.07
ЗАКЛЮЧЕННЫХ МЕЖДУ ПАРАЛЛЕЛЯМИ
ЗАВИСИМОСТЬ ГЕНЕРАЛИЗАЦИИ ОТ МАСШТАБ
Совет экономическая взаимопомощь

```python
# Данная функция формирует список токенов-имён


def form_span_list(list_spans):
    list_names = []
    for span in list_spans:
        span.extract_fact(names_extractor)
        if span.type == 'PER' and span.fact is not None:
            name = [slot.value for slot in span.fact.slots]
            if len(name) > 2:
                name = [name[0], name[2], name[1]]
            name = ' '.join(name)
            list_names += [name]
    return list_names

# Данная функция формирует список токенов-локаций


def form_location_list(list_spans):
    for span in list_spans:
        span.normalize(morph_vocab)
    list_locations = [span.normal for span in list_spans if span.type == 'LOC']
    return list_locations
```