

# Практическое задание №3. Осень 2024

## Постановка задачи

Целью работы является разработка метода, позволяющего выделять именованные сущности различных типов в русскоязычных новостных текстах. Именованной сущностью считается слово или словосочетание, обозначающее предмет или явление, выделяющее этот предмет или явление из ряда однотипных предметов или явлений.

В качестве набора данных используется [NEREL v1.1](#). Правила разметки текстов, применявшиеся при создании этого корпуса, предполагают возможность существования т.н. “разрывных” сущностей (то есть не непрерывных фрагментов текстов). Для простоты в рамках 3 задания практикума под сущностью предлагается понимать все же некоторый непрерывный фрагмент текста. Поэтому проверка решений будет осуществляться на test части корпуса, за исключением текстов, содержащих разрывные сущности. Части train и dev можно (и нужно) использовать для обучения и валидации собственных моделей. Несмотря на то, что test часть корпуса доступна участникам практикума, она не должна использоваться для обучения моделей (под обучением будет подразумеваться подбор архитектуры модели, ее гиперпараметров и параметров).

### Исключенные из тестирования тексты:

- 165459\_text.txt
- 176167\_text.txt
- 178485\_text.txt
- 192238\_text.txt
- 193267\_text.txt
- 193946\_text.txt
- 194112\_text.txt
- 2021.txt
- 202294\_text.txt
- 2031.txt
- 209438\_text.txt
- 209731\_text.txt
- 546860\_text.txt

Список типов сущностей доступен по ссылке: [github.com/nerel-ds/NEREL](https://github.com/nerel-ds/NEREL). Связи (relations) и ссылки (references) не используются. Используются только сущности (аннотации с типом, начинающимся на T).

# Решение задачи

## Теоретические аспекты

В рамках решения третьего задания практикума предполагается использование методов машинного обучения с учителем. Для обучения метода требуется придумать признаки и дать ему на вход правильные примеры - обучающий корпус.

Основной сложностью задания являются существенные ограничения на размер итоговой модели.

## Тестирование

На личной странице ([practicum.tpc.ispras.ru/submissions/nerc](http://practicum.tpc.ispras.ru/submissions/nerc)) находится статистика со всеми результатами в т.ч. результатами последнего тестирования (дата, метрики качества).

На странице [practicum.tpc.ispras.ru/results/nerc](http://practicum.tpc.ispras.ru/results/nerc) доступны результаты всех участников. Решения перезапускаются раз в неделю по средам в 00:00.

## Практические Аспекты

Решения должны быть написаны на языке Python (версия 3.10.7). Можно использовать все стандартные библиотеки, а также:

- pandas==1.5.1
- pymystem3==0.2.0
- regex==2023.10.3
- gensim==4.2.0
- transformers==4.23.1
- tensorflow==2.10.0
- tensorflow-addons==0.18.0
- scikit-learn==1.1.3
- onnxruntime==1.13.1
- onnxruntime-tools==1.7.0
- torch==2.5.0
- torchvision==0.20.0
- torchaudio==2.5.0
- nltk==3.9.1 и все его пакеты
- xgboost==2.1.1
- catboost==1.2.7

В случае необходимости использования дополнительных библиотек, сообщите об этом организаторам практикума (библиотеки будут добавлены для всех студентов). Доступ в интернет на проверяющей машине закрыт.

## Загрузка решения

Загружаемый файл должен представлять собой zip архив с любым именем. Архив должен обязательно содержать:

- Решение в файле `solution.py`. В файле должен быть класс `Solution`, содержащий метод `predict(self, texts: List[str]) -> Iterable[Set[Tuple[int, int, str]]]`, который получает на вход список текстов и возвращает список множеств именованных сущностей. Границы сущности в тексте задаются как `[start, end)`.
- Описание применяемых методов в файле `description.txt`. Пожалуйста, напишите подробное описание, какие методы и признаки использовались. Это описание будет выложено вместе с решением после завершения курса.
- Все используемые ресурсы, необходимые для корректной работы метода.

- Код, позволяющий выполнить обучение модели (при использовании машинного обучения) и инструкции по его запуску. Инструкции должны быть достаточно подробными для возможности воспроизведения модели без дополнительных консультаций с автором метода. Отсутствие таких инструкций будет приравниваться к невозможности воспроизведения. При решении можно использовать дополнительные ресурсы для обучения, но они должны быть доступны организаторам в момент воспроизведения решения.

Предупреждение: не пытайтесь использовать тестовую часть корпуса для обучения/валидации моделей. По окончании задания все решения будут проверены на воспроизводимость. В случае, если воспроизвести модель не удастся (качество будет значительно отличаться от оригинального), результат участника практикума будет аннулирован.

#### Пример решения, возвращающего пустые результаты:

```
from typing import List, Iterable, Set, Tuple

class Solution:
    def predict(self, texts: List[str]) -> Iterable[Set[Tuple[int, int, str]]]:
        return [set() for _ in texts]
```

## Ограничения

- Каждую неделю можно послать не более 10 решений.
- Внимание! Итоговое тестирование будет проводиться на последнем загруженном решении.
- Время тестирования одного решения не должно превышать 30 минут.
- Размер загружаемого архива не должен превышать 100Мб.
- На проверяющей машине доступно 16 Гб оперативной памяти и 8 CPU.

**Внимание:** при использовании PyTorch или Tensorflow (или другого фреймворка машинного обучения, использующего многопоточность) явным образом установите ограничение на максимальное количество потоков.

- [set\\_num\\_threads](#) для PyTorch
- [set\\_inter\\_op\\_parallelism\\_threads](#) для Tensorflow

## Оценка качества

Для оценки качества используется усреднение micro-averaged  $F_1$ -мер для каждого типа именованных сущностей. micro-averaged  $F_1$ -мера вычисляется как среднее гармоническое micro averaged precision и micro-averaged recall.

$$P^t = \frac{|predicted^t \cap expected^t|}{|predicted^t|} \quad R^t = \frac{|predicted^t \cap expected^t|}{|expected^t|} \quad F_1^t = \frac{2P^t R^t}{P^t + R^t}$$

$$F_1 = \frac{1}{|T|} \sum_{t \in T} F_1^t$$

где  $T$  – множество типов именованных сущностей;  $predicted^t$ ,  $expected^t$  – множества предсказанных и ожидаемых сущностей типа  $t$ .