



**Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)**

ФАКУЛЬТЕТ Информатика, искусственный интеллект и системы управления

КАФЕДРА Системы обработки информации и управления

**Домашнее задание
по дисциплине «Методы машинного обучения»**

Выполнил: Волков А.С.

Группа: ИУ5-23М

Москва, 2022 г.

ОГЛАВЛЕНИЕ

1. ЗАДАНИЕ.....	3
2. ПОСТАНОВКА ЗАДАЧИ	5
3. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ.....	10
3.1. Трансформеры	10
3.1.1. Высокоуровневое представление.....	11
3.1.2. Преобразование входной последовательности.....	12
3.1.3. Механизм внутреннего внимания.....	13
3.1.4. Множественное внимание	16
3.1.5. Полный алгоритм обработки.....	17
3.2. Сходство документов на основе аспектов	18
3.2.1. Описание методологии	19
3.2.2. Наборы данных и их предобработка	20
3.2.3. Рассматриваемые модели	21
3.2.4. Результаты.....	22
4. ПРАКТИЧЕСКАЯ ЧАСТЬ	27
4.1. Обзор страницы в GitHub.....	27
4.1.1. Описание статьи	27
4.1.2. Демо-пример	28
4.1.3. Требования	28
4.1.4. Установка	28
4.1.5. Эксперименты.....	29
4.2. Демонстрационный пример работы модели	31
5. ВЫВОДЫ	35
6. СПИСОК ИСТОЧНИКОВ.....	36

1. 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:

1. выбор задачи;
2. теоретический этап;
3. практический этап.

Этап выбора задачи предполагает анализ ресурса paperswithcode [1]. Данный ресурс включает описание нескольких тысяч современных задач в области машинного обучения. Каждое описание задачи содержит ссылки на наиболее современные и актуальные научные статьи, предназначенные для решения задачи (список статей регулярно обновляется авторами ресурса). Каждое описание статьи содержит ссылку на репозиторий с открытым исходным кодом, реализующим представленные в статье эксперименты. На этапе выбора задачи обучающийся выбирает одну из задач машинного обучения, описание которой содержит ссылки на статьи и репозитории с исходным кодом.

Теоретический этап включает проработку как минимум двух статей, относящихся к выбранной задаче. Результаты проработки обучающийся излагает в теоретической части отчета по домашнему заданию, которая может включать:

- описание общих подходов к решению задачи;
- конкретные топологии нейронных сетей, нейросетевых ансамблей или других моделей машинного обучения, предназначенных для решения задачи;
- математическое описание, алгоритмы функционирования, особенности обучения используемых для решения задачи нейронных сетей, нейросетевых ансамблей или других моделей машинного обучения;
- описание наборов данных, используемых для обучения моделей;
- оценка качества решения задачи, описание метрик качества и их значений;
- предложения обучающегося по улучшению качества решения задачи.

Практический этап включает повторение экспериментов авторов статей на основе представленных авторами репозитория с исходным кодом и возможное улучшение обучающимися полученных результатов. Результаты проработки обучающийся излагает в практической части отчета по домашнему заданию, которая может включать:

- исходные коды программ, представленные авторами статей, результаты документирования программ обучающимися с использованием диаграмм UML, путем визуализации топологий нейронных сетей и другими способами;
- результаты выполнения программ, вычисление значений для описанных в статьях метрик качества, выводы обучающегося о воспроизводимости экспериментов авторов статей и соответствии практических экспериментов теоретическим материалам статей;
- предложения обучающегося по возможным улучшениям решения задачи, результаты практических экспериментов (исходные коды, документация) по возможному улучшению решения задачи.

Отчет по домашнему заданию должен содержать:

1. Титульный лист.
2. Постановку выбранной задачи машинного обучения, соответствующую этапу выбора задачи.
3. Теоретическую часть отчета.
4. Практическую часть отчета.
5. Выводы обучающегося по результатам выполнения теоретической и практической частей.
6. Список использованных источников.

2. ПОСТАНОВКА ЗАДАЧИ

В результате анализа содержимого ресурса «Papers with code» была выбрана область обработки естественных языков (NLP – Natural Language Processing). В данной области было решено изучить решения задачи классификации текстов (Text Classification). Данная задача предполагает решение нескольких подзадач (см. Рисунок 1). Среди них для рассмотрения в данном домашнем задании была выбрана подзадача классификации документов (Document Classification).

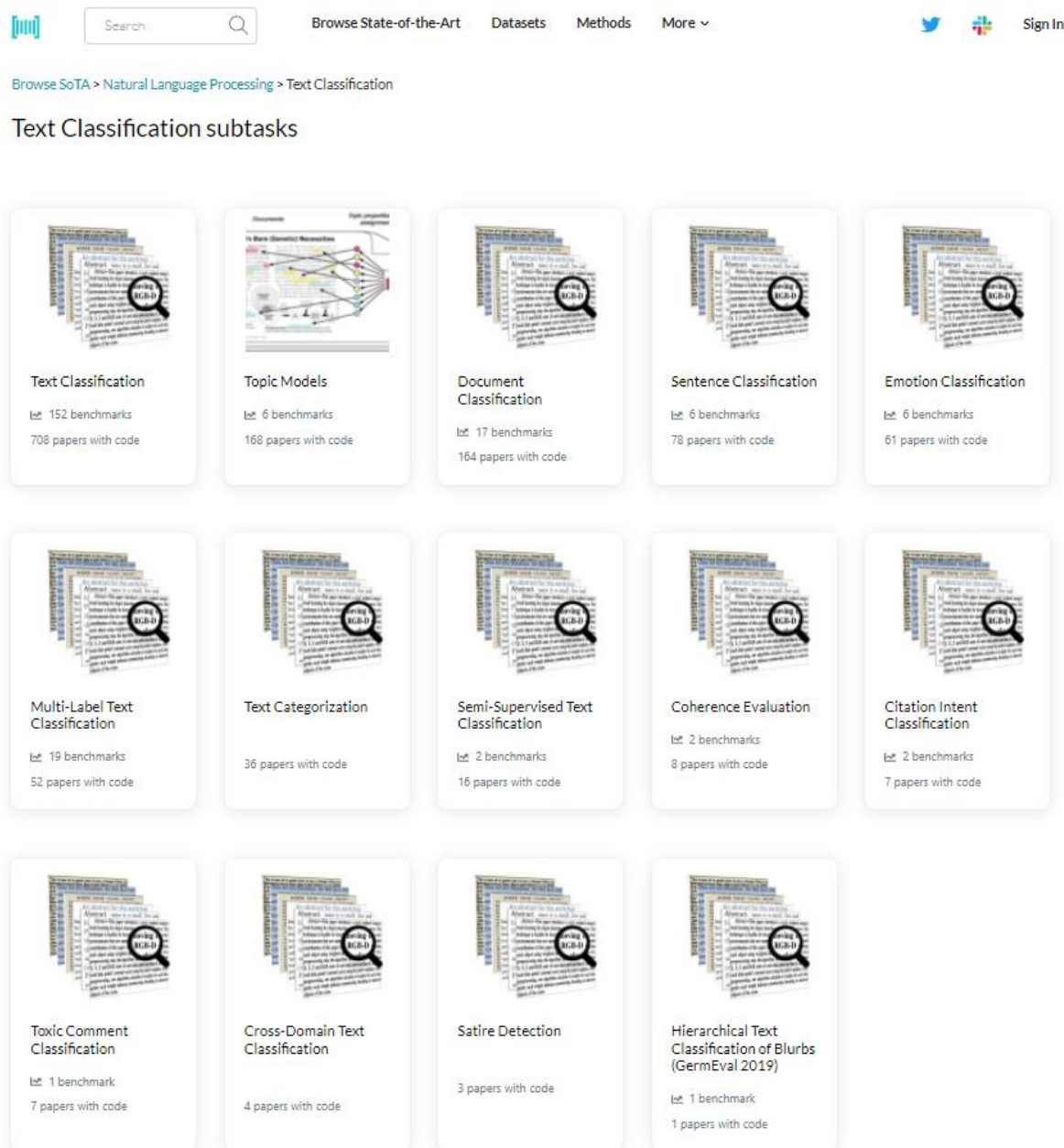


Рисунок 1 - Подзадачи «Классификации текстов»

Классификация документов — это процедура присвоения документу одной или нескольких меток из заранее определенного набора меток. [2]. Специфика данной

подзадачи заключается в том, что зачастую под «документами» подразумеваются текст, обладающие следующими характеристиками:

- объемные текстовые данные (большая длина текста, что ограничивает круг возможных применяемых методов для решения задачи классификации);
- внутренняя структура (заголовки, подзаголовки);
- специфичная лексика, соответствующая определённой предметной области.

Все эти факты позволяют выделить из всей области классификации текстов отдельную подзадачу классификации документов.

Рисунок 2 демонстрирует страницу, посвященную данной задаче на ресурсе Papers With Code. После формального описания задачи следует таблица метрик сравнения качества работы методов, предназначенных для решения поставленной задачи (Benchmarks).

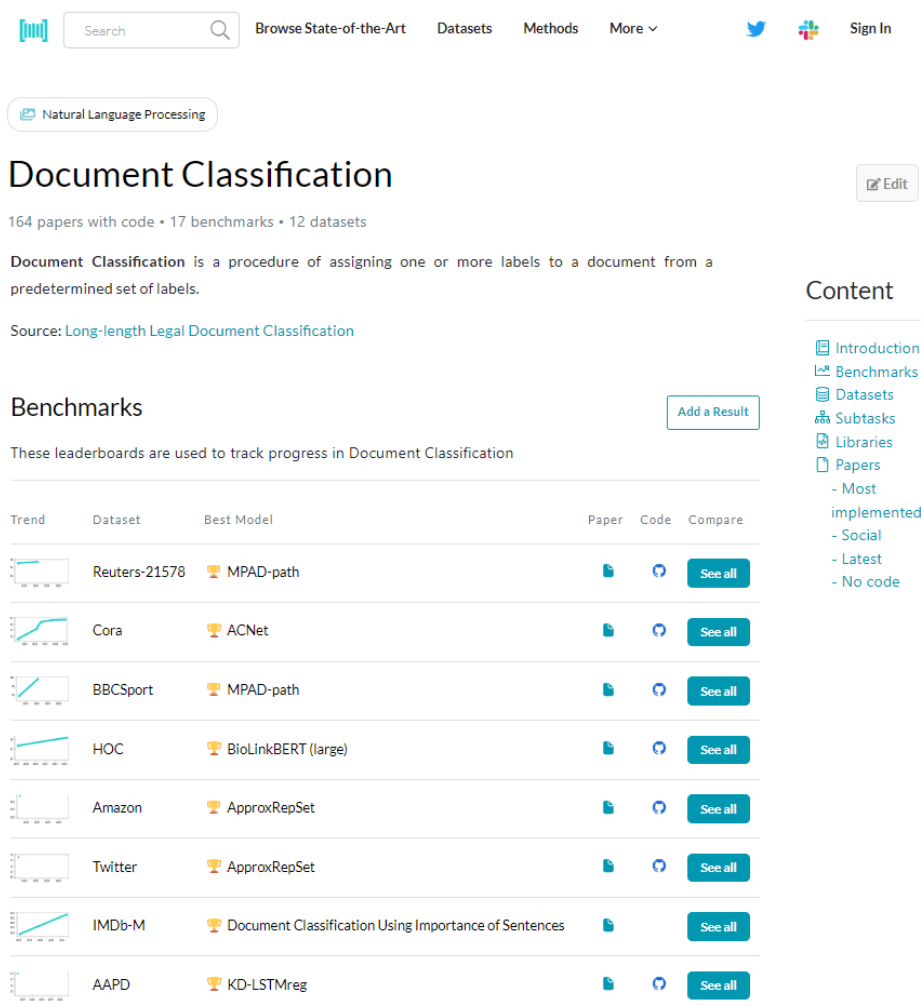


Рисунок 2 - Страница, посвященная классификации документов

Таблица состоит из колонок:

- Trend — тенденция качества решения задачи классификации на соответствующем датасете;

- Dataset – набор данных, на котором производится оценка качества работы алгоритма классификации;
- Best Model – метод (или модель машинного обучения), показавший на данный момент лучшие результаты в решении задачи классификации на соответствующем датасете;
- Paper – индикатор наличия статьи, описывающей соответствующий метод;
- Code – индикатор наличия кода, реализующего соответствующий метод (модель);
- Compare – ссылка для перехода на новую страницу со сравнением всех моделей и датасетов.

Далее ниже на той же странице (см. Рисунок 3) можно найти следующие разделы:



- раздел, посвященный библиотекам (Libraries), содержащим проверенные временем модели, предназначенные для решения задачи классификации документов;
- раздел Datasets, содержащий все датасеты, которые можно использовать для решения поставленной задачи;
- раздел Subtasks, описывающий возможные подзадачи, выделяемые в данной задаче.
- раздел Most Implemented Papers, содержащий статьи, отсортированные по применимости сопутствующим им кода и/или теоретической информации.

Последний раздел также можно отсортировать по недавним статьям, выпущенным по данной теме.

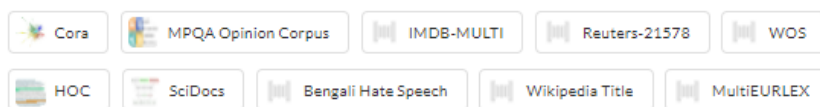
Следует также отметить, что данный раздел содержит не только статьи, описывающие методы, применяемые исключительно для классификации документов, но и для классификации текстов, и для других задач. Подробнее это можно посмотреть на странице, посвященной конкретной статье. Например, наиболее применяемая статья называется «Graph Attention Networks», и описывает общий подход к созданию и описанию GAN-сетей, широко применяемых в том числе и в других задачах.

Libraries ⓘ

Use these libraries to find Document Classification models and implementations

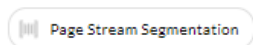
 sergioburdisso/pyss3	2 papers	254 ★
 eske/multivec	2 papers	115 ★

Datasets



See all 12 document classification datasets


Subtasks





Most implemented papers

Most implemented Social Latest No code


Search for a paper, author or keyword





Graph Attention Networks


 PetarV-/GAT •  tensorflow • ICLR 2018

We present graph attention networks (GATs), novel neural network architectures that operate on graph-structured data, leveraging masked self-attentional layers to address the shortcomings of prior methods based on graph convolutions or their approximations.

 Paper

 Code

 73



Semi-Supervised Classification with Graph Convolutional Networks


 27


Рисунок 3 - Продолжение страницы, посвященной классификации документов

В рамках данного домашнего задания было интересно найти статью, описывающую решение некой специфичной для данной области задачи. В итоге выбор остановился на статье «Aspect-based Document Similarity for Research Papers» [3], которая предполагает определение схожести не просто научных статей в целом, но их отдельных разделов (введения, выводов, проведенных экспериментов и т.д.).

Рисунок 4 демонстрирует страницу, посвященную данной статье. На данной странице можно:

- прочитать Аннотацию (Abstract) к статье;
- получить текст статьи в формате PDF;

- открыть код, реализующий методы, описанные в данной статье;
- посмотреть задачи, решение которых предлагают авторы статьи;
- посмотреть наборы данных, на которых обучалась или тестировалась модель;
- посмотреть результаты решения задачи (необязательно указываются на данной сайте авторами статей).



[Browse State-of-the-Art](#)
[Datasets](#)
[Methods](#)
[More ▾](#)

[Twitter](#)
[Google+](#)
[Sign In](#)

Aspect-based Document Similarity for Research Papers

COLING 2020 · Malte Ostendorff, Terry Ruas, Till Blume, Bela Gipp, Georg Rehm · [Edit social preview](#)

Traditional document similarity measures provide a coarse-grained distinction between similar and dissimilar documents. Typically, they do not consider in what aspects two documents are similar. This limits the granularity of applications like recommender systems that rely on document similarity. In this paper, we extend similarity with aspect information by performing a pairwise document classification task. We evaluate our aspect-based document similarity for research papers. Paper citations indicate the aspect-based similarity, i.e., the section title in which a citation occurs acts as a label for the pair of citing and cited paper. We apply a series of Transformer models such as RoBERTa, ELECTRA, XLNet, and BERT variations and compare them to an LSTM baseline. We perform our experiments on two newly constructed datasets of 172,073 research paper pairs from the ACL Anthology and CORD-19 corpus. Our results show SciBERT as the best performing system. A qualitative examination validates our quantitative results. Our findings motivate future research of aspect-based document similarity and the development of a recommender system based on the evaluated techniques. We make our datasets, code, and trained models publicly available.

[PDF](#)
[Abstract](#)
[COLING 2020 PDF](#)
[COLING 2020 Abstract](#)

Code

[Edit](#)

malteos/aspect-document-similarity ★ 44 PyTorch

[Quickstart in Colab](#)

Tasks

[Edit](#)

[Document Classification](#)
[Recommendation Systems](#)

Datasets

[Edit](#)

[CORD-19](#)

Results from the Paper

[Edit](#)

Submit [results from this paper](#) to get state-of-the-art GitHub badges and help the community compare results to other papers.

Рисунок 4 - Страница, посвященная статье

3. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

В рамках домашнего задания рассматриваются материалы статьи «Сходство документов на основе аспектов для исследовательских работ» (Aspect-based Document Similarity for Research Papers).

Традиционные меры сходства документов грубо определяют различие между похожими и непохожими документами. Как правило, они не учитывают, в каких аспектах (разделах) два документа похожи. Это ограничивает степень детализации приложений, таких как рекомендательные системы, которые полагаются на сходство документов. В статье авторы предлагают дополнение методики определения сходства информацией об аспектах (разделах), выполняя задачу парной классификации документов. Сходство документов определяется на основе аспектов (разделов) для исследовательских работ. Цитаты из статей указывают на сходство на основе аспектов (разделов), т. е. заголовок раздела, в котором встречается цитата, действует как метка для пары «статья с цитатой» и «цитируемая статья». В рамках статьи применяются различные варианты моделей трансформеров, таких как RoBERTa, ELECTRA, XLNet и BERT, и сравниваются их с baseline-решением на LSTM. Эксперименты проводятся на двух недавно созданных наборах данных из 172073 пар исследовательских работ из антологии ACL и корпуса CORD-19. Результаты работы показывают, что SciBERT является самой эффективной моделью в рамках решаемой задачи. Качественное исследование результатов подтверждает количественные результаты. Полученные результаты мотивируют будущие исследования сходства документов на основе аспектов (разделов) и разработку рекомендательной системы на основе оцененных методов. Авторы выкладывают наборы данных, код и обученные модели в общем доступе.

3.1. Трансформеры

Архитектура модели «трансформер» основана на механизме внимания - чрезвычайно распространенном методе в современных моделях глубокого обучения, позволяющий улучшить показатели эффективности приложений нейронного машинного перевода. В данном разделе будет рассмотрена модель Трансформер, которая использует механизм внимания для повышения скорости обучения. Более того, для ряда задач Трансформеры превосходят модель нейронного машинного перевода от Google. Однако самое большое преимущество Трансформеров заключается в их высокой эффективности в условиях параллелизации (parallelization) [4].

3.1.1. Высокоуровневое представление

В высокоуровневом представлении трансформер состоит из кодирующего компонента, декодирующего компонента и связи между ними (см. Рисунок 5). Кодирующий компонент – это стек энкодеров; Рисунок 6 демонстрирует 6 энкодеров, расположенных друг над другом (можно экспериментировать и с любым другим числом кодировщиков). Декодирующий компонент – это стек декодеров, представленных в том же количестве.

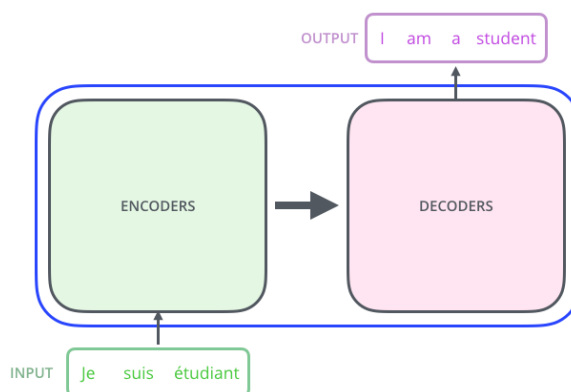


Рисунок 5 - Упрощенная архитектура Трансформера

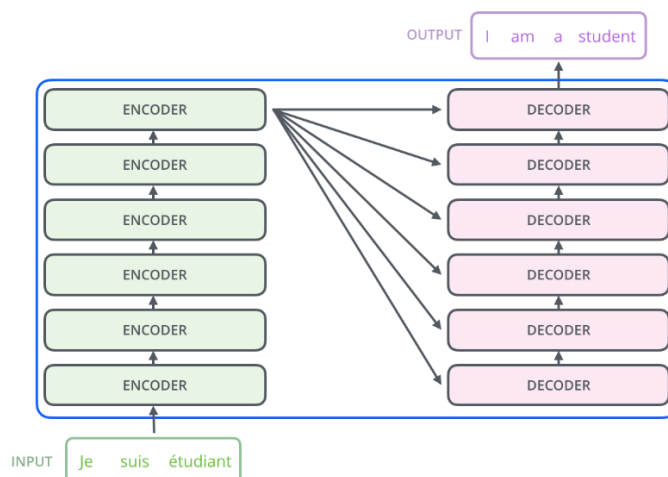


Рисунок 6 - Стек кодировщиков и декодировщиков

Все энкодеры идентичны по структуре, хотя и имеют разные веса. Каждый можно разделить на два подслоя, см. Рисунок. Входная последовательность, поступающая в энкодер, сначала проходит через слой внутреннего внимания (self-attention), помогающий энкодеру посмотреть на другие слова во входящем предложении во время кодирования конкретного слова. Мы рассмотрим этот механизм далее в статье.

Выход слоя внутреннего внимания отправляется в нейронную сеть прямого распространения (feed-forward neural network). Точно такая же сеть независимо применяется для каждого слова в предложении.

Декодер также содержит два этих слоя, но между ними есть слой внимания, который помогает декодеру фокусироваться на релевантных частях входящего предложения (это схоже с тем, как механизм внимания организован в моделях seq2seq).

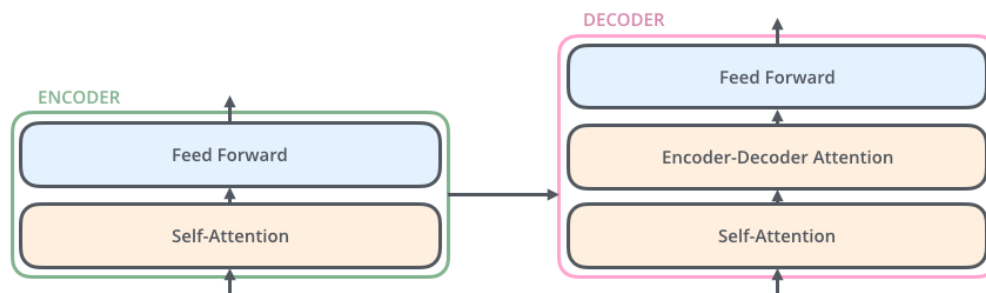


Рисунок 7 - Внутреннее устройство энкодеров и декодеров

3.1.2. Преобразование входной последовательности

Посмотрим на различные векторы/тензоры, и как они передаются от компонента к компоненту, преобразуя входную последовательность обученной модели в выходную.

Как и в случае любого NLP-приложения, мы начинаем с того, что преобразуем слово в вектор, используя алгоритм эмбедингов слов (word embeddings). Каждое слово преобразовывается в вектор размерностью 512. На рисунках вектора будут изображены в виде последовательности квадратов.

Эмбединги применяются только в самом нижнем энкодере. На уровне абстракции, общей для всех энкодеров, происходит следующее: энкодеры получают набор векторов размерностью 512 (для самого нижнего энкодера это будут эмбединги слов, для других – выходные вектора нижестоящих энкодеров). Размер этого набора векторов является гиперпараметром, который мы можем устанавливать, и, по сути, равен длине самого длинного предложения в обучающем корпусе.

После того как слова входящего предложения преобразовались в эмбединги, каждый из них в отдельности проходит через два слоя энкодера.

Одна из основных особенностей Трансформера: каждое слово идет по своей собственной траектории в энкодере. И, хотя существуют зависимости между этими траекториями в слое внутреннего внимания, в слое сети прямого распространения таких зависимостей нет, что позволяет различным траекториям выполняться параллельно во время прохождения через этот слой.

Энкодер получает на вход и обрабатывает набор векторов, проводя их через слой внутреннего внимания и далее – через нейронную сеть прямого распространения, пока, наконец, не передает свой выход следующему энкодеру (см. Рисунок 8).

Слова в каждой из позиций проходят через слой внутреннего внимания. Далее каждое из них переходит в отдельную, но абсолютно идентичную нейронную сеть прямого распространения.

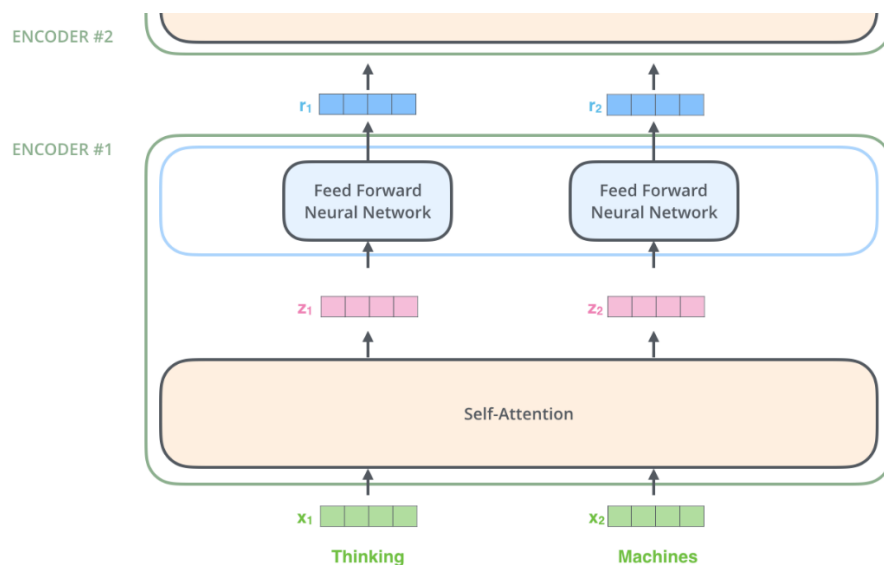


Рисунок 8 - Работа со словами в энкодере

3.1.3. Механизм внутреннего внимания

Механизм внутреннего внимания является современным методом моделирования «понимания» других релевантных слов при обработке конкретного слова. Прежде всего рассмотрим суть данного механизма высокоуровнево.

Пусть следующее предложение – это входящее предложение, которое мы хотим перевести: «The animal didn't cross the street because it was too tired». К чему относится «it» в этом предложении? К улице (street) или к животному (animal)? Простой вопрос для человека становится целой проблемой для алгоритма.

Когда модель обрабатывает слово «it», слой внутреннего внимания помогает понять, что «it» относится к «animal».

По мере того, как модель обрабатывает каждое слово (каждую позицию во входной последовательности), внутреннее внимание позволяет модели взглянуть на другие позиции входной последовательности и найти подсказку, помогающую лучше закодировать данное слово.

В рекуррентных нейронных сетях сохранение скрытого состояния позволяет включать представление предыдущих слов/векторов, которые уже были обработаны, в текущее обрабатываемое слово. Механизм внутреннего внимания – это метод, который Трансформер использует для того, чтобы смоделировать «понимание» других релевантных слов при обработке конкретного слова.

Рассмотрим механизм внимания немного подробнее. Первый этап в вычислении внутреннего внимания – это создать три вектора из каждого входящего вектора (в нашем случае – эмбединга каждого слова): вектор запроса (Query vector), вектор ключа (Key vector) и вектор значения (Value vector). (см. Рисунок 9). Эти векторы создаются с помощью перемножения эмбединга на три матрицы, которые мы обучили во время процесса обучения.

Эти новые векторы меньше в размере, чем векторы эмбедингов. Их размерность составляет 64, в то время как эмбединги и входящие/выходные векторы энкодера имеют размерность 512. Они не обязаны быть меньше, но в данной случае выбор данной архитектуры модели обусловлен желанием сделать вычисления в слое множественного внимания (multi-head attention) более стабильными.

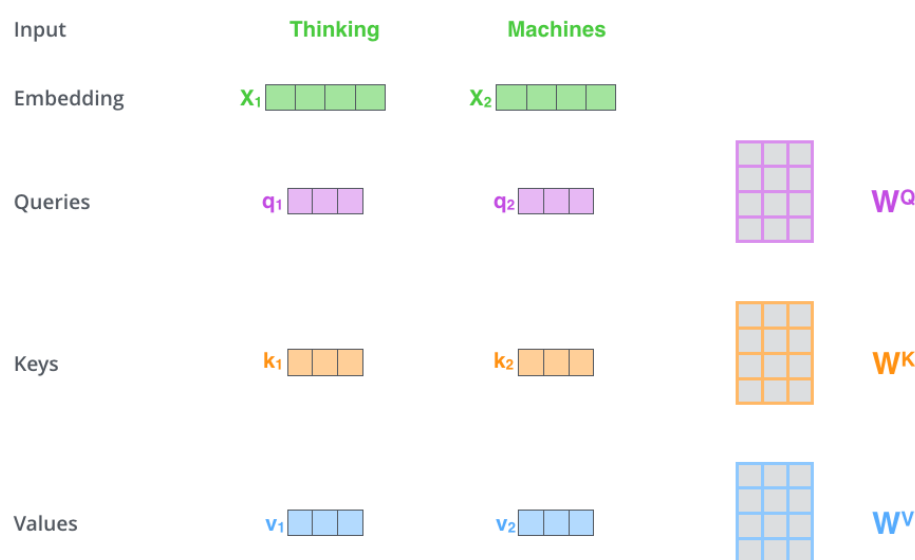


Рисунок 9 - Векторы механизма внимания

Умножение x_1 на матрицу весов W^Q производит q_1 , вектор «запроса», относящийся к этому слову. В итоге мы создаем проекции «запроса», «ключа» и «значения» для каждого слова во входящем предложении.

Второй этап вычисления внутреннего внимания – получение коэффициента (score). Допустим, мы подсчитываем внутреннее внимание для первого слова в нашем примере – «Thinking». Нам нужно оценить каждое слово во входящем предложении по отношению к данному слову. Коэффициент определяет, насколько нужно сфокусироваться на других частях входящего предложения во время кодирования слова в конкретной позиции.

Коэффициент подсчитывается с помощью скалярного произведения вектора запроса и вектора ключа соответствующего слова. Таким образом, если мы вычисляем

внутреннее внимание для слова в позиции #1, первый коэффициент будет скалярным произведением q_1 и k_1 , второй — скалярным произведением q_1 и k_2 .

Третий и четвертый этапы – разделить эти коэффициенты на 8 (квадратный корень размерности векторов ключа, используемой в статье – 64; данное значение обеспечивает более стабильные градиенты и используется по умолчанию, но возможны также и другие значения), а затем пропустить результат через функцию софтмакс (softmax). Данная функция нормализует коэффициенты так, чтобы они были положительными и в сумме давали 1.

Полученный софтмакс-коэффициент (softmax score) определяет, в какой мере каждое из слов предложения будет выражено в определенной позиции. Очевидно, что слово в своей позиции получит наибольший софтмакс-коэффициент, но иногда полезно учитывать и другое слово, релевантное к рассматриваемому.

Пятый этап – умножить каждый вектор значения на софтмакс-коэффициент (перед их сложением). Интуиция здесь следующая: нужно держать без изменений значения слов, на которых мы фокусируемся, и отвести на второй план нерелевантные слова (умножив их на небольшие значения, например, 0.001).

Шестой этап – сложить взвешенные векторы значения. Это и будет представлять собой выход слоя внутреннего внимания в данной позиции (для первого слова).

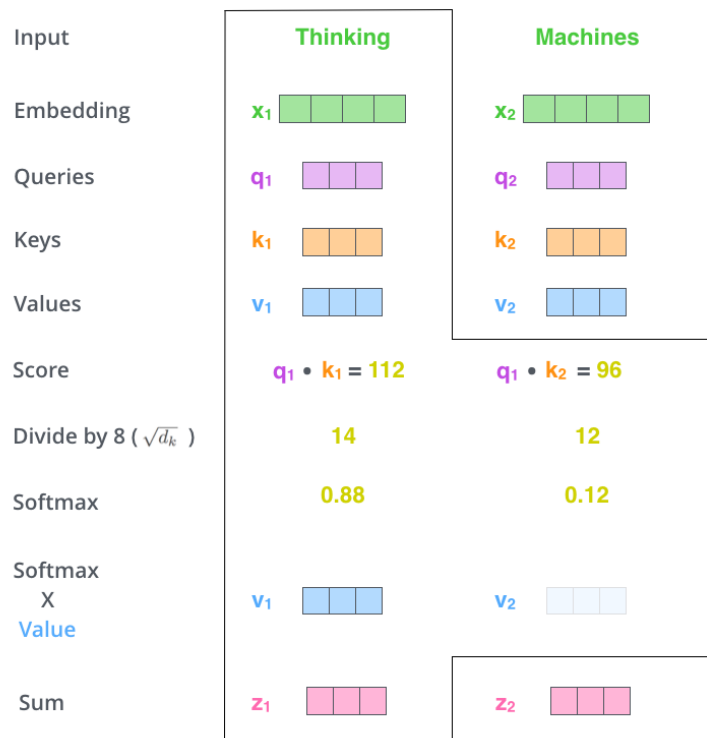


Рисунок 10 - Схема вычисления внутреннего внимания

На этом завершается вычисление внутреннего внимания. В результате мы получаем вектор, который можем передавать дальше в нейронную сеть прямого

распространения. В настоящих реализациях, однако, эти вычисления делаются в матричной форме для более быстрой обработки.

3.1.4. Множественное внимание

Внутреннее внимание совершенствуется с помощью добавления механизма, называющегося множественным вниманием (multi-head attention). Данная техника улучшает производительность слоя внутреннего внимания за счет следующих аспектов:

Повышается способность модели фокусироваться на разных позициях. Да, в примере выше, z_1 содержит немного от всех других кодировок, но он не может доминировать над самим словом. В случае с переводом предложения вроде «The animal didn't cross the street because it was too tired», мы хотим знать, к какому слову относится «it».

Слой внимания снабжается множеством «подпространств представлений» (representation subspaces). Как мы увидим далее, с помощью множественного внимания у нас есть не один, а множество наборов матриц запроса/ключа/значения (Трансформер использует 8 «голов» внимания, так что в итоге у нас получается 8 наборов для каждого энкодера/декодера). Каждый из этих наборов создается случайным образом. Далее после обучения каждый набор используется для отображения входящих эмбеддингов (или векторов с нижестоящих энкодеров/декодеров) в разных подпространствах представлений.

В случае множественного внимания, мы располагаем отдельными $WQ/WK/WV$ матрицами весов для каждой «головы» (см. Рисунок), что в результате дает разные $Q/K/V$ матрицы. Как мы делали ранее, умножаем X на $WQ/WK/WV$ матрицы для получения $Q/K/V$ матриц.

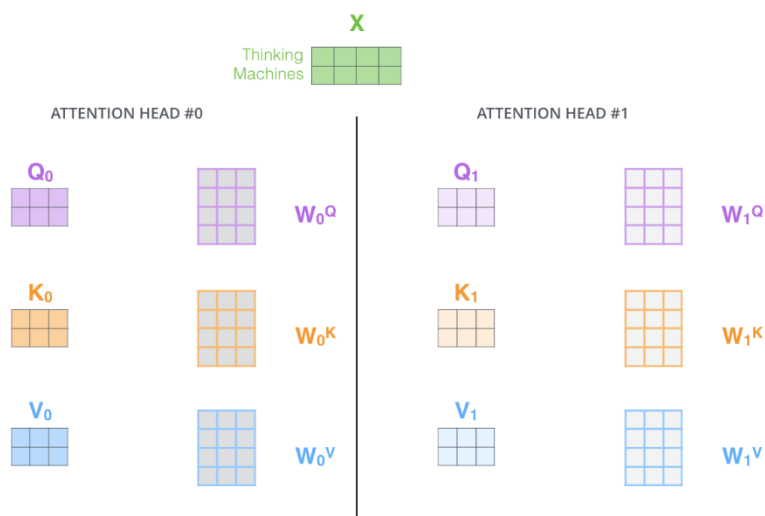


Рисунок 11 - Множественное внимание

После завершения фазы кодирования начинается фаза декодирования. Каждый этап фазы декодирования возвращает элемент выходной последовательности (в данном случае – переводное предложение на английском).

3.1.5. Полный алгоритм обработки

Рассмотрим весь процесс обработки предложения целиком (см. Рисунок). Энкодер начинает обрабатывать входящее предложение. Выход верхнего энкодера затем преобразуется в набор векторов внимания K и V . Они используются всеми декодерами в их «энкодер-декодер» слое внимания, что помогает им фокусироваться на подходящих местах во входящем предложении.

Следующие шаги повторяются до появления специального символа, сообщающего, что декодер Трансформера завершил генерацию выходной последовательности. Выход каждого этапа отправляется на нижний декодер в следующем временном промежутке, и декодеры генерируют свой результат так же, как это делают энкодеры. И точно так же, как мы делали с входами энкодеров, мы добавляем позиционное кодирование на те входы декодеров, которые указывают на позицию каждого слова.

В декодере слой внутреннего внимания может фокусироваться только на предыдущих позициях в выходном предложении. Это делается с помощью маскировки всех позиций после текущей (устанавливая их в $-\infty$) перед этапом софтмакс в вычислении внутреннего внимания.

Стек декодеров на выходе возвращает вектор чисел с плавающей точкой. Как можно получить из этого вектора слово? За это отвечает линейный слой и следующий за ним слой софтмакс.

Линейный слой – это простая полносвязная нейронная сеть, которая переводит вектор, созданный стеком декодеров, в значительно больший вектор, называемый логит вектором (logits vector).

Слой софтмакс переводит этот показатель в вероятности (положительные числа, сумма которых равна 1). Выбирается ячейка с наиболее высокой вероятностью и на выход данного временного отрезка подается соответствующее слово.

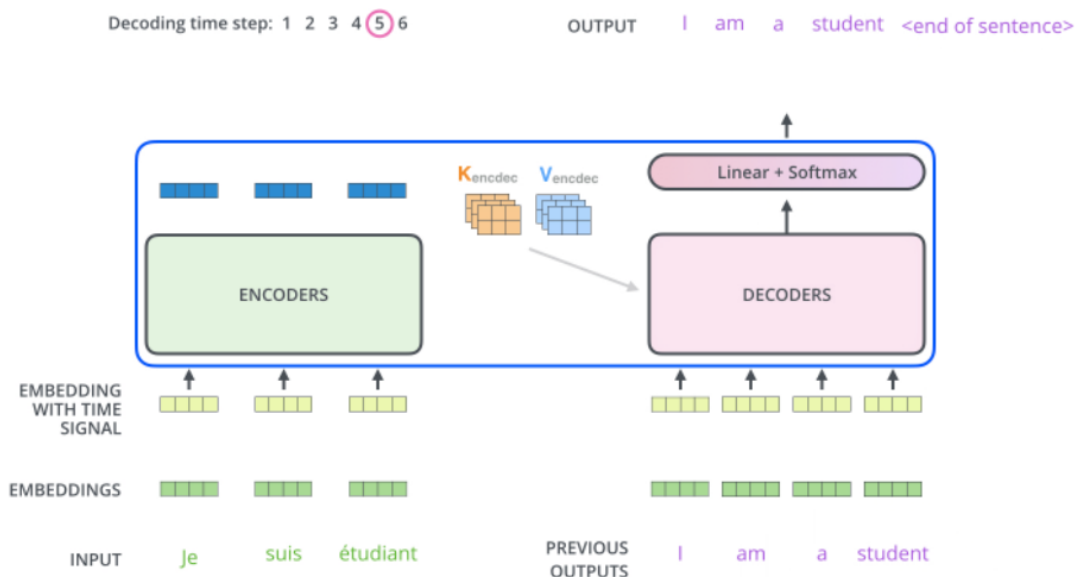


Рисунок 12 - Порядок обработки предложения

3.2. Сходство документов на основе аспектов

Рекомендательные системы (РС) помогают исследователям в поиске соответствующих документов для их работы. Когда отзывы пользователей скудны или недоступны, используются подходы, основанные на контенте, и соответствующие меры сходства документов. РС рекомендуют документ-кандидат в зависимости от того, похож ли он или отличается от начального документа. Эта грубая оценка сходства (похоже или нет) пренебрегает многими аспектами, которые могут сделать два документа похожими. Что касается общей концепции сходства, некоторые даже утверждают, что сходство является плохо определенным понятием, если только нельзя сказать, к каким аспектам относится сходство. В РС для научных работ сходство часто связано с несколькими аспектами (разделами) представленного исследования, например, методом, результатами. Учитывая, что сходство документов позволяет дифференцировать аспекты исследования, можно получить конкретные индивидуальные рекомендации. Например, можно было бы рекомендовать документ с аналогичными методами, но другими выводами. Такая РС облегчила бы открытие аналогий в исследовательской литературе. В статье описывается лежащее в основе многоаспектное сходство в исследовательских работах как сходство документов на основе аспектов. Рисунок 13 иллюстрирует аспектное сходство в отличие от безаспектного сходства (aspect-free). Следуя примеру исследовательской работы, аспект a_1 касается результатов, а аспект a_2 - методов (красный и зеленый, Рисунок 13(b)).

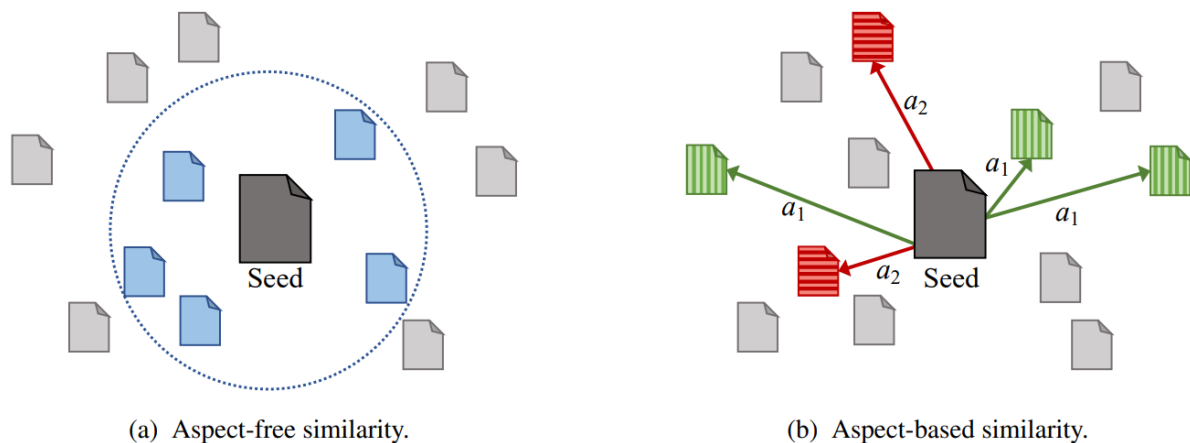


Рисунок 13 - Сравнение мер сходства

3.2.1. Описание методологии

Для обучения классификатора для решения задачи аспектного сходства используются заголовки разделов из цитат в качестве меток для пар документов (см. Рисунок 14). Разделы определяют аспекты сходства. Для классификации используется модель Transformer с заголовками (title) и аннотациями (abstract) в качестве входных данных.

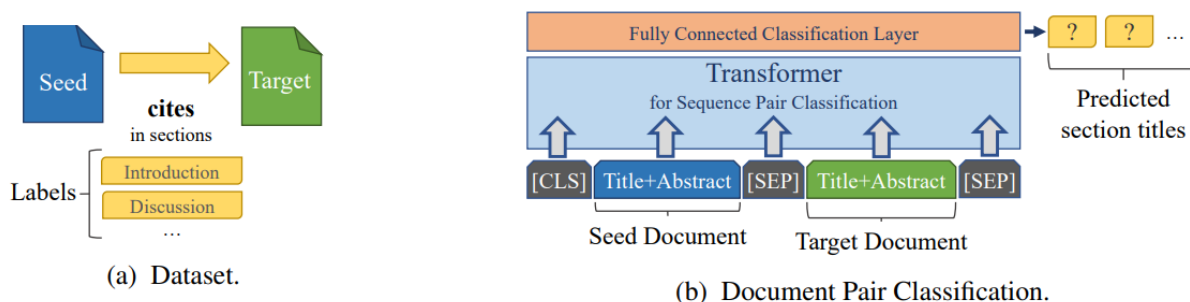


Рисунок 14 - Используемая методология

Генерация аннотированных человеком данных для рекомендаций по исследовательским работам требует больших затрат и обычно ограничивается небольшими количествами. Небольшой размер набора данных затрудняет применение алгоритмов обучения. Чтобы смягчить проблему нехватки данных, исследователи полагаются на цитаты как на основную истину, т. е. когда цитата существует между двумя статьями, эти две статьи считаются похожими. Существует цитата или не соответствует метке для бинарной классификации. Чтобы сделать сходство аспектным, мы переносим эту идею на многометочной многоклассовой классификации. В качестве основной истины мы принимаем название раздела, в котором цитирование из исходной статьи (seed) в целевую (target), как класс метки (Рисунок 14 (a)). Классификация является многоклассовой из-за нескольких заголовков разделов, и с несколькими

метками, потому что статья А может цитировать В в нескольких разделах. Например, статья, цитируемая в разделе «Введение и обсуждение», будет соответствовать одному экземпляру набора данных.

Выполняется классификация пар последовательностей с помощью моделей, основанных на архитектуре Transformer. Модели на основе трансформеров часто используются в задачах определения сходства текстов. XLNet, превосходит сиамские сети и традиционные вложения слов (например, GloVe), Paragraph Vectors в задаче попарной классификации документов. Следовательно, мы исключаем сиамские сети и предварительно обученные модели встраивания слов в наши эксперименты. Вместо этого мы исследуем шесть вариантов трансформеров и дополнительную baseline-модель для сравнения. Заголовки и тезисы (аннотации – abstract) пар исследовательских работ используются в качестве входных данных для модели, при этом токен [SEP] разделяет исходную и целевую статью (Рисунок 14 (b)). Авторы не используют полные тексты в экспериментах, так как многих документов нет в свободном доступе, и для выбранных моделей Transformer установлено жесткое ограничение в 512 токенов.

3.2.2. Наборы данных и их предобработка

В работе используется два набора данных:

ACL Anthology. Используется Справочный корпус антологии ACL в качестве набора данных. Корпус включает 22 878 научных статей по компьютерной лингвистике. Помимо полных текстов, набор данных ACL Anthology предоставляет дополнительные данные о цитировании. Цитаты снабжены аннотациями с названием раздела, в котором расположены маркеры цитирования. Эта информация необходима для дальнейших экспериментов.

CORD-19. Набор данных открытых исследований COVID-19 (COVID-19 Open Research Dataset - CORD-19) представляет собой сборник статей о COVID-19 и связанных с ним исследованиях коронавируса из нескольких биомедицинских цифровых библиотек. Цитаты и метаданные всех документов CORD-19 стандартизированы в соответствии с принятыми правилами обработки. Цитаты в CORD-19 также аннотируются названиями разделов.

Таким образом, получены два набора данных для попарной классификации документов с несколькими метками и несколькими классами. Заголовки разделов цитат, т. е. классы меток, представлены в таблице 1. Заголовки разделов нормализованы (строчные, только буквы, единственное число во множественное число) и объединенные разделы разделены на несколько («Заключение и Будущая работа» до «Заключение; Будущая Работа»). Недействительные статьи без текста или дубликаты также удаляются.

Мы разделяем оба набора данных, ACL Anthology и CORD-19, на десять классов в соответствии с количеством экземпляров, при этом первые девять составляют наиболее популярные заголовки разделов, а десятый («Другое») группирует оставшиеся, пренебрегая вариациями названий разделов в литературе, наша модель по-прежнему удваивает количество определенных аспектов в предыдущих исследованиях. Полученное в результате распределение по классам несбалансировано, но оно отражает истинную природу корпусов, как показывает Рисунок 15. Сценарии для воспроизведения наборов данных доступны с исходным кодом.

Label class	Count	Label class	Count	Label class	Count	Label class	Count
Introduction	16,279	Conclusion	1,158	Introduction	15,108	Background	454
Related Work	12,600	Discussion	1,132	Discussion	13,258	Materials	420
Experiment	4,025	Evaluation	971	Conclusion	1,003	Virus	218
Background	1,365	Methods	719	Results	910	Future work	171
Results	1,181	Other	22,249	Methods	523	Other	43,154

(a) ACL Anthology

(b) CORD-19

Рисунок 15 - Описание наборов данных

В дополнение к десяти положительным классам вводится класс «Нет» (None), который работает как отрицательный аналог для наших положительных образцов в той же пропорции. Пары документов «Нет» выбираются случайным образом и являются несходными. Случайная пара статей является отрицательным экземпляром, когда статьи не существуют как положительная пара, не цитируются вместе, не имеют общих авторов и не опубликованы в одном и том же месте. Создано 24275 отрицательных образцов для ACL Anthology и 33083 для CORD-19. Эти экземпляры позволяют моделям различать похожие и непохожие документы.

3.2.3. Рассматриваемые модели

Baseline-модель LSTM. В качестве baseline используется двунаправленный LSTM. Чтобы получить представления для пар документов, мы передаем название и аннотацию двух документов через LSTM, в результате чего документы разделяются специальным токеном-разделителем. Используется токенизатор SpaCy и векторы слов из fastText. Векторы слов предварительно обучены на рефератах ACL Anthology или наборах данных CORD-19.

BERT, Covid-BERT и SciBERT. BERT — это нейронная языковая модель, основанная на архитектуре Transformer. Обычно модели BERT предварительно обучаются на больших текстовых корпусах без учителя. Двумя целями предварительного обучения являются восстановление замаскированных токенов и предсказание следующего предложения (NSP). После предварительного обучения

модели BERT дополнительно настроены для конкретных задач, таких как сходство предложений или классификация документов. В открытом доступе несколько моделей BERT, предварительно обученных на разных корпусах. В экспериментах оценивается три варианта BERT: (1) Модель BERT, обученная на английской Википедии и BooksCorpus; (2) SciBERT - вариант BERT, адаптированный для научной литературы, информационных и биомедицинских исследовательских работ; (3) Covid-BERT — оригинальная модель BERT, но настроенная на корпусе CORD-19.

BioBERT — еще одна модель BERT, специализирующаяся на биомедицинской области. Тем не менее, она исключается из экспериментов, поскольку SciBERT превосходит ее в биомедицинских задачах.

Все три модели, т. е. BERT, SciBERT и Covid-BERT, схожи по своей структуре, за исключением корпуса, используемого во время обучения языковой модели.

RoBERTa. Это модель BERT, обученная на больших партиях (batch) данных, обученная длительное время и исключаяющая задачу NSP из своей цели. Кроме того, RoBERTa использует дополнительные корпуса для предварительной подготовки, а именно Common Crawl News и STORIES.

XLNet. В отличие от BERT, XLNet — это не автоэнкодер, а авторегрессивная языковая модель. XLNet не использует NSP. Модель предварительно обучена на Википедии, BooksCorpus и Common Crawl.

ELECTRA. Модель имеет в дополнение к моделированию языка масок цель предварительной подготовки по обнаружению замененных токенов во входной последовательности. Для этой цели используется генератор, который заменяет токены, и сеть дискриминатора, которая обнаруживает замены. Генератор и дискриминатор являются моделями Transformer. ELECTRA не преследует цель NSP. Для экспериментов используется модель дискриминатора ELECTRA. Предварительно обученная модель дискриминатора ELECTRA предварительно обучена на тех же данных, что и BERT.

3.2.4. Результаты

Учитывая все оценки баллы (см. Рисунок 16), SciBERT является лучшим методом с 0,326 макро-F1 и 0,678 микро-F1 по ACL Anthology, а также с 0,439 макро-F1 и 0,833 микро-F1 по CORD-19. Все модели Transformer превосходят baseline LSTM по всем параметрам, за исключением micro-precision на ACL Anthology. Разрыв между средними макро- и микро-результатами обусловлен несоответствием между классами меток (см. далее). BERT, SciBERT и Covid-BERT в среднем работают лучше для ACL Anthology и CORD-19 по сравнению с baseline моделью и другими моделями на основе Transformer. Для ACL Anthology методы оцениваются одинаково как для макро, так и для микро.

SciBERT показывает самые высокие оценки с большим отрывом, за ним следуют Covid-BERT, XLNet и BERT. Худшие показатели у RoBERTa (0,626 микро-F1) и ELECTRA (0,616 микро-F1). Что касается показателя macro average, методы имеют одинаковый рейтинг для CORD-19 и ACL Anthology, за исключением BERT, который превосходит XLNet. Только для микросреднего на CORD-19 результат отличается, т. е. ELECTRA и RoBERTa получают более высокие баллы F1, чем Covid-BERT и XLNet. Несмотря на то, что Covid-BERT точно настроен на CORD-19, его производительность дает 0,818 микро-F1.

Dataset	ACL Anthology						CORD-19					
	macro avg			micro avg			macro avg			micro avg		
	F1(std)	P	R	F1(std)	P	R	F1(std)	P	R	F1(std)	P	R
LSTM _{baseline}	.063 ± .001	.069	.058	.290 ± .004	.761	.179	.128 ± .001	.137	.121	.579 ± .005	.758	.469
BERT	.256 ± .002	.317	.238	.641 ± .002	.719	.578	.387 ± .011	.619	.357	.822 ± .002	.840	.806
Covid-BERT	.270 ± .006	.404	.253	.648 ± .005	.715	.592	.394 ± .010	.578	.364	.818 ± .001	.836	.802
SciBERT	.326 ± .005	.458	.303	.678 ± .002	.725	.637	.439 ± .010	.560	.401	.833 ± .003	.846	.820
RoBERTa	.250 ± .003	.285	.232	.626 ± .003	.703	.564	.332 ± .008	.473	.316	.820 ± .001	.840	.801
XLNet	.263 ± .011	.372	.250	.645 ± .011	.705	.595	.362 ± .025	.523	.345	.817 ± .002	.832	.804
ELECTRA	.245 ± .005	.287	.228	.616 ± .021	.693	.554	.280 ± .001	.306	.276	.820 ± .002	.840	.801

Рисунок 16 - Оценка качества работы моделей

Оба набора данных, ACL Anthology и CORD-19, делятся на 11 классов меток с положительными и отрицательными примерами. Каждый класс представляет отдельный раздел, в котором статья цитируется. В разделе указано, в каких аспектах две статьи похожи. Аспекты также могут быть неоднозначными, что затрудняет классификацию меток. Рисунок демонстрирует эффективность классификации по отношению к различным классам меток. Здесь представлены оценка F1, точность и полнота SciBERT для всех 11 меток. Кроме того, включены общие результаты для экземпляров с одной и несколькими метками (например, 2 метки). Остальные методы предыдущего рисунка имеют более низкие, но пропорциональные аналогичные оценки.

ACL Anthology					CORD-19				
Label	Samples	F1 (Std)	P	R	Label	Samples	F1 (Std)	P	R
Background	341	0.436 ± 0.045	0.651	0.329	Background	113	0.617 ± 0.042	0.655	0.588
Conclusion	289	0.000 ± 0.000	0.000	0.000	Conclusion	250	0.274 ± 0.039	0.563	0.182
Discussion	283	0.000 ± 0.000	0.000	0.000	Discussion	3314	0.636 ± 0.008	0.641	0.631
Evaluation	242	0.008 ± 0.007	0.396	0.004	Future work	42	0.032 ± 0.064	0.150	0.018
Experiment	1006	0.360 ± 0.008	0.491	0.284	Introduction	3777	0.644 ± 0.004	0.669	0.620
Introduction	4069	0.527 ± 0.005	0.576	0.486	Materials	105	0.241 ± 0.038	0.552	0.157
Methods	179	0.014 ± 0.028	0.208	0.007	Methods	130	0.205 ± 0.030	0.519	0.130
Related work	3150	0.638 ± 0.012	0.660	0.617	Results	227	0.322 ± 0.021	0.558	0.227
Results	295	0.015 ± 0.011	0.475	0.008	Virus	54	0.000 ± 0.000	0.000	0.000
Other	5562	0.645 ± 0.005	0.646	0.645	Other	10788	0.876 ± 0.002	0.872	0.879
None	6068	0.942 ± 0.002	0.934	0.951	None	8270	0.979 ± 0.001	0.980	0.977
1 label	15652	0.721 ± 0.002	0.717	0.726	1 label	22885	0.860 ± 0.003	0.844	0.876
2 labels	1968	0.540 ± 0.003	0.738	0.425	2 labels	1632	0.656 ± 0.004	0.849	0.535
≥ 3 labels	585	0.492 ± 0.015	0.857	0.345	≥ 3 labels	295	0.590 ± 0.010	0.925	0.433

Рисунок 17 - Качество работы SciBERT

«None» имеет самую высокую оценку F1 (0,942 для ACL Anthology, 0,980 для CORD-19) с большим отрывом. «Другой» показывает второй лучший показатель F1, который в сценарии классификации «похож»-«не-похож» можно интерпретировать как класс, противоположный метке «Нет». Остальные положительные метки имеют более низкие оценки, но и меньшее количество экземпляров. В CORD-19 существует 10 788 тестовых образцов «Другой» по сравнению с 3 777 образцами «Введение», что является наиболее распространенным названием раздела. Тем не менее, меньшее количество выборок не обязательно коррелирует с низкой точностью. В ACL Anthology метка «Связанная работа» (3150 экземпляров) дает более высокие баллы по сравнению с «Введением» (4069 экземпляров) с оценкой F1 0,638 и 0,527 соответственно. Метка «Background» в CORD-19 имеет оценку F1 0,617, несмотря на то, что в ней всего 113 образцов. Результаты показывают влияние классов меток на общую производительность. Шесть меток (Антология ACL — «Заключение», «Обсуждение», «Оценка» и «Методы»; CORD-19 — «Будущая работа» и «Вирус») имеют баллы F1 от нуля до 0,05. Несоответствие количества выборок и сложность раскрытия скрытой информации из аспектов способствуют снижению точности некоторых меток. Даже для экспертов в предметной области не так просто предсказать, где одна статья цитирует другую, например, во «введении» или в «эксперименте».

Нижние строки иллюстрируют эффект множественных меток. Показатели F1 уменьшаются в обоих наборах данных по мере увеличения количества меток. Это связано с уменьшением recall. Точность увеличивается с увеличением количества меток. При наличии двух или более меток SciBERT часто правильно предсказывает одну из меток, но не другие. Например, две метки «Обсуждение и введение» (D, I) имеют только 22% правильных тестовых образцов. Тем не менее, SciBERT правильно предсказывает для остальных образцов одну из двух меток, то есть либо *Обсуждение* (35%), либо *Введение* (31%). Мы видим сопоставимые результаты для других мульти-меток, таких как «Обсуждение», «Введение» и «Другое» (D, I, O).

Чтобы подтвердить наши количественные результаты, также качественно оценивается прогноз SciBERT по антологии ACL. Рисунок 18 демонстрирует для каждого примера показан прогноз SciBERT, цитирует ли исходная статья целевую статью и в каком разделе должно быть цитирование. Прогнозы вручную проверяются на их правильность.

	Seed Paper	Target Paper	Citation	Prediction
1	UKP: Computing Semantic Textual Similarity by Combining Multiple Content Similarity Measures (Bär et al., 2012)	SemEval-2012 Task 6: A Pilot on Semantic Textual Similarity (Agirre et al., 2012)	Other	Introduction×
2	Query segmentation based on eigenspace similarity (Zhang et al., 2009)	Unsupervised query segmentation using generative language models and wikipedia (Tan and Peng, 2008)	Introduction, Experiment	Introduction✓, Experiment✓
3	Transition-Based Parsing of the Chinese Treebank using a Global Discriminative Model (Zhang and Clark, 2009)	Enhancing Statistical Machine Translation with Character Alignment (Xi et al., 2012)	None	Experiment×
4	Experiments in evaluating interactive spoken language systems (Polifroni et al., 1992)	Evaluating information presentation strategies for spoken recommendations (Winterboer and Moore, 2007)	None	Introduction×, Other×
5	Similarity-based Word Sense Disambiguation (Karov and Edelman, 1998)	Targeted disambiguation of ad-hoc, homogeneous sets of named entities (Wang et al., 2012)	None	None✓
6	SciSumm: A Multi-Document Summarization System for Scientific Articles (Agarwal et al., 2011)	Improving question-answering with linking dialogues (Gandhe et al., 2006)	None	None✓

Рисунок 18 - Качественная оценка SciBERT

Первый пример является правильным предсказанием. Учитывая основную истину, аспект — «Другой» (цитирование происходит в разделе «Результаты тестовых данных»). Мы оцениваем «Введение» как потенциально верный прогноз, поскольку исходная статья была предложена в рамках конференции целевой статьи. Поэтому целевую статью можно было бы указать во введении. Все предсказания в примере 2 верны. По сравнению с другими примерами мы рассматриваем пример 2 как простой случай, поскольку обе статьи упоминают свою тему (т. е. сегментацию запросов) в заголовке и в первом предложении аннотации (подсказка для метки «Введение»). Оба реферата примера 2 также ссылаются на «взаимную информацию и оптимизацию ЕМ» в качестве своих методов. В примере 3 статьи не имеют общего цитирования. Следовательно, паре статей присваивается метка «Нет» (None) в соответствии с основной истиной, даже если они тематически связаны. Обе статьи относятся к китайскому машинному переводу. Тем не менее, мы не согласны с предсказанием «Эксперимент» нашей моделью, поскольку в двух статьях проводятся разные эксперименты, что делает «Эксперимент» неверным предсказанием. Предсказания примера 4 верны. Исходная статья публикуется раньше целевой и, следовательно, цитирования быть не может. Тем не менее, эти два документа охватывают связанную тему. Таким образом, можно было бы ожидать цитирования во вводной части, как и предсказывал SciBERT. Наша модель находит это семантическое сходство с учетом их скрытой информации по теме. Пример 5-6 представляет две пары, для которых «None» был правильно предсказан в соответствии с основной истиной. Статьи из примера 6 тематически не связаны, как уже следует из их названий. Однако статьи в примере 5

разделяют тему устранения «неоднозначности». Таким образом, мы согласны с предсказанием положительной метки.

Таким образом, качественная оценка не противоречит количественным выводам. SciBERT различает документы на более высоком уровне и классифицирует, какие аспекты делают их похожими. В дополнение к традиционному сходству документов предсказания на основе аспектов позволяют оценить, как две статьи соотносятся друг с другом на семантическом уровне. Например, сходство двух статей в аспектах введения или эксперимента является ценной информацией, особенно в обзорах литературы.

4. ПРАКТИЧЕСКАЯ ЧАСТЬ

Авторы статьи выложили в общий доступ все материалы, относящиеся к опубликованной статье, включая коды программ, наборы данных, обученные модели и результаты [5].

4.1. Обзор страницы в GitHub

Рисунок 19 демонстрирует страницу на сайте GitHub, посвященную данной статье.

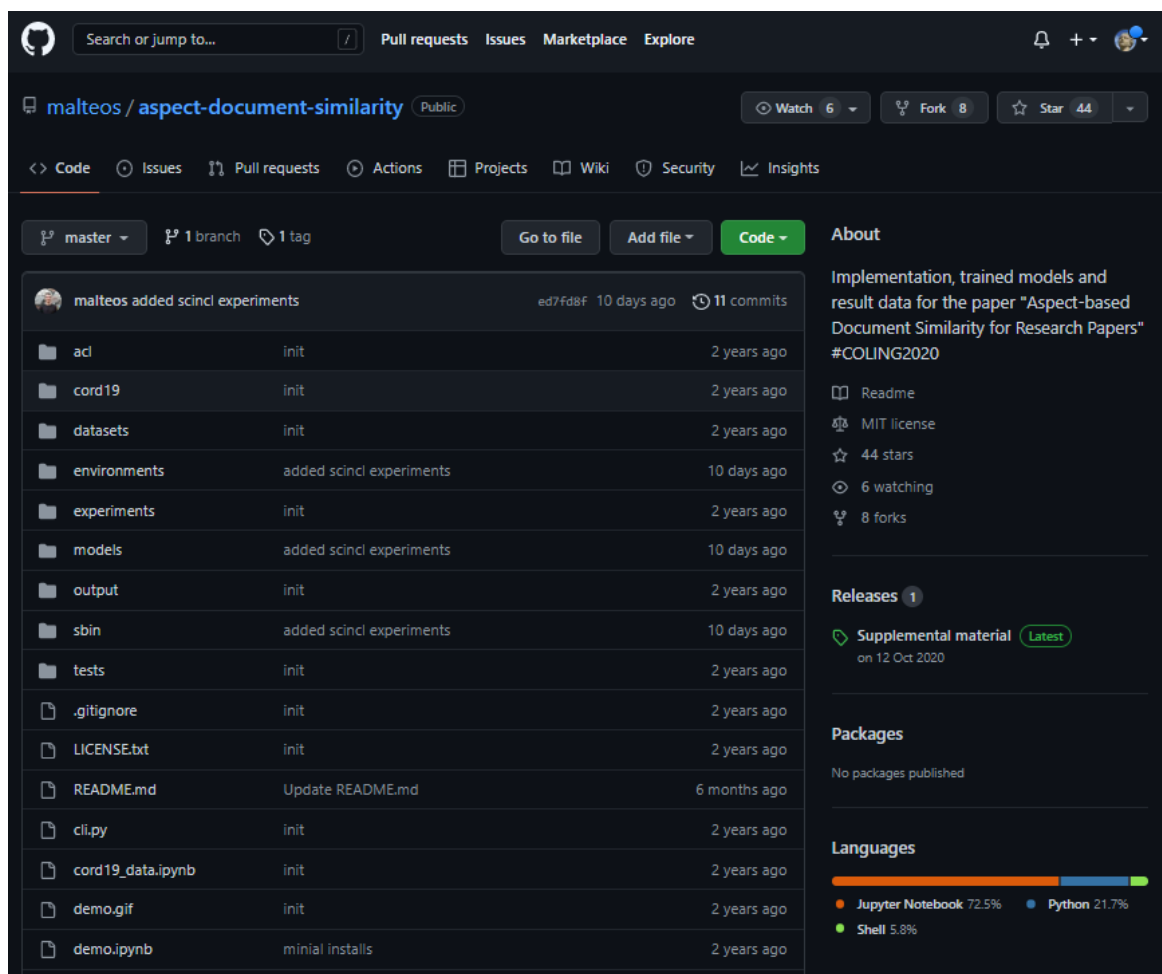


Рисунок 19 - Страница статьи в GitHub

На данной странице авторы дают краткое описание содержимого репозитория, а также описывают порядок работы с ним в нескольких разделах страницы.

4.1.1. Описание статьи

В первом разделе помимо описания авторы также указывают ссылки на сторонние источники, связанные со статьей. Текст статьи можно найти на сайте arxiv.org [6]. Дополнительные материалы по статье можно найти на отдельной странице, связанной с репозиторием [7]. Наборы данных, использованные в статье, можно найти на платформе

Hugging Face [8]. На той же платформе находится личная страница авторов статьи с принадлежащими им моделями [9].

4.1.2. Демо-пример

В следующем разделе авторы оставили ссылку на демонстрационный пример работы модели на платформе Google Colaboratory []. Также приводится анимация, объясняющая порядок работы с демонстрационным примером (см. Рисунок 20).

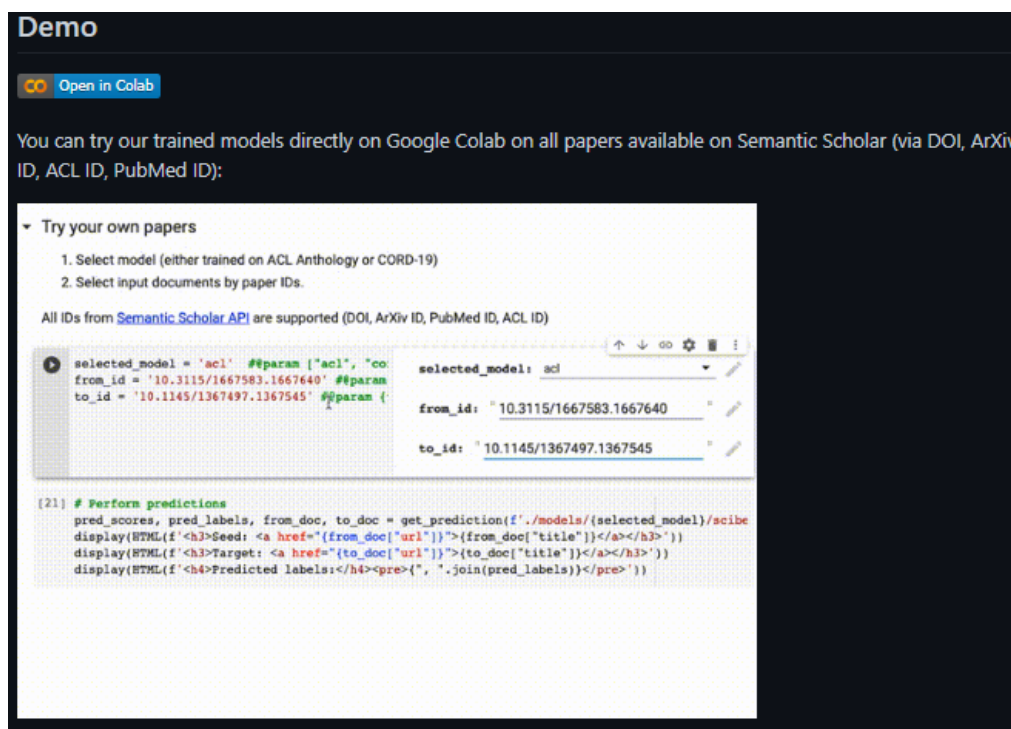


Рисунок 20 - Описание демонстрационного примера

4.1.3. Требования

В следующем разделе авторы перечисляют требования, необходимые для более углубленного тестирования приведенного программного кода. Требования следующие:

- Python версии 3.7;
- CUDA GPU для работы с Трансформерами;
- Наборы данных:
 - ACL Anthology Reference Corpus (ACL ARC);
 - COVID-19 Open Research Dataset (CORD 19).

4.1.4. Установка

Далее разработчики описывают порядок загрузки и установки репозитория локально для подробного изучения и тестирования работы.

Прежде всего требуется создать новое виртуальное пространство с помощью Conda:

```
conda create -n paper python=3.7
conda activate paper
```

Затем необходимо клонировать репозиторий и установить все требуемые библиотеки, указанные в requirements.txt:

```
git clone https://github.com/malteos/aspect-document-similarity.git repo
cd repo
pip install -r requirements.txt
```

4.1.5. Эксперименты

Далее следует раздел, посвященный самостоятельному проведению экспериментов с предоставленными материалами.

Для **подготовки наборов данных** необходимо выполнить следующие команды:

```
export DIR=./output

# ACL Anthology
# Get parscit files from: https://acl-arc.comp.nus.edu.sg/archives/acl-arc-160301-parscit/)
sh ./sbin/download_parscit.sh

# CORD-19
wget https://ai2-semantic scholar-cord-19.s3-us-west-2.amazonaws.com/historical_releases/cord-19_2020-03-13.tar.gz

# Get additional data (collected from Semantic Scholar API)
wget https://github.com/malteos/aspect-document-similarity/releases/download/1.0/acl_s2.tar
wget https://github.com/malteos/aspect-document-similarity/releases/download/1.0/cord19_s2.tar
```

Далее **наборы данных** необходимо правильным образом **сохранить** с указанием папок для входных и выходных данных:

```
# ACL
python -m acl.dataset save_dataset <input_dir> <parscit_dir> <output_dir>

# CORD-19
python -m cord19.dataset save_dataset <input_dir> <output_dir>
```

Для **использования набора данных** внутри собственного программного кода необходимо вставить следующий фрагмент кода:

```
from nlp import load_dataset

# Training data for first CV split
train_dataset = load_dataset(
    './datasets/cord19_docrel/cord19_docrel.py',
    name='relations',
    split='fold_1_train'
)
```

Для **использования моделей** внутри собственного программного кода необходимо вставить следующий фрагмент кода:

```
from models.auto_modelling import
AutoModelForMultiLabelSequenceClassification

# Load models with pretrained weights from Huggingface model hub
acl_model = AutoModelForMultiLabelSequenceClassification('malteos/aspect-
acl-scibert-scivocab-uncased')
cord19_model = AutoModelForMultiLabelSequenceClassification('malteos/aspect-
cord19-scibert-scivocab-uncased')

# Use the models in standard Huggingface fashion ...
# acl_model(input_ids, token_type_ids, ...)
# cord19_model(input_ids, token_type_ids, ...)
```

Для обучения моделей в командной строке необходимо выполнить код, хранящийся в файле `trainer_cli.py` со следующими параметрами:

```
python trainer_cli.py --cv_fold $CV_FOLD \
    --output_dir $OUTPUT_DIR \
    --model_name_or_path $MODEL_NAME \
    --doc_id_col $DOC_ID_COL \
    --doc_a_col $DOC_A_COL \
    --doc_b_col $DOC_B_COL \
    --nlp_dataset $NLP_DATASET \
    --nlp_cache_dir $NLP_CACHE_DIR \
    --cache_dir $CACHE_DIR \
    --num_train_epochs $EPOCHS \
    --seed $SEED \
    --per_gpu_eval_batch_size $EVAL_BATCH_SIZE \
```

```
--per_gpu_train_batch_size $TRAIN_BATCH_SIZE \  
--learning_rate $LR \  
--do_train \  
--save_predictions
```

Для оценки полученных результатов можно запустить утилиту Jupyter notebook с подготовленным заранее авторами файлом:

```
jupyter notebook evaluation.ipynb
```

4.2. Демонстрационный пример работы модели

Авторами статьи был предоставлен .ipynb-файл, в котором можно проверить работу обученных моделей на практике. В начале файла следуют служебные строки кода, необходимые для установки и импорта всех библиотек, предназначенных для работы с моделями и для вывода данных (см. Рисунок 21).

Demo for Aspect-oriented Similarity between Research Papers

Download models & install dependencies

```
[ ] !git clone https://github.com/malteos/aspect-document-similarity.git repo  
    %cd repo  
  
[ ] # Download models (scibert-scivocab-uncased, trained on first CV fold)  
    !mkdir -p models/acl models/cord19  
    !wget https://github.com/malteos/aspect-document-similarity/releases/download/1.0/acl_fold-1_scibert-scivocab-uncased.tar.gz  
    !tar -xzf acl_fold-1_scibert-scivocab-uncased.tar.gz  
    !mv scibert-scivocab-uncased models/acl  
  
    !wget https://github.com/malteos/aspect-document-similarity/releases/download/1.0/cord19_fold-1_scibert-scivocab-uncased.tar.gz  
    !tar -xzf cord19_fold-1_scibert-scivocab-uncased.tar.gz  
    !mv scibert-scivocab-uncased models/cord19  
  
    !wget -O models/cord19/scibert-scivocab-uncased/vocab.txt https://github.com/malteos/aspect-document-similarity/releases/download/1.0/cord19_fold-1_scibert-scivocab-uncased.tar.gz  
    !cp models/cord19/scibert-scivocab-uncased/vocab.txt models/acl/scibert-scivocab-uncased/vocab.txt  
  
[ ] # Install dependencies (for colab)  
    !pip install requests transformers==2.10.0  
  
    # Install all dependencies  
    !pip install -r requirements.txt  
  
[ ] from IPython.core.display import display, HTML  
    from demo_utils import get_prediction
```

Рисунок 21 - Служебные фрагменты кода

Далее следует фрагмент кода, с которым можно взаимодействовать для проверки работы моделей (см. Рисунок 22). В поле `selected_model` указывается модель, которую требуется проверить. В поле `from_id` вставляется идентификатор исходной статьи в форматах, поддерживаемых Semantic Scholar API (DOI, ArXiv ID, PubMed ID, ACL ID). В поле `to_id` также вписывается идентификатор статьи, но теперь уже целевой, с которой требуется сравнить исходную статью и определить в каком разделе она может быть указана в исходной статье, т.е. какой раздел у статей может совпадать.

Try your own papers

1. Select model (either trained on ACL Anthology or CORD-19)
2. Select input documents by paper IDs.

All IDs from [Semantic Scholar API](#) are supported (DOI, ArXiv ID, PubMed ID, ACL ID)

```
[ ] selected_model = 'acl' #@param ["acl", "cord19"]
    from_id = '10.3115/1667583.1667640' #@param {type:"string"}
    to_id = '10.1145/1367497.1367545' #@param {type:"string"}

selected_model: acl
from_id: "10.3115/1667583.1667640"
to_id: "10.1145/1367497.1367545"

[ ] # Perform predictions
    pred_scores, pred_labels, from_doc, to_doc = get_prediction(f'./models/{selected_model}/scibert-scivocab-uncased', from_id, to_id)

    display(HTML(f'<h3>Seed: <a href="{from_doc["url"]}">{from_doc["title"]}</a></h3>'))
    display(HTML(f'<h3>Target: <a href="{to_doc["url"]}">{to_doc["title"]}</a></h3>'))
    display(HTML(f'<h4>Predicted labels:</h4><pre>{", ".join(pred_labels)}</pre>'))
```

Рисунок 22 - Фрагменты кода для проверки работы моделей

Выполним все фрагменты кода без изменений и проанализируем вывод. В служебных фрагментах кода были выполнены все необходимые установки, а в интересующем нас фрагменте проверки моделей были выведены результаты выполнения предсказания сходства статей (см. Рисунок). В результате были выведены ссылки на исходную (Seed) и целевую (Target) статью, а также метки классов, обозначающие разделы, в которых, по мнению моделей, статьи совпадают. В данном случае сходство статей определено во «Введении» (introduction) и «Эксперименте» (experiment).

Try your own papers

1. Select model (either trained on ACL Anthology or CORD-19)
2. Select input documents by paper IDs.

All IDs from [Semantic Scholar API](#) are supported (DOI, ArXiv ID, PubMed ID, ACL ID)

```
[5] selected_model = 'acl' #@param ["acl", "cord19"]
    from_id = '10.3115/1667583.1667640' #@param {type:"string"}
    to_id = '10.1145/1367497.1367545' #@param {type:"string"}

selected_model: acl
from_id: "10.3115/1667583.1667640"
to_id: "10.1145/1367497.1367545"

[6] # Perform predictions
    pred_scores, pred_labels, from_doc, to_doc = get_prediction(f'./models/{selected_model}/scibert-scivocab-uncased', from_id, to_id)

    display(HTML(f'<h3>Seed: <a href="{from_doc["url"]}">{from_doc["title"]}</a></h3>'))
    display(HTML(f'<h3>Target: <a href="{to_doc["url"]}">{to_doc["title"]}</a></h3>'))
    display(HTML(f'<h4>Predicted labels:</h4><pre>{", ".join(pred_labels)}</pre>'))

Seed: Query Segmentation Based on Eigenspace Similarity
Target: Unsupervised query segmentation using generative language models and wikipedia
Predicted labels:
introduction, experiment
```

Рисунок 23 - Результат предсказания на примере авторов

Действительно, данные статьи схожи по содержанию в том плане, что описывают подходы к query segmentation. Соответственно, логично было бы предположить, что будут совпадать введения этих статей, а также будут описаны эксперименты схожей направленности.

Выполним аналогичное предсказание, но с помощью модели cord19 (см. Рисунок 24). Результат оказался другим. Теперь модель предсказывает, что статьи могут совпадать по «другим» (other) разделам. Однако в любом случае подтверждено неоспоримое сходство тематики статей.

```
[7] selected_model = 'cord19' #@param ["acl", "cord19"]
from_id = '10.3115/1667583.1667640' #@param {type:"string"}
to_id = '10.1145/1367497.1367545' #@param {type:"string"}
```

selected_model: cord19

from_id: " 10.3115/1667583.1667640"

to_id: " 10.1145/1367497.1367545"

```
[8] # Perform predictions
pred_scores, pred_labels, from_doc, to_doc = get_prediction(f'./models/{selected_model}/scibert-scivocab-uncased', from_id, to_id)

display(HTML(f'<h3>Seed: <a href="{from_doc["url"]}">{from_doc["title"]}</a></h3>'))
display(HTML(f'<h3>Target: <a href="{to_doc["url"]}">{to_doc["title"]}</a></h3>'))
display(HTML(f'<h4>Predicted labels:</h4><pre>{"", ".join(pred_labels)}</pre>'))
```

Seed: [Query Segmentation Based on Eigenspace Similarity](#)
Target: [Unsupervised query segmentation using generative language models and wikipedia](#)
Predicted labels:
other

Рисунок 24 - Использование модели cord19 на тех же статьях

Ради эксперимента попробуем определить сходство статьи с самой собой. Скопируем идентификатор статьи из поля from_id в поле to_id (см. Рисунок 25). Сходство было подтверждено. Модель предсказала, что статьи будут совпадать во «введении» и «другом» разделе.

```
[9] selected_model = 'acl' #@param ["acl", "cord19"]
from_id = '10.3115/1667583.1667640' #@param {type:"string"}
to_id = '10.3115/1667583.1667640' #@param {type:"string"}
```

selected_model: acl

from_id: " 10.3115/1667583.1667640"

to_id: " 10.3115/1667583.1667640"

```
[10] # Perform predictions
pred_scores, pred_labels, from_doc, to_doc = get_prediction(f'./models/{selected_model}/scibert-scivocab-uncased', from_id, to_id)

display(HTML(f'<h3>Seed: <a href="{from_doc["url"]}">{from_doc["title"]}</a></h3>'))
display(HTML(f'<h3>Target: <a href="{to_doc["url"]}">{to_doc["title"]}</a></h3>'))
display(HTML(f'<h4>Predicted labels:</h4><pre>{"", ".join(pred_labels)}</pre>'))
```

Seed: [Query Segmentation Based on Eigenspace Similarity](#)
Target: [Query Segmentation Based on Eigenspace Similarity](#)
Predicted labels:
introduction, other

Рисунок 25 - Предсказание сходства одной и той же статьи

Найдем две статьи, схожие по тематике, но в названии которых и аннотации не упоминаются одни и те же слова. В результате поиска была выбрана статья 2014 года «Convolutional Neural Networks for Sentence Classification» и статья 2019 года «BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding». Обе статьи рассматривают задачу обарботки естественных языков, но совершенно разными методами. Авторы первой статьи не знали ничего о BERT на момент написания своих материалов. Выполним предсказание: пусть первая статья будет исходная, а вторая

целевой (см. Рисунок 26). В результате модель предсказала, что статьи будут сходны по своему введению, т.е. во введении первой статьи может упоминаться вторая. Если не брать в расчет дату публикации статьи, о которой модель не знает, такое предсказание можно считать правильным. Авторы исходной статьи могли, например, упомянуть BERT как один из подходов к классификации предложений.

Try your own papers

1. Select model (either trained on ACL Anthology or CORD-19)
2. Select input documents by paper IDs.

All IDs from [Semantic Scholar API](#) are supported (DOI, ArXiv ID, PubMed ID, ACL ID)

```
[11] selected_model = 'acl' #@param ["acl", "cord19"]
from_id = '10.3115/v1/D14-1181' #@param {type:"string"}
to_id = '10.18653/v1/N19-1423' #@param {type:"string"}
```

selected_model: acl

from_id: "10.3115/v1/D14-1181"

to_id: "10.18653/v1/N19-1423"

```
[12] # Perform predictions
pred_scores, pred_labels, from_doc, to_doc = get_prediction(f'./models/{selected_model}/scibert-scivocab-uncased', from_id, to_id)

display(HTML(f'<h3>Seed: <a href="{from_doc["url"]}">{from_doc["title"]}</a></h3>'))
display(HTML(f'<h3>Target: <a href="{to_doc["url"]}">{to_doc["title"]}</a></h3>'))
display(HTML(f'<h4>Predicted labels:</h4><pre>{"", ".join(pred_labels)}</pre>'))
```

Seed: [Convolutional Neural Networks for Sentence Classification](#)

Target: [BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding](#)

Predicted labels:
introduction

Рисунок 26 - Предсказание сходства близких по тематике статей

Теперь выберем статьи из совершенно разных сфер. Выберем статью с оптимизацией времени посадки в самолет при использовании автобусов «A Two-Door Airplane Boarding Approach When Using Apron Buses» и уже классическую статью «Attention is All you Need». Сделаем вторую статью исходной, а первую – целевой. Выполним предсказание (см. Рисунок 27). Действительно, в результате модель предсказала, что статьи никак между собой не связаны (метка «None»).

```
[18] selected_model = 'acl' #@param ["acl", "cord19"]
from_id = 'arXiv:1706.03762' #@param {type:"string"}
to_id = '10.3390/SU10103619' #@param {type:"string"}
```

selected_model: acl

from_id: "arXiv:1706.03762"

to_id: "10.3390/SU10103619"

```
[19] # Perform predictions
pred_scores, pred_labels, from_doc, to_doc = get_prediction(f'./models/{selected_model}/scibert-scivocab-uncased', from_id, to_id)

display(HTML(f'<h3>Seed: <a href="{from_doc["url"]}">{from_doc["title"]}</a></h3>'))
display(HTML(f'<h3>Target: <a href="{to_doc["url"]}">{to_doc["title"]}</a></h3>'))
display(HTML(f'<h4>Predicted labels:</h4><pre>{"", ".join(pred_labels)}</pre>'))
```

Seed: [Attention is All you Need](#)

Target: [A Two-Door Airplane Boarding Approach When Using Apron Buses](#)

Predicted labels:
none

Рисунок 27 - Определение сходства никак не связанных статей

5. ВЫВОДЫ

В рамках домашнего задания был выполнен обзор теоретических и практических материалов, связанных со статьей «Aspect-based Document Similarity for Research Papers».

В статье рассматривалась попарная многоуровневая многоклассовая классификация научных статей для вычисления оценки сходства документов на основе аспектов. Названия разделов являются аспектами статьи и соответствующим образом помечаются цитаты, встречающиеся в этих разделах. Исследуемые модели обучены прогнозировать цитирование и соответствующую метку на основе названия статьи и аннотации. Оценивались модели Transformer BERT, Covid-BERT, SciBERT, ELECTRA, RoBERTa и XLNet, а также baseline модель LSTM в двух научных корпусах ACL Anthology и CORD-19. В целом, SciBERT показал лучшие результаты в экспериментах. Несмотря на сложную задачу, SciBERT предсказал сходство документа на основе аспектов с оценкой F1 вплоть до 0,83. Результаты SciBERT мотивируют дальнейшие исследования в этом направлении. Кажется разумным включить задачу подобию документа на основе аспектов в качестве новой цели предварительного обучения в архитектуре Transformers. В будущем авторы статьи планируют интегрировать сходство документов на основе аспектов в рекомендательную систему. Таким образом, проведение большого исследования пользователей позволит подтвердить выводы о том, что сходство документов на основе аспектов действительно помогает пользователям находить более релевантные рекомендации. Однако обширный эмпирический анализ уже показывает, что Трансформеры хорошо подходят для правильного вычисления аспектного сходства документов для исследовательских работ.

В практической части был выполнен обзор содержимого репозитория авторов статьи на GitHub, а также было выполнено тестирование моделей в демо-примере. В результате тестирования можно подтвердить, что модели действительно были обучены определять сходство статей по разделам.

6. СПИСОК ИСТОЧНИКОВ

1. Browse State-of-the-Art. – Текст. Изображение: электронные // Papers With Code : [сайт]. – URL: <https://paperswithcode.com/sota> (дата обращения: 25.05.2022).
2. Document Classification. – Текст. Изображение: электронные // Papers With Code : [сайт]. – URL: <https://paperswithcode.com/task/document-classification> (дата обращения: 25.05.2022).
3. Aspect-based Document Similarity for Research Papers. – Текст. Изображение: электронные // Papers With Code : [сайт]. – URL: <https://paperswithcode.com/paper/aspect-based-document-similarity-for-research> (дата обращения: 25.05.2022).
4. Transformer в картинках. – Текст. Изображение: электронные // Хабр : [сайт]. – URL: <https://habr.com/ru/post/486358/> (дата обращения: 25.05.2022).
5. aspect-document-similarity. – Текст. Изображение: электронные // GitHub : [сайт]. – URL: <https://github.com/malteos/aspect-document-similarity> (дата обращения: 25.05.2022).
6. Aspect-based Document Similarity for Research Papers. – Текст. Изображение: электронные // arXiv : [сайт]. – URL: <https://arxiv.org/abs/2010.06395> (дата обращения: 25.05.2022).
7. Releases - malteos/aspect-document-similarity. – Текст. Изображение: электронные // GitHub : [сайт]. – URL: <https://github.com/malteos/aspect-document-similarity/releases> (дата обращения: 25.05.2022).
8. datasets. – Текст. Изображение: электронные // GitHub : [сайт]. – URL: <https://github.com/huggingface/datasets> (дата обращения: 25.05.2022).
9. malteos. – Текст. Изображение: электронные // Hugging Face : [сайт]. – URL: <https://huggingface.co/malteos> (дата обращения: 25.05.2022).
10. demo.ipynb. – Текст. Изображение: электронные // Colaboratory : [сайт]. – URL: <https://colab.research.google.com/github/malteos/aspect-document-similarity/blob/master/demo.ipynb> (дата обращения: 25.05.2022).