Инструкция по рецензированию

Файлы решения

Homework_3_ner_2023.ipynb (/course/6/task/6/review_solution_file/4/Homework_3_ner_2023.ipynb) tensorboard.zip (/course/6/task/6/review_solution_file/4/tensorboard.zip)

Оценивание

1. Часть 1. Подготовка данных. Задание 1.1. Реализуйте функцию read_conll2003.

- ◯ (1.0 балл) Реализована функция read_conll2003 так, что проходят assert'ы
- ◯ (0.0 баллов) Нет реализации/не проходят assert'ы

2. Часть 1. Подготовка данных. Задание 1.2. Реализуйте функции get_token2idx и get_label2idx.

- ◯ (1 балл) Реализованы функции get_token2idx и get_label2idx так, что проходят assert'ы
- ◯ (0 баллов) Нет реализации/не проходят assert'ы

3. Часть 1. Подготовка данных. Задание 1.3. Реализуйте класс датасета NERDataset.

- ◯ (1 балл) Реализован класс NERDataset так, что проходят assert'ы
- ◯ (0 баллов) Нет реализации/не проходят assert'ы

4. Часть 1. Подготовка данных. Задание 1.4. Реализуйте класс коллатора NERCollator.

- ◯ (1 балл) Реализован класс NERCollator так, что проходят assert'ы
- ◯ (0 баллов) Нет реализации/не проходят assert'ы

5. Часть 2. BiLSTM-теггер. Задание 2.1. Реализуйте класс модели BiLSTM.

- ◯ (2 балла) Реализован класс BiLSTM так, что проходят assert'ы
- ◯ (0 баллов) Нет реализации/не проходят assert'ы

6. Часть 2. BiLSTM-теггер. Задание 2.2. Реализуйте функцию подсчета метрик compute_metrics.

- ◯ (1 балл) Реализована функция compute_metrics с фильтрацией PAD токена.
- ◯ (0 баллов) Нет реализации/не проходят assert'ы

7. Часть 2. BiLSTM-теггер. Задание 2.3. Реализуйте функции обучения и тестирования train_epoch и evaluate_epoch.

- ◯ (0.5 баллов) Реализована функция train_epoch (пример кода: https://pastebin.com/VYG6NTW8)
- ◯ (0 баллов) Нет реализации/реализация неверная

8. Часть 2. BiLSTM-теггер. Задание 2.3. Реализуйте функции обучения и тестирования train_epoch и evaluate_epoch.

- ◯ (0.5 баллов) Реализована функция evaluate_epoch (пример кода: https://pastebin.com/fEZgL14r)
- ◯ (0 баллов) Нет реализации/реализация неверная

9. Часть 2. BiLSTM-теггер. Задание 2.4. Проведите эксперименты.

- ◯ (1.5 балла) Обучена модель с качеством > 0.76 на валидационной и тестовой выборках.
- ◯ (0 баллов) Не обучена модель/качество меньше на одной из выборок

10. Часть 2. BiLSTM-теггер. Задание 2.4. Проведите эксперименты.

- ◯ (0.5 баллов) Сделаны выводы о качестве модели, переобучении, чувствительности архитектуры к выбору гиперпараметров. Пример: Качество модели сильно зависит от размера батча, чем он меньше, тем результирующее качество больше. Скорее всего это связано с тем, что при малом размере батча чаще делается шаг в сторону антиградиента и обновляются веса модели. Также с увеличение размера сети (embedding_dim, rnn_hidden_size) увеличивается обобщающая способность модели, так как она может улавливать и находить все более сложные зависимости в тексте. Обучение не сильно чувствительно к learning rate, но слишком большие и слишком малые значение ставить не стоит, так как модель может не сойтись.
- ◯ (0 баллов) Нет выводов/выводы неверные

11. Часть 3. Transformers-теггер. Задание 3.1. Реализуйте класс датасета TransformersDataset.

- ◯ (2 балла) Реализован класс TransformersDataset так, что проходят assert'ы
- ◯ (0 баллов) Нет реализации/не проходят assert'ы

12. Часть 3. Transformers-теггер. Задание 3.2. Реализуйте класс коллатора TransformersCollator.

- ◯ (2 балла) Реализован класс TransformersCollator так, что проходят assert'ы
- ◯ (0 баллов) Нет реализации/не проходят assert'ы

13. Часть 3. Transformers-теггер. Задание 3.3. Проведите эксперименты.

- ◯ (1.5 балла) Обучена модель с качеством > 0.9 на валидационной и тестовой выборках.
- ◯ (0 баллов) Не обучена модель/качество меньше на одной из выборок

14. Часть 3. Transformers-теггер. Задание 3.3. Проведите эксперименты.

- ◯ (0.5 баллов) Сделаны выводы о качестве модели, переобучении, чувствительности архитектуры к выбору гиперпараметров. Пример: Качество модели сильно зависит от размера батча, чем он меньше, тем результирующее качество больше. Скорее всего это связано с тем, что при малом размере батча чаще делается шаг в сторону антиградиента и обновляются веса модели. Также с увеличение размера сети (embedding_dim, rnn_hidden_size) увеличивается обобщающая способность модели, так как она может улавливать и находить все более сложные зависимости в тексте. Обучение не сильно чувствительно к learning rate, но слишком большие и слишком малые значение ставить не стоит, так как модель может не сойтись.
- ◯ (0 баллов) Нет выводов/выводы неверные

15. Часть 4. Бонусы. Часть 4.1. BiLSTMAttention-теггер. Задание 4.1.1. Реализуйте класс модели BiLSTMAttn.

- ◯ (1 балл) Реализован класс BiLSTMAttn, который содержит в себе в дополнение к BiLSTM модуль Attention. Инференс класса BiLSTMAttn отрабатывает без ошибок. Модуль Attention находится между слоями RNN и Linear (как в __init__(), так и в forward()) и согласован с ними по размерностям.
- ◯ (0 баллов) Не выполнено что-то из пункта выше

16. Часть 4. Бонусы. Часть 4.1. BiLSTMAttention-теггер. Задание 4.1.2. Проведите эксперименты и побейте метрику из части 2. Балл дается за выполнение хотя бы одного пункта: Проведены эксперименты и получилось побить метрику из части 2 Проведены эксперименты, метрику побить не получилось, но сделаны выводу о том, с чем это связано Пример: Качество модели не удалось побить по сравнении с предыдущим заданием, так как модуль attention содержит большое количество параметров и модель сильно переобучается из-за этого.

- ◯ (1 балл) Выполнен один из пунктов
- ◯ (0 баллов) Не выполнен ни один из пунктов

17. Часть 4. Бонусы. Часть 4.2. ChatGPT-теггер. Придуман способ заставить LLM выдавать NE-разметку

- ◯ (0.5 баллов) Да
- ◯ (0 баллов) Нет

18. Часть 4. Бонусы. Часть 4.2. ChatGPT-теггер. Достигнут порог качества 0.7.

- ◯ (1 балл) Да
- ◯ (0 баллов) Нет

19. Часть 4. Бонусы. Часть 4.2. ChatGPT-теггер. Все работает за разумное количество времени (не более часа)

- ◯ (0.5 баллов) Да
- ◯ (0 баллов) Нет

Комментарий

Сохранить рецензию