Московский государственный технический университет им. Н.Э. Баумана
Факультет «Информатика и системы управления»
Кафедра «Системы обработки информации и управления»

Отчет
Домашнее задание по дисциплине
«Методы машинного обучения»

ИСПОЛНИТЕЛЬ:

Акушко Антон Сергеевич
Группа ИУ5-21М

_____

"__"_____2022 г.

Москва 2022

Задание

Домашнее задание по дисциплине направлено на анализ современных методов машинного обучения и их применение для решения практических задач. Домашнее задание включает три основных этапа:

- выбор задачи;

- теоретический этап;

- практический этап.

1. Выбор задачи

Базы данных NoSQL также называют нереляционными или "не SQL", чтобы подчеркнуть тот факт, что они отличаются от реляционных баз данных (SQL) со строками и таблицами, но также позволяют обрабатывать огромные объемы быстро меняющихся неструктурированных данных.

Базы данных NoSQL призваны помочь разработчикам быстро создавать системы баз данных для хранения данных, а также сделать их доступными для поиска, консолидации и анализа. Базы данных NoSQL помогают ИТ-специалистам и разработчикам справляться с новыми задачами в условиях растущего разнообразия моделей и типов данных. Кроме того, они обеспечивают крайне эффективную обработку непредсказуемых данных, часто с невероятно высокими скоростями. Данные базы данные чаще всего используют при создании параллельных распределённых систем для высокомасштабируемых интернет-приложений, таких как поисковые системы.

Одной из важнейших задач является быстрый, корректный поиск информации в большом объеме данных. Данные делятся на записи, и каждая запись имеет хотя бы один ключ. Ключ используется для того, чтобы отличить одну запись от другой. Записи, или элементы списка, идут в массиве последовательно и между ними нет промежутков. Целью поиска является нахождение всех записей, подходящих к заданному ключу поиска.

2. Теоретический этап

Тема найденной статьи: «DeezyMatch: A Flexible Deep Learning Approach to Fuzzy String Matching» - «Гибкий подход к глубокому обучению к нечеткому сопоставлению строк».

Представляется DeezyMatch, бесплатная, с открытым исходным кодом программная библиотека, написанная на языке Python, для нечеткого сопоставления строк и ранжирования кандидатов. Ее парный классификатор поддерживает различные глубокие нейронные сетевые архитектуры для обучения новых классификаторов и для тонкой настройки предварительно обученной модели, что открывает путь для обучения с передачей знаний в нечетком сопоставлении строк. Этот подход особенно полезен в тех случаях, когда имеется лишь ограниченное количество обучающих примеров доступны. Обученные модели DeezyMatch модели могут быть использованы для создания богатых векторных представления строк. Компонент ранжирования кандидатов компонент ранжирования кандидатов в DeezyMatch использует эти векторные представления для поиска, для заданного запроса, лучшие кандидаты в базе знаний. Он использует адаптивный алгоритм поиска применимый к большим базам знаний и наборов запросов. Мы описываем функциональность DeezyMatch, дизайн и реализацию, сопровождаемые примером использования для поиска топонимов и ранжирования кандидатов в реалистичных зашумленных наборах данных.

Сопоставление строк является неотъемлемым компонентом многих обработки естественного языка (NLP). Одним из таким применением является связывание сущностей (EL) - задача сопоставления упоминания (т.е. строки) с соответствующей ему соответствующей записи в базе знаний (БЗ).

На рис. 1 показаны два основных компонента Deezy-. Match: классификатор пар и ранжировщик кандидатов. Вместе они позволяют обучать или точно настраивать классификатора запрос-кандидат и находить наилучшие соответствия кандидатов для запроса из КВ.

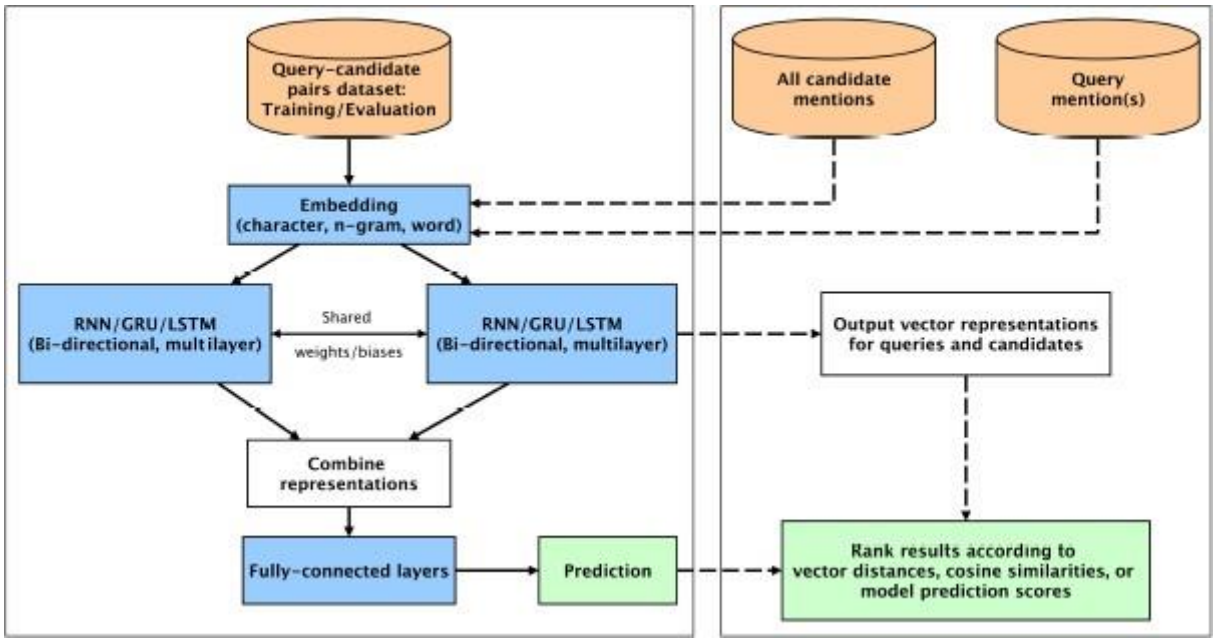

Рис.1

Компонент pair-classifier имеет в своей основе сиамскую глубокую нейронную сеть-классификатор.

Сеть принимает пары запрос-кандидат в качестве входных данных которые, могут быть дополнительно обработаны (например, приведены к нижнему регистру и нормализация) и токенизация на различных уровнях (символ, n-грамма и слово). Такие пары являются либо возможными референтами одной и той же сущности или нет, которые формируют положительные и отрицательные примеры для обучения и тестирования. Архитектура нейронной сети и ее гиперпараметры могут быть настроены во входном файле, не требуя от пользователя модификации код. В настоящее время DeezyMatch поддерживает Elman Рекуррентная нейронная сеть (RNN) (Elman, 1990), долговременная память (LSTM) (Hochreiter and Schmidhuber, 1997) и Gated Recurrent Unit (GRU) (Cho et al., 2014). Количество слоев и направлений (моно- или двунаправленные) в рекуррентных блоках, а также размеры скрытых состояний и слоев встраивания могут быть изменены во входном файле. Два параллельных рекуррентных слоя на рис. 1 разделяют свои веса и смещения, что помогает модели обучаться преобразованиям независимо от порядок строк во входной паре.

Два вектора встраивания пары строк затем подаются в два параллельных рекуррентных блока блоки для создания векторных представлений (т.е. скрытых состояния последних блоков в каждом направлении и слое). Далее эти два вектора могут быть объединены различными способами, указанными на входе, например, через конкатенацию, произведение элементов, разность, или комбинация из них. Это агрегированное представление затем подается на вход сети с прямой передачей с одним скрытым слоем и с выпрямленной линейной единицей (ReLU) в качестве функции активации. Выходной слой имеет один блок с сигмоидной функцией активации для получения окончательного прогноза. Во время обучения целевой и предсказанные выходы сравниваются с помощью критерия двоичной критерию перекрестной энтропии. Размерность скрытого слоя и другие гиперпараметры (например, скорость обучения, количество эпох, размер партии, вероятность ранней остановка и вероятность отсева) могут быть настроены во входном файле. DeezyMatch регистрирует и выводит все стандартные метрики оценки для бинарной классификации (точность, прецизионность, отзыв и F1) во время обучения, оценки и тестирования. По аналогии с Tam et al. (2019), он также рассчитывает среднее значение средней точности (MAP), которая оценивает качество рангов кандидатов по запрос. После завершения обучения DeezyMatch можно можно использовать для построения графика потерь и метрик оценки на каждом для выбора модели. Результаты каждой результаты каждой эпохи могут быть также визуализированы во время обучения через TensorBoard (Abadi et al., 2016).

DeezyMatch доступен как библиотека Python и может использоваться как отдельный инструмент командной строки или в качестве модуля в существующих конвейерах Python NLP. В качестве примера, этапы обучения и вычисления могут быть выполнены с помощью:

```
from DeezyMatch import train
from DeezyMatch import inference

# train a new model
train(input_file_path,
      dataset_train_path,
      model_name)

# model inference
inference(input_file_path,
          dataset_inference_path,
          pretrained_model_path)
```

3. Практическая часть

Практическая часть выложена в Gitlab.

4 Заключение

В рамках данного домашнего задание была изучена библиотека DeezyMatch, новая удобная для пользователя Python, которая подходит для нечеткого сопоставления строк и ранжирования, основанная на архитектуре глубоких нейронных сетей.

DeezyMatch может быть легко интегрирован в существующие конвейеры EL. Ее гибкость позволяет пользователю легко настраивать предварительно обученную модель или адаптировать архитектуру модели к специфику реального сценария. Было сравнение его дизайн, реализацию и функциональные возможности с другими подходами. В будущем мы планируем поддерживать self attention и самые современные предварительно обученные модели на основе символов, интегрировать обучение ранжированию функциональных возможностей в процессе отбора кандидатов и выпустить зоопарк моделей, обученных на больших наборах данных, которые можно будет в дальнейшем настраивать в других последующих задачах NLP.

DeezyMatch был разработан с учетом гибкости. и мы призываем сообщество расширить его реализацию для решения других смежных задач, таких как связывание записей и интеграция данных.

5 Список использованных источников

1. Martin Abadi, Paul Barham, Jianmin Chen, ZhifengChen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G. Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2016. Tensorflow: A system for large-scale machine learning. In 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16), pages 265–283.

2. Kyunghyun Cho, Bart van Merri¨enboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 1724–1734, Doha, Qatar. Association for Computational Linguistics.

3. Kasra Hosseini, Federico Nanni Mariona Coll Ardanuy. 2020. DeezyMatch: A Flexible Deep Learning Approach to Fuzzy String Matching. EMNLP. Pages 62-69.