САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Гао Тянь

Магистерская диссертация

Разработка новых методов анализа настроений, анализа текста в социальных сетях для прогнозирования предпочтения пользователей

Направление 02.04.02

Фундаментальная информатика и информационные технологии

Магистерская программа «Технологии баз данных»

Научный руководитель
Кафедра технологии программирования
доцент
Сергеев Сергей Львович
Рецензент
Пашкевич Василий Эрикович

Санкт-Петербург
2021

SAINT-PETERSBURG STATE UNIVERSITY

GaoTian

Graduation project

Development of new methods for sentiment analysis, text analysis in social networks to predict user preferences

Master level 02.04.02

«Fundamental Informatics and Information Technology»

«Database Technologies» №19/5503/1

Research supervisor:
Department of Programming Technology
Associate Professor
Sergeev Sergey Lvovich
Reviewer
Pashkevich Vasilii Ericovich

Saint-Petersburg
2021

Содержание

Введение

Анализ настроения текста: также известен как поиск мнений, анализ тенденций и т.д. Проще говоря, это процесс анализа, обработки, обобщения и рассуждения о субъективном тексте с эмоциональным подтекстом. Интернет генерирует большое количество пользовательских комментариев к ценной информации, такой как люди, события, продукты и так далее. Эти отзывы выражают различные эмоциональные окраски и тенденции людей, такие как счастье, гнев, печаль, радость, критика, похвала и так далее. Исходя из этого, потенциальные пользователи могут просматривать эти субъективные комментарии, чтобы понять общественное мнение о событии или продукте.

Анализ настроений или поиск мнений - это процесс, с помощью которого оцениваются мнения, эмоции и отношение людей к таким объектам, как продукты, услуги и организации. Эта область быстро развивалась и взлетала благодаря стремительному росту социальных сетей в Интернете, таких как обзоры продуктов, обсуждения на форумах, твиты и сайты обмена фотографиями, поскольку впервые в истории человечества такой огромный объем цифровых данных был зафиксирован в форме. С начала 2000-х годов анализ настроений стал одним из наиболее активных направлений исследований в области обработки естественного языка (NLP). Также проводятся обширные исследования в области интеллектуального анализа данных, интеллектуального анализа веб-информации, интеллектуального анализа текста и информационного поиска. Фактически, она распространилась от компьютерных наук до науки управления и социальных наук, таких как маркетинг, финансы, политология, коммуникации, медицина и даже история, вызывая общий интерес во всем обществе из-за своей важности для бизнеса. Такое распространение объясняется тем, что мнения занимают центральное место в том, что почти все виды человеческой деятельности в значительной степени заботятся о том, что думают другие. По этой причине, когда нам нужно принять решение, мы часто обращаемся к мнению других людей. Это касается не только предприятий, но и частных лиц.

Сегодня, если человек хочет купить потребительский товар, он больше не ограничивается тем, что спрашивает мнение своих друзей и родственников, поскольку в Интернете существует множество пользовательских обзоров и публичных форумов для обсуждения товаров, где мы можем узнать то, что хотим знать в обзорах, и, возможно, получить неожиданную выгоду. Для организации может отпасть необходимость в проведении опросов,

анкетирования и фокус-групп для изучения общественного мнения.

В последние годы мы стали свидетелями того, как статьи, которые перекраивают имиджи компаний, обсуждают жизнь знаменитостей, вызывают общественные настроения и эмоции через социальные сети, помимо всего прочего, оказывают глубокое влияние на наши социальные и политические системы, и такие статьи мобилизуют изменения в массовой политике. В то время как мы сетуем на боязливость человеческой речи, мы также должны признать огромные этические проблемы, которые порождает стремительный рост социальных сетей. Так появился анализ настроения, который используется для предотвращения возникновения злонамеренных или ложных событий.

Вообще говоря, цель анализа настроений - выяснить отношение говорящего или автора к определенным темам или в ответ на полярные точки зрения в тексте. Это отношение может быть его или ее личным суждением или оценкой, или это может быть эмоциональное состояние момента (то есть эмоциональное состояние автора в момент, когда было сделано заявление), или предполагаемая эмоциональная коммуникация автора (то есть эмоция, которую автор хочет, чтобы испытал читатель).

Постановка задачи

С быстрым развитием электронной коммерции все больше потребителей размещают отзывы о товарах на интернет-платформах. Столкнувшись с более прямой обратной связью от пользователей, компании столкнулись с серьезной проблемой интеграции информации обратной связи и быстрого реагирования на нее. Рост платформ социальных сетей, таких как Twitter, также предоставил широкую базу данных для анализа общественного мнения и опросов общественного мнения, основанных на онлайн-данных. В отличие от общего анализа настроений, анализ настроений на основе атрибутов или характеристик является более "гранулированным" и нацелен на предоставление серии кратких выражений, основанных на информации из обзоров, для указания предпочтений группы потребителей по каждому атрибуту продукта.

В этой статье мы рассмотрим, какие методы используются в анализе настроений и как они применяются. Сравнивая сильные и слабые стороны этих методов, мы нашли новый способ выполнения задачи анализа настроений с пятью категориями (наша цель - классифицировать комментарии как: очень нравится, нравится, нейтральный, не нравится и очень не нравится).

Введение в процесс эмоционального анализа

Анализ настроений - это процесс анализа, обработки, обобщения и рассуждения о субъективных текстах с эмоциональным подтекстом. В соответствии с различными категориями обрабатываемых текстов, его можно разделить на анализ настроений на основе новостных обзоров и анализ настроений на основе обзоров продуктов. Среди них первый в основном используется для мониторинга мнений и прогнозирования информации, а второй может помочь пользователям понять репутацию определенного продукта в общественном сознании.

Существует два основных распространенных метода анализа полярности настроений: методы, основанные на ручном составлении словарей настроений, и методы автоматических систем, основанные на машинном обучении.

Подход машинного обучения является более точным, поскольку лексическое соответствие подвержено большим ошибкам из-за богатства семантических выражений, а подход машинного обучения - нет. И его можно использовать в более разнообразных сценариях. Будь то предметно-целевая классификация или классификация положительных и отрицательных настроений, машинное обучение может справиться с поставленной задачей. При этом нет необходимости спускаться на уровень слов, предложений и грамматики, как в случае ручного построения лексикона настроений.

Ручной подход к созданию словарей настроений применим к гораздо более широкому кругу корпусов, будь то товары, такие как мобильные телефоны или компьютеры, или корпусы, такие как книжные обзоры или рецензии на фильмы. Но машинное обучение, с другой стороны, чрезвычайно зависит от корпуса, и если взять классификатор, обученный на корпусе сотовых телефонов, и классифицировать его для книжного обзора, то он обречен на провал.

Использование машинного обучения для анализа настроений, что можно выразить по-другому с тем же смыслом, заключается в использовании контролируемых (требующих ручной маркировки категорий) методов машинного обучения для классификации текста.

Это принципиально отличается от ручного подхода к созданию словаря настроений для сопоставления. Ручное составление словаря для сопоставления - это прямое вычисление слов настроения в тексте для получения их баллов склонности к настроению. В отличие от этого, идея подхода машинного обучения заключается в том, чтобы сначала выбрать часть текста, выражающую позитивные настроения, и часть текста, выражающую негативные настроения, и обучить их с помощью методов машинного обучения, чтобы получить классификатор настроений. Затем этот классификатор настроений используется для дихотомической классификации всех текстов на положительные и отрицательные. Окончательная классификация может дать категорию, например, 0 или 1 для текста или значение вероятности, например, "этот текст с вероятностью 90% является положительным и с вероятностью 10% -

отрицательным".

Но наша задача - это задача мелкозернистой классификации пяти настроений, поэтому прежде чем приступить к выполнению задачи анализа настроений, нам необходимо иметь общее представление об используемых методах и процессе, которому следует следовать при выполнении задачи анализа настроений.

1. Языковая модель

При построении языковой модели возникает проблема слишком большого количества свободных параметров, и для решения этой проблемы мы вводим марковское предположение, что вероятность появления произвольного слова связана только с конечным числом n слов, которые встречаются перед ним. Статистическая модель языка, основанная на вышеуказанных предположениях, называется N-граммной моделью языка. Как правило, значение n не может быть слишком большим, иначе остается проблема слишком большого количества свободных параметров.

(1) Когда n=1, т.е. появление слова не зависит от окружающих его слов, такой вид мы называем униграммой, т.е. монадической моделью языка, когда величиной свободного параметра является размер лексикона V.

(2) Когда n=2, т.е. когда появление слова связано только с одним словом перед ним, такое мы называем биграммой, называется бинарной моделью языка, также называется цепью Маркова первого порядка, когда величина свободного параметра равна V^2.

(3) Когда n=3, т.е. появление слова связано только с двумя словами перед ним, это называется триграммой, называется тернарной моделью языка, также называется цепью Маркова второго порядка, когда порядок величины свободного параметра равен V^3.

В общем случае используются только указанные выше значения, поскольку, как видно из вышеизложенного, порядок величины свободных параметров экспоненциально кратен значению n. С точки зрения эффективности модели, теоретически, чем больше значение n, тем лучше результаты. Однако по мере увеличения значения n величина улучшения эффекта уменьшается. Существует также проблема надежности и различимости, чем больше параметров, тем лучше различимость, но в то же время количество экземпляров отдельных параметров становится меньше, что снижает надежность.

2. Разделение слов

(1) Проблема разделения слов в английском языке

(Обратите внимание: русский и английский языки на этом этапе разделения слов обрабатываются практически одинаково, здесь я буду использовать английский язык в качестве примера)

В английском языке в качестве разделителей используются естественные пробелы, что удобно для разделения слов.

Английские слова имеют богатые морфологические преобразования. Чтобы справиться с этими сложными преобразованиями, в английском NLP по сравнению с китайским существуют некоторые уникальные этапы обработки, которые мы называем лексико-морфологической редукцией (Lemmatization) и извлечением стволов слов (Stemming).

Лексическая редукция: "does", "done", "doing", "did" должны быть восстановлены до "do" путем лексической редукции.

Корень слова: слова "cities", "children", "teeth" необходимо преобразовать в основные формы "city", "child", "tooth".

В то же время следует обратить внимание на падеж каждого слова при выполнении задания по расщеплению английского слова (расщепление русского слова), например: "Stop", следует преобразовать в "stop"

Инструмент для разделения английских слов: Keras,Spacy,Gensim,NLTK

(2) Проблема разделения китайских слов

Разделение китайских слов является очень важным компонентом обработки китайского естественного языка и имеет относительно долгую историю исследований как в академических, так и в промышленных кругах, и есть несколько более зрелых решений. Разделение китайских слов - это фундаментальный шаг в обработке китайского текста и базовый модуль для китайского взаимодействия человека и компьютера на естественном языке. В отличие от английского языка, в китайских предложениях нет границ слов, поэтому при обработке китайского естественного языка обычно необходимо сначала выполнить разделение слов, а эффект разделения слов напрямую влияет на эффект лексических и синтаксических деревьев и других модулей. Конечно, разделение слов - это всего лишь инструмент, и сценарии бывают разные, и требования тоже разные. При взаимодействии человека и компьютера на естественном языке зрелые алгоритмы разделения китайских слов могут достичь лучшего эффекта обработки естественного языка и помочь компьютерам понять сложный китайский язык.

Инструмент для разделения китайских слов: Hanlp,ansj,LTP,KCWS ,jieba

3. Векторизация слов

После выполнения задания по разделению слов мы получили каждое слово.

Но компьютеры не могут понять человеческий язык, потому что они могут распознавать только 0 и 1.

(Для получения дополнительной информации о составе компьютеров и о том, почему они распознают только 0 и 1, а также об основных структурах компьютеров, вы можете обратиться к учебнику "Архитектуры вычислительных систем" [1], в котором можно узнать о составе и распознавании компьютеров, а также об основных структурах современных)

Итак, после получения разделенных слов нам необходимо преобразовать человеческий язык в компьютерный, и цель техники векторизации слов - преобразовать естественный человеческий язык в "машинный язык", который может быть распознан компьютером.

3.1 Репрезентативность слов

3.1.1 Дискретное представление (one-hot representation)

Традиционные подходы к естественной семантической обработке, основанные на правилах или статистике, рассматривают слово как атомарный символ
называется одномоментным представлением. Одномоментное представление представляет каждое слово в виде длинного вектора. Размерность этого вектора - размер таблицы слов, и только одно измерение вектора имеет значение 1, а остальные - 0. Это измерение представляет текущее слово.
Пример.
Apple [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, ...]

Одноточечное представление эквивалентно присвоению идентификатора каждому слову, что приводит к представлению, которое не показывает связи между словами. Кроме того, одноточечное представление приведет к очень большому пространству признаков, но также дает то преимущество, что многие прикладные задачи линейно разделимы в высокоразмерном пространстве.

3.1.2 Распределенное представление (distribution representation)

Встраивание слов относится к преобразованию слова в распределенное представление, которое представление представляет слово как непрерывный плотный вектор постоянной длины.

Преимущества распределенного представления:

(1) Между словами существуют отношения сходства.

Именно понятие "расстояние" между словами является очень полезным для многих задач обработки естественного языка.

(2) Содержит больше информации.

Векторы слов могут содержать больше информации, и каждое измерение имеет конкретное значение.

3.2 Генерировать векторы слов

В этом подразделе кратко описаны методы генерации векторов слов.

Существует множество способов создания векторов слов, все они следуют идее, что значение любого слова может быть представлено окружающими его словами.

Способы генерации векторов слов можно разделить на методы, основанные на статистике, и методы, основанные на языковых моделях.

3.2.1 На основе статистических методов

Подсчитывая количество совпадений слов в окне заранее определенного размера, количество совпадающих слов вокруг слова используется в качестве вектора текущего слова. В частности, мы определяем представление слова путем построения матрицы совпадений из большого корпуса текстов.

Например, существует следующий корпус:

“I like deep learning.”

“I like NLP.”

“I enjoy flying.”

Тогда его матрица коокуррентности имеет следующий вид(Таб.1).

counts	I	like	enjoy	deep	learning	NLP	fiying	.
I	0	2	1	0	0	0	0	0
like	2	0	0	1	0	1	0	0
enjoy	1	0	0	0	0	0	1	0
deep	0	1	0	0	1	0	0	0
learning	0	0	0	1	0	0	0	1
NLP	0	1	0	0	0	0	0	1
flying	0	0	1	0	0	0	0	1
.	0	0	0	0	1	1	1	0

Таб.1

Далее нам необходимо решить проблему разреженности данных и катастрофы размерности.

3.2.2 SVD (разложение по сингулярным значениям)

Поскольку дискретный вектор слов, полученный на основе матрицы кокуррентности, страдает от проблемы высокой размерности и разреженности, естественной идеей решения является уменьшение размерности исходного вектора слов для получения плотного непрерывного вектора слов.

Для матрицы из подраздела 2.1.1 выполняется SVD-разложение для получения матричной ортогональной матрицы U. Нормализация U дает матрицу в следующем виде (Таб.2)

I	0.24	0.21	0.10	0.38	-0.18	-0.18	-0.42	-0.06
like	0.20	0.82	-0.17	0.31	0.18	-0.23	0.13	0.14
enjoy	0.37	0.64	0.16	0.00	-0.58	0.64	0.00	-0.31
deep	0.36	0.38	0.35	-0.07	0.45	0.08	0.55	-0.47
learning	0.40	0.52	-0.5	-0.43	0.35	0.16	-0.47	-0.40
NLP	0.35	0.35	-0.22	-0.19	0.13	0.49	0.21	0.66
flying	0.41	0.42	-0.40	-0.38	-0.51	-0.43	0.42	-0.12
.	0.38	0.58	0.59	-0.62	-0.03	-0.23	-0.26	0.24

Таб.2

SVD дает плотную (dense) матрицу слов, которая обладает многими хорошими свойствами: семантически похожие слова похожи в векторном пространстве и даже могут в некоторой степени отражать линейную связь между словами.

3.3 Языковая модель (language model)

Языковая модель генерирует векторы слов путем обучения нейросетевой языковой модели (NNLM: neural network language model), при этом векторы слов являются побочными выходами языковой модели. Основная идея NNLM заключается в предсказании слов, которые появляются в контекстном окружении, и это предсказание контекстного окружения, по сути, является обучением статистическим характеристикам совпадений. .

Некоторые из наиболее известных методов генерации векторов слов с помощью нейросетевой языковой модели - Skip-gram, CBOW, LBL, NNLM, C&W, GloVe и др.

Это также широко используется в задачах анализа настроений промышленного уровня. Наиболее представительным из них является хорошо известный - word2vec.

Этот метод не был создан трудом одного ученого, а совершенствовался разными людьми.

Миколов и др. предложили модели CBOW (Continuous Bagof-Words) и Skip-gram в своей работе 2013 года [2]. Основной целью разработки двух моделей

является получение векторов слов более эффективным способом. Поэтому они получили эти две модели путем упрощения существующих моделей и сохранения основных частей, основываясь на своем предыдущем опыте работы с моделями NNLM, RNNLM и C&W.

В 2013 году в [3] Миколов предложил две стратегии обучения моделей скип-грамм: иерархический Softmax и отрицательная выборка.

Миколов предложил doc2vec в 2014 году [4].

Оригинальная статья, написанная Миколовым, не была хорошо понята, и с тех пор было представлено подробное выведение оригинальной статьи [5,6].

И в этой статье мы собираемся использовать метод под названием GloVe. После того, как Томаш Миколов и др. предложили word2vec в 2013 году, Glove был предложен Джеффри Пеннингтоном, Ричардом Сочером, Кристофером Д. Мннингом в 2014 году [7].

Суть модели Glove включает в себя новейшие методы глобальной матричной факторизации и локального захвата контекстного окна, так что учитывается как глобальная статистическая информация, так и информация локального окна; результаты также показывают, что Glove достигает хорошей производительности в задачах, связанных со словами.

Также я буду использовать Bert в качестве перекрестной ссылки. Bert - это новый фреймворк, открытый Google, BERT может использоваться в системах вопросов и ответов, анализа настроений, фильтрации спама, распознавания именованных сущностей, кластеризации документов и других задачах. BERT концептуально прост и эмпирически силен. Он получил новые современные результаты на одиннадцати задачах обработки естественного языка, включая повышение оценки GLUE до 80,5% (абсолютное улучшение на 7,7%), точности MultiNLI до 86,7% (абсолютное улучшение на 4,6%), теста F1 для ответов на вопросы SQuAD v1.1 до 93,2 (абсолютное улучшение на 1,5 балла) и теста F1 для SQuAD v2.0 до 83,1 (абсолютное улучшение на 5,1 балла).[8].

Здесь я просто использую модель встраивания слов Bert, потому что вся система Bert очень сложна и очень требовательна к производительности компьютера, а у меня нет такого высокопроизводительного компьютера.

Я подробнее остановлюсь на конкретных принципах работы Bert и GloVe в разделе о реализации.

4. Алгоритм анализа настроения

После выполнения описанных выше действий мы можем получить все слова, которые были векторизованы. Это также означает, что компьютер правильно определил каждое слово. Теперь нам нужно выбрать алгоритм для нашей задачи, задача анализа настроений, в принципе, является классификацией текста. Возьмем в качестве примера эту задачу - задачу пяти классификаций, это как если бы у нас было пять корзин и мы клали различные предметы в соответствующие корзины в соответствии с правилами.

Правила, упомянутые выше, относятся к алгоритму, который нам нужно

найти.

Существуют различные методы и алгоритмы для реализации анализа настроений, которые можно классифицировать как.

1. на основе правил: проведение анализа настроения на основе набора правил, разработанных вручную.

2. автоматизированные системы: которые опираются на методы машинного обучения для получения знаний из данных.

4.1 На основе правил

Как правило, подходы, основанные на правилах, определяют набор правил с помощью сценариев, которые используются для классификации настроений.

Правила могут использовать различные исходные данные, такие как.

(1) классические методы NLP, такие как строфы слов, символы, лексическая аннотация и синтаксический разбор.

(2) Другие ресурсы, такие как словари (т.е. списки слов и выражений).

Базовый пример реализации на основе правил выглядит следующим образом.

Определите два списка поляризованных слов (например, отрицательные слова, такие как плохой, худший, уродливый и т.д., и положительные слова, такие как хороший, лучший, красивый и т.д.).

Дан текст.

Подсчитайте количество положительных слов, встречающихся в тексте.

Подсчитайте количество отрицательных слов, встречающихся в тексте.

Возвращает положительные настроения, если число положительных вхождений превышает число отрицательных вхождений слов, и наоборот, возвращает отрицательные настроения. В противном случае верните нейтральное положение.

Этот алгоритм очень прост, поскольку не учитывает, как слова сочетаются в последовательности. Возможна более сложная обработка, но такие алгоритмы быстро становятся очень сложными. Их может быть очень сложно поддерживать, поскольку для поддержки пар новых выражений и словарей могут потребоваться новые правила. Кроме того, добавление новых правил может привести к плохим результатам из-за смешения с предыдущими правилами. Поэтому эти алгоритмы требуют больших усилий по ручной настройке и поддержанию правил.

4.2 Автоматизированные системы

В отличие от систем, основанных на правилах, автоматизированные подходы полагаются не на составленные вручную правила, а на машинное обучение. Задачи анализа настроения обычно моделируются как задачи классификации, где текст подается на классификатор, который затем выдает

соответствующую категорию, например, хорошую, плохую или нейтральную.

Классификаторы машинного обучения, как правило, могут быть реализованы в виде следующих шагов(Рис.3):

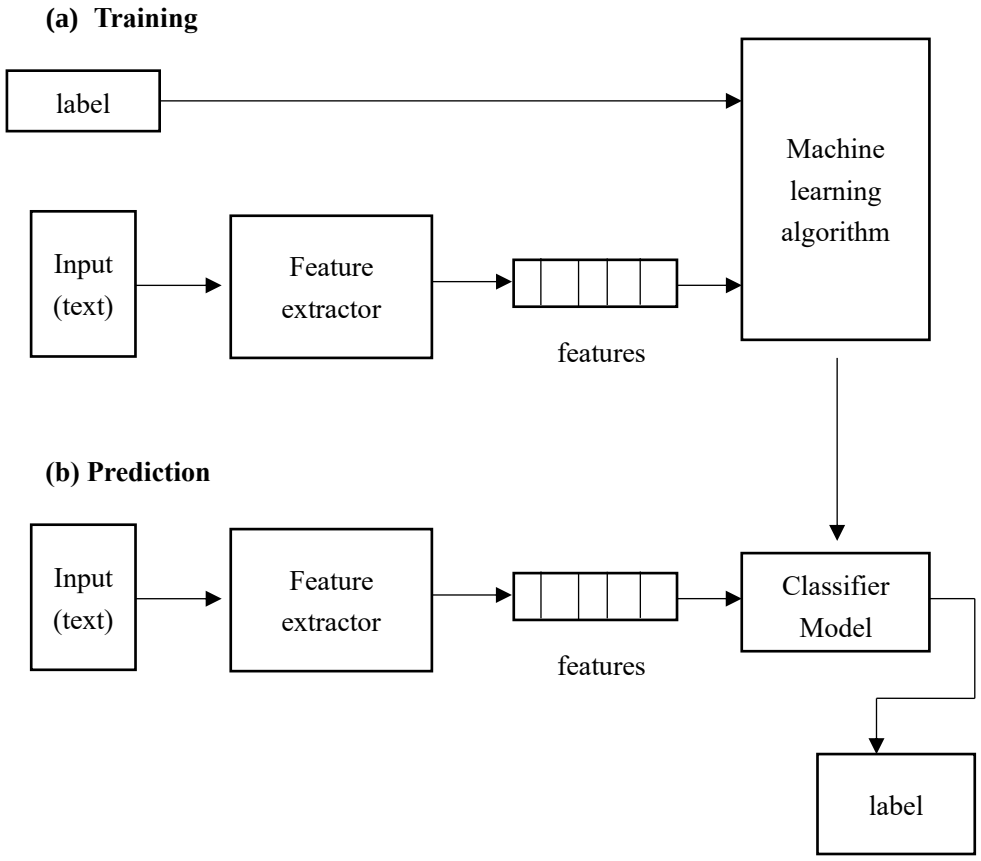


Рис.3

(1) Процессы обучения и прогнозирования:

В процессе обучения (a) наша модель учится ассоциировать определенные входы (т.е. текст) с соответствующими выходами (метками) на основе обучающих образцов. Экстрактор признаков преобразует текстовые данные в векторы признаков. Пары векторов признаков и меток (например, положительных, отрицательных или нейтральных) подаются в алгоритм машинного обучения для создания модели.

В процессе предсказания (b) используется экстрактор признаков, который берет невидимые текстовые данные и преобразует их в векторы признаков. Эти векторы признаков затем вводятся в модель, которая генерирует прогностические метки (положительные, отрицательные или нейтральные).

(2) Извлечение признаков из текста:

Первым шагом в классификаторе текста для машинного обучения является преобразование текста в числовое представление, обычно в вектор. Как правило, каждый компонент вектора представляет собой частоту слов или выражений в заранее определенном словаре (например, лексиконе полярных слов). Этот

процесс называется извлечением признаков или векторизацией текста, и классическим подходом был пакет слов или пакет инграмм с частотами.

(3) Алгоритмы классификации:

Алгоритмы классификации обычно используют статистические модели, такие как NaïveBayes, логистическая регрессия, машины опорных векторов или нейронные сети:

(a) NaïveBayes: вероятностная статистическая модель, которая использует теорему Байеса для предсказания класса текста.

(b) Линейная регрессия: очень известный статистический алгоритм для прогнозирования определенных значений (Y) с учетом набора признаков (X).

(c) Support vector machine(SVM): невероятностная модель, которая использует представление текстовых примеров в виде точек в многомерном пространстве. Примеры отображаются таким образом, что различные классы (эмоции) примеров принадлежат к различным областям этого пространства. Затем новые тексты отображаются в этом же пространстве и на основании того, в какой регион они попадают, прогнозируется их принадлежность к той или иной категории.

(d) Глубокое обучение: разнообразный набор алгоритмов, которые пытаются обрабатывать данные с помощью искусственных нейронных сетей, имитируя работу человеческого мозга.

В этой задаче классификации пяти объектов я намерен использовать фреймворк глубокого обучения, CNN, поскольку традиционные алгоритмы машинного обучения уже не способны удовлетворить наши потребности в тонкой детализации.

CNN были первоначально использованы для решения проблем с изображениями и привели к огромным изменениям в обработке изображений (CV). Представим это так, в тексте, если каждое слово можно представить вектором строк, то предложение можно представить матрицей, в таком случае обработка текста похожа на обработку изображений.

Я подробно остановлюсь на принципах реализации нейронной сети CNN в разделе конкретной реализации модели.

Принцип и конкретные шаги по реализации

Глава 1. Системная среда и инструментарий

Я должен обратить ваше внимание на то, что для выполнения этой задачи мне пришлось заменить свой компьютер на новый из-за аппаратного сбоя.

1) Компьютерная система, на которой генерировались векторы слов, представляла собой Ubuntu 20.04, стриженную версию linux, python версии 3.6 и Tensorflow версии 1.10, с использованием bert-as-service для генерации векторов

слов.

2) Реализация кода CNN

Я использую Windows 10. язык программирования - python 3.8. фреймворк глубокого обучения - tensorflow2.4.1, использующий основной вычислительный модуль CPU (если на вашем компьютере есть GPU, вы можете установить версию, которая использует GPU в качестве основного вычислительного модуля). numpy версия-1.18.5. pandas версия-1.2.3. Если вы хотите использовать Anaconda в качестве инструмента управления средой python, моя версия Anaconda - 4.10.1 (я рекомендую использовать Anaconda в качестве инструмента управления средой python).

Глава 2. Доступ к информации

Наборы данных, которые мы используем в задачах анализа настроений, обычно получают двумя способами.

1. общедоступные наборы данных (например, такие наборы данных, как конкурсы kaggle).

2. использование веб-краулеров для сбора данных.

2.1 Общедоступные наборы данных

Первый способ получения набора данных - это использование общедоступного набора данных с открытым исходным кодом.

В настоящее время существует очень большое количество наборов данных с открытым исходным кодом, охватывающих широкий спектр областей, вот несколько распространенных сайтов, где можно найти наборы данных и некоторые наборы изображений, обычно используемые в компьютерном зрении.

Наборы данных Kaggle:Каждый набор данных - это небольшое сообщество, где пользователи могут обсуждать данные, находить публичный код или создавать собственные проекты на ядре. Содержит широкий спектр реальных наборов данных.

Amazon Datasets: Этот источник данных содержит наборы данных из нескольких различных областей, таких как общественный транспорт, экологические ресурсы, спутниковые снимки и многое другое. Также имеется поисковая строка, которая поможет пользователям найти нужный набор данных, а также описания и примеры всех наборов данных, которые информативны и просты в использовании!

UCI Machine Learning Repository: большой репозиторий Школы информации и компьютерных наук Калифорнийского университета, содержащий более 100 наборов данных. Пользователи могут найти одномерные и многомерные наборы данных временных рядов, наборы данных для

классификации, регрессии или рекомендательных систем.

Google Dataset Search Engine: это набор инструментов, позволяющий искать наборы данных по имени.

Microsoft Datasets: в июле 2018 года компания Microsoft совместно с внешним исследовательским сообществом объявила о запуске "Microsoft Research Open Data". Он содержит хранилище данных в облаке для облегчения сотрудничества между мировым исследовательским сообществом. Он предоставляет коллекцию обработанных наборов данных для опубликованных исследований.

Awesome Public Datasets Collection: коллекция наборов данных на Github, организованная по "темам", таким как биология, экономика, образование и др. Большинство наборов данных бесплатны, но пользователям необходимо проверить лицензионные требования перед использованием любого набора данных.

Наборы данных компьютерного зрения: Visual Data содержит ряд больших наборов данных, которые могут быть использованы для построения моделей компьютерного зрения (CV). Пользователи могут искать конкретные наборы данных по определенным темам CV, таким как семантическая сегментация, создание подписей к изображениям, генерация изображений, или даже по решениям (набор данных по самоуправляемым автомобилям).

Некоторые из широко используемых наборов данных изображений.

Mnist: набор данных рукописного цифрового текста, содержащий 60 000 обучающих и 10 000 тестовых наборов. (Однако этот набор данных обычно используется только в качестве простой демонстрации, если вы хотите проверить производительность модели алгоритма, лучше провести тестирование на большем наборе данных, чтобы иметь достаточную уверенность в результатах эксперимента).

Cifar: Существуют Cifar10 и Cifar100. Первый содержит 60 000 изображений в 10 категориях по 6000 изображений в каждой. Последняя - это 100 категорий с 600 изображениями в каждой категории. Категории включают животных, таких как кошки, собаки, птицы, и транспортные средства, такие как самолеты, автомобили и лодки.

ImageNet: предположительно самый большой набор данных изображений с открытым исходным кодом, содержащий 15 миллионов изображений и 22 000 категорий.

LFW: набор данных лиц, содержащий 13 000+ изображений и 1680 различных людей.

CelebA: набор данных лиц, содержащий около 20w изображений, в общей сложности 10177 различных людей, и 5 помеченных точек для каждого изображения, 40 атрибутивной информации.

2.2 Использование веб-краулеров для сбора данных

Принципы реализации различных типов веб-краулеров отличаются, но в

этих принципах реализации будет много общего. Здесь я возьму два типичных типа веб-краулеров (т.е. общие веб-краулеры и специализированные веб-краулеры) в качестве примеров, чтобы объяснить принципы реализации веб-краулеров.

2.2.1 общий веб-краулер

Давайте сначала рассмотрим принцип реализации универсального веб-краулера. Принципы и процесс реализации универсального веб-краулера можно кратко описать следующим образом.

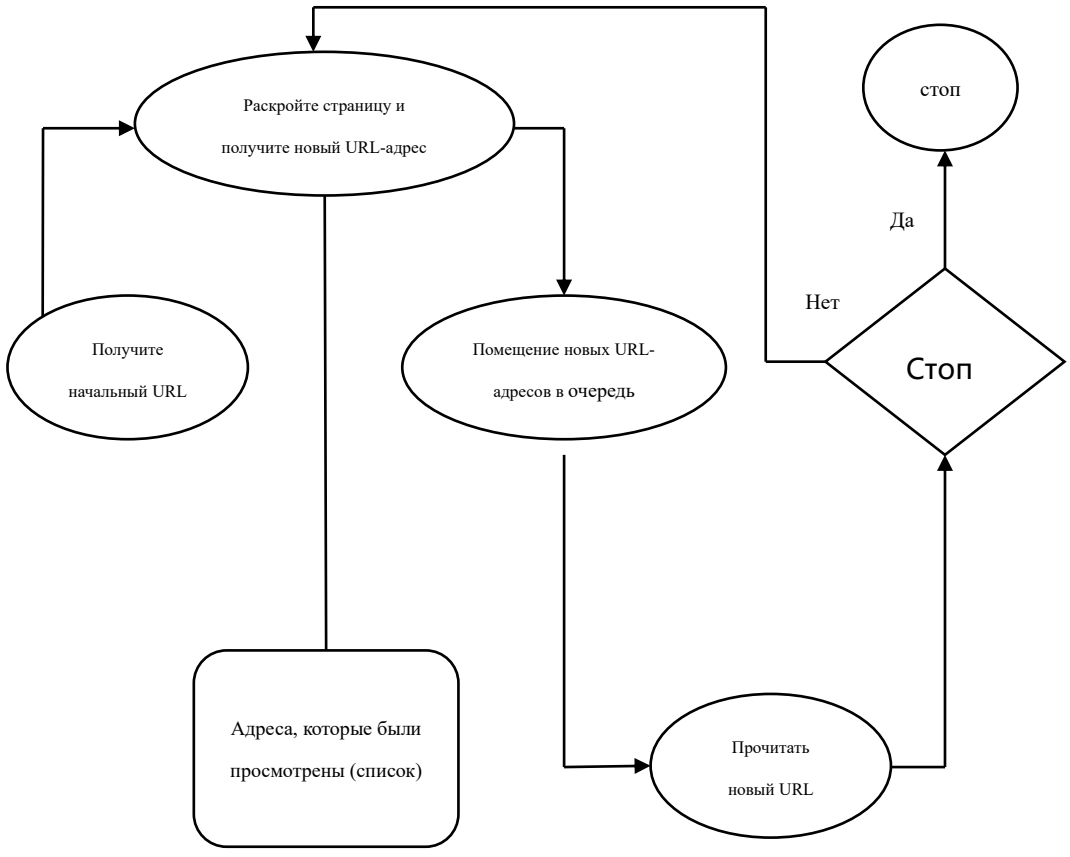


Рис.4

Следуя приведенной выше схеме, я подробно остановлюсь на точных шагах каждого этапа:

(1) Получение начального URL-адреса. начальный URL-адрес может быть указан пользователем искусственно, или он может быть определен по одной или нескольким начальным просмотренным страницам, указанным пользователем.

(2) Переползание страниц на основе исходного URL и получение новых URL. после получения исходного адреса URL сначала необходимо переползти веб-страницы по соответствующему адресу URL, и после переползания веб-страниц по соответствующему адресу URL веб-страницы сохраняются в исходной базе данных, а во время переползания веб-страниц обнаруживаются новые адреса URL, и переполненные адреса URL сохраняются в списке URL. используется для дедупликации и для определения процесса ползания.

(3) Поместите новые URL-адреса в очередь URL-адресов. После получения очередного нового адреса URL на шаге 2, новый адрес URL помещается в очередь URL.

(4) Новый URL считывается из очереди URL, и веб-страница просматривается на основе нового URL, новый URL получается из новой веб-страницы, и вышеупомянутый процесс просмотра повторяется.

(5) Остановите ползание, когда будет выполнено условие остановки, установленное системой ползания. При написании краулера обычно задаются соответствующие условия остановки. Если условие остановки не задано, краулер будет продолжать ползать до тех пор, пока не сможет получить новый адрес URL, а если задано условие остановки, краулер прекратит ползать, когда условие остановки будет выполнено.

2.2.2 Специализированные веб-краулеры

Специализированные веб-краулеры, поскольку им необходимо выполнять поиск с определенной целью, должны добавить механизмы определения цели и фильтрации для веб-краулеров общего назначения, в частности, на данном этапе их принципы выполнения и процессы требуют на три этапа больше, чем веб-краулеры общего назначения, т.е. определение цели, фильтрация нерелевантных ссылок, выбор адреса URL для следующего просмотра и т.д.

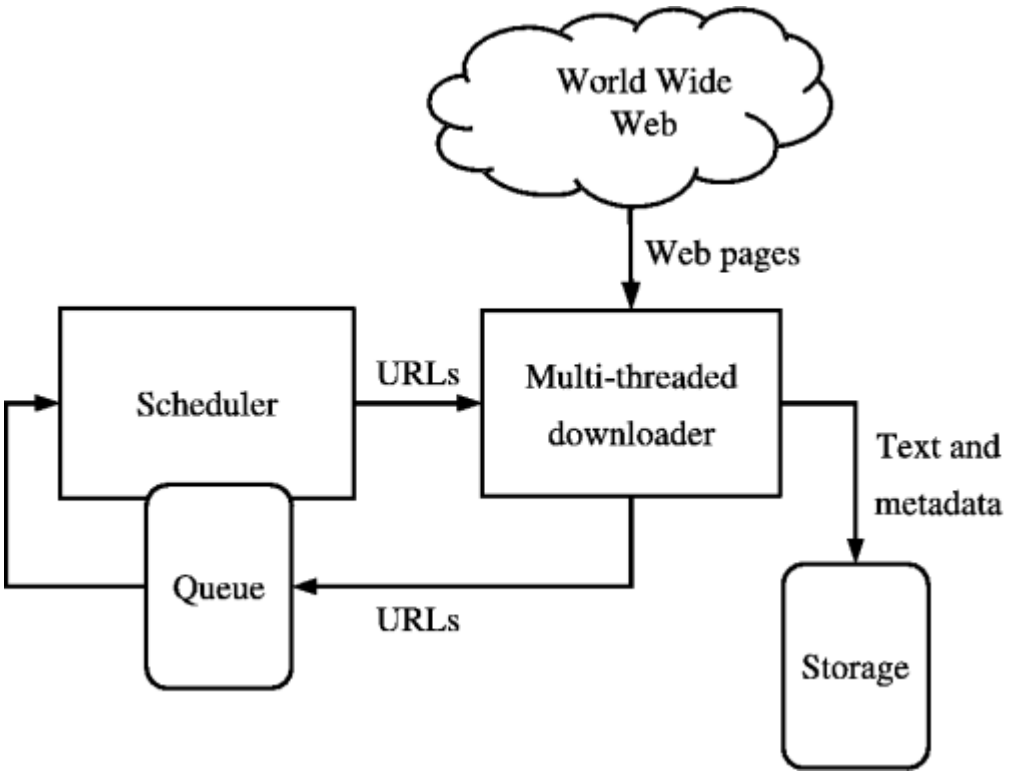


Рис.5

Эта блок-схема выглядит сложной, поэтому позвольте мне объяснить точные шаги.

(1) Определение и описание цели ползания. В сфокусированном веб-краулере мы сначала определяем цель ползания этого сфокусированного веб-краулера на основе требований к ползанию, а также выполняем соответствующее описание.

(2) Получите исходный URL-адрес.

(3) Перейдите на страницу в соответствии с исходным URL и получите новый URL.

(4) Фильтрация ссылок с нового URL, которые не относятся к цели ползания. Поскольку сфокусированный веб-краулер целенаправленно просматривает веб-страницы, страницы, не имеющие отношения к цели, будут отфильтрованы. Также необходимо хранить адреса URL, которые были просмотрены, в списке URL, который используется для дедупликации и определения процесса просмотра.

(5) Поместите отфильтрованные ссылки в очередь URL.

(6) Из очереди URL-адресов определите приоритет URL-адресов и определите адреса URL-адресов, которые будут просмотрены следующими, на основе алгоритма поиска. Для универсального веб-краулера менее важно, какие адреса URL следует просматривать дальше, но для целенаправленного веб-краулера относительно важно, какие адреса URL следует просматривать дальше, поскольку он имеет целенаправленный характер. Для целенаправленных веб-краулеров различный порядок обхода может привести к различной эффективности выполнения, поэтому нам необходимо определить, какие адреса

URL должны быть обследованы следующими, основываясь на стратегии поиска.

(7) Из URL-адреса, который должен быть просмотрен следующим, считывается новый URL-адрес, затем веб-страница просматривается на основе нового URL-адреса, и вышеописанный процесс просмотра повторяется.

(8) Когда условие остановки, установленное в системе, выполнено, или когда новый адрес URL не может быть получен, ползание останавливается.

2.2.3 Стратегия ползания

В процессе ползания веб-краулера в списке URL-адресов, подлежащих ползанию, может быть много URL-адресов, поэтому какой из этих URL-адресов краулер должен ползать первым, а какой - потом?

В общих веб-краулерах, хотя порядок ползания не так важен, во многих других краулерах, таких как ориентированные веб-краулеры, порядок ползания очень важен, и порядок ползания, в общем, определяется стратегией ползания.

Основными стратегиями ползания являются стратегия ползания в глубину, стратегия ползания в ширину, стратегия "первый большой сайт", стратегия обратного связывания, другие стратегии ползания и т.д.

Предположим, есть веб-сайт, ABCDEFG - это страницы сайта, а стрелки на рисунке указывают на иерархию страниц.

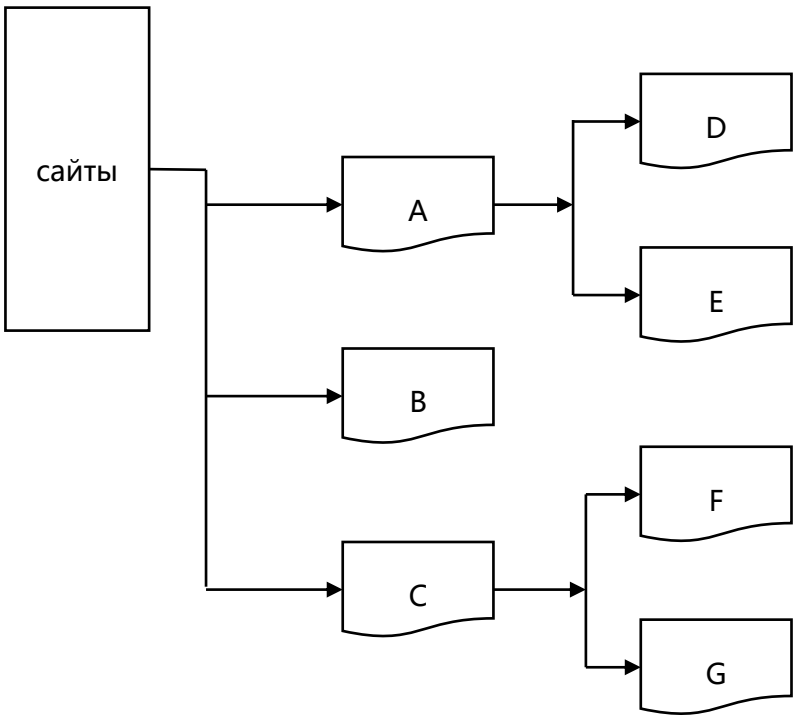


Рис.6

Если в это время веб-страница ABCDEFG находится в очереди на обползание, то в соответствии с различными стратегиями обползания, порядок

обползания будет разным.

Например, если придерживаться стратегии поиска по глубине, то сначала будет просмотрена веб-страница, а затем нижние ссылки этой веб-страницы будут поочередно просмотрены по глубине, прежде чем вернуться на верхний уровень для просмотра.

Поэтому, если следовать стратегии поиска по глубине, порядок поиска на рисунке 3-3 может быть следующим: A → D → E → B → C → F → G.

Если мы выполняем ползание в соответствии со стратегией ползания "по ширине", то сначала мы будем ползать по веб-страницам одного уровня, а после того, как мы проползем все веб-страницы одного уровня, мы выберем следующий уровень веб-страниц для ползания, например, на приведенном выше веб-сайте, если мы выполняем ползание в соответствии со стратегией ползания "по ширине", порядок ползания может быть следующим: A → B → C → D → E → F → G.

В дополнение к двум вышеупомянутым стратегиям ползания, мы также можем использовать стратегию ползания по крупным станциям. Мы можем классифицировать соответствующие веб-страницы в соответствии с сайтами, к которым они относятся, и если сайт имеет большое количество веб-страниц, то мы называем его большим сайтом, и в соответствии с этой стратегией, чем больше количество веб-страниц, тем больше сайт, и тогда приоритетным является переползание URL-адреса веб-страницы на большом сайте.

Количество обратных ссылок на веб-страницу означает число ссылок на нее со стороны других веб-страниц, и это число в определенной степени отражает количество ссылок на веб-страницу со стороны других веб-страниц. Таким образом, если ползание осуществляется в соответствии со стратегией обратных ссылок, то какая веб-страница имеет большее количество обратных ссылок, та и будет ползаться первой.

Однако на самом деле, если просто руководствоваться стратегией обратных ссылок для определения приоритета веб-страницы, то здесь может быть много обмана. Например, создать несколько групп помоек и связать эти сайты между собой, если так, то каждый сайт получит более высокую обратную ссылку, тем самым достигнув цели обмана.

Как участники проекта краулера, мы, конечно, не хотим, чтобы нас беспокоило такое мошенническое поведение, поэтому обычно учитывается надежное количество обратных ссылок, если для краулинга используется стратегия обратных ссылок.

Помимо перечисленных выше стратегий краулинга, на практике существует множество других стратегий краулинга, таких как стратегия OPIC, стратегия частичного PageRank и т.д.

2.2.4 Стратегия обновления веб-сайтов

Веб-страницы сайта часто обновляются, как сторона краулера, после

обновления веб-страницы, мы должны повторно ползать эти страницы, так когда ползать соответствующие? Если сайт обновляется слишком медленно, а краулеры ползают слишком часто, это, безусловно, увеличит нагрузку на краулеры и веб-серверы; если сайт обновляется быстрее, но краулеры ползают с большими интервалами, то версия контента будет слишком старой, что не способствует ползанию нового контента.

Очевидно, что чем ближе частота обновления сайта к частоте посещения сайта краулерами, тем лучше результат, конечно, когда ресурсы сервера краулера ограничены, в это время краулер также должен сделать различные веб-страницы с различными приоритетами обновления согласно соответствующей стратегии, с высоким приоритетом обновления веб-страниц, получит более быстрый отклик краулера.

В частности, существуют три общие стратегии обновления веб-страниц: стратегия пользовательского опыта, стратегия исторических данных, стратегия кластерного анализа и т.д.

Когда поисковая система запрашивает ключевое слово, появляется результат ранжирования. В результате ранжирования, как правило, имеется большое количество веб-страниц, однако большинство пользователей обращают внимание только на веб-страницы, находящиеся в верхнем рейтинге, поэтому в условиях ограниченных ресурсов сервера краулер будет отдавать приоритет обновлению веб-страниц, находящихся в верхнем рейтинге.

Эта стратегия обновления называется стратегией пользовательского опыта, поэтому в этой стратегии, когда краулеры на самом деле ползают по страницам, занимающим верхние строчки рейтинга? В этот момент краулер сохраняет несколько исторических версий соответствующих веб-страниц и проводит соответствующий анализ для определения цикла краулинга этих веб-страниц на основе информации об обновлении контента, влиянии на качество поиска и пользовательском опыте этих нескольких исторических версий.

Кроме того, мы также можем использовать стратегию исторических данных для определения цикла ползания обновлений веб-страниц. Например, мы можем основываться на исторических данных обновления конкретной веб-страницы и моделировать их с помощью процесса Пуассона и т.д., чтобы предсказать время следующего обновления этой веб-страницы, чтобы определить время следующего переползания этой веб-страницы, т.е. определить цикл обновления.

Обе вышеперечисленные стратегии требуют исторических данных в качестве основы. Иногда, если веб-страница является новой, соответствующих исторических данных не существует, и если анализ основан на исторических данных, сервер краулера должен сохранить информацию об исторической версии соответствующей веб-страницы, что, несомненно, увеличивает нагрузку на сервер краулера.

Если мы хотим решить эти проблемы, то нам необходимо принять новые стратегии обновления. Чаще всего используется стратегия кластерного анализа. Что же представляет собой стратегия кластерного анализа?

В жизни, я думаю, мы уже очень хорошо знакомы с классификацией,

например, мы идем в торговый центр, товары в торговом центре обычно разделены на категории, чтобы покупатели могли легко пойти и купить соответствующий товар, в это время категории классификации товаров фиксированы, уже сформулированы.

Однако, если количество товаров огромно, их невозможно классифицировать заранее, или мы просто не знаем, какие категории товаров у нас будут, как в это время решить проблему категоризации товаров?

В это время, мы можем использовать способ кластеризации для решения проблемы, на основе общности между товарами для анализа соответствующих, более общих товаров в класс, в это время, количество товаров, собранных в класс не является определенным, но может гарантировать, что товары вместе должны иметь некоторую общность, то есть, на основе идеи "вещи собраны по классам", чтобы достичь.

Позвольте мне обобщить все вышесказанное в несколько конкретных шагов:

(1) Во-первых, после обширных исследований было установлено, что веб-страницы могут иметь различное содержание, но в целом веб-страницы с похожими атрибутами обновляются с одинаковой частотой. Это обязательное условие для применения алгоритма кластерного анализа при обновлении веб-страниц краулером.

(2) Используя идею (1), мы можем сначала провести кластерный анализ большого количества веб-страниц, и после кластеризации будет сформировано несколько классов, а веб-страницы в каждом классе имеют схожие атрибуты, т.е., как правило, имеют схожую частоту обновления.

(3) После завершения кластеризации мы можем сделать выборку веб-страниц в одном кластере, а затем найти среднее значение обновления результата выборки для определения частоты ползания для каждого кластера.

Выше, является использование краулеров ползать в Интернете, общие 3 стратегии обновления, мы освоили его алгоритм мысли, в последующем мы осуществляем фактическое развитие краулера, выписать краулер выполнения эффективность будет выше, и логика выполнения будет более разумным.

Глава 3. Правила очистки данных

Очистка данных в основном основана на некоторых выводах, полученных после разведочного анализа, а затем обрабатываются четыре типа аномальных данных; недостающее значение, выбросы, дублирующиеся данные и обработка шумовых данных.

3.1 Разведочный анализ

Исследовательская часть анализа, для всех данных, заключается в получении предварительного понимания данных и процесса исследовательского анализа априорных знаний. В моем процессе добычи соответствующих данных,

я в основном использую связанные с python библиотеки научных вычислений для предварительного исследования данных, таких как типы данных, недостающие значения, размер набора данных, распределение данных по каждому признаку и т.д., и использую сторонние библиотеки отображения для интуитивное наблюдение для получения основных свойств и распределения данных, кроме того, одномерный анализ и многомерный анализ позволяют предварительно изучить взаимосвязь между признаками в наборе данных для подтверждения гипотез, выдвинутых на этапе бизнес-анализа.

3.2 Отсутствующие значения

Недостающие значения в наборе данных можно получить непосредственно с помощью различных методов, поставляемых с pandas. Недостающие значения часто встречаются в большинстве наборов данных, поэтому обработка недостающих значений будет напрямую влиять на конечные результаты модели. Способ работы с отсутствующими значениями в основном зависит от важности атрибута, в котором находятся отсутствующие значения, и от распределения отсутствующих значений.

Например, если данные распределены равномерно, используйте среднее значение для заполнения данных; если распределение данных перекошено, используйте медиану для заполнения данных. Если атрибут является атрибутом категории, он может быть заполнен глобальной константой 'Unknow', однако, как правило, это работает плохо, так как алгоритм может распознать его как совершенно новую категорию, поэтому он редко используется.

2. если процент пропусков высок (>95%), а важность атрибута низкая, достаточно просто удалить атрибут. Однако, когда недостающее значение велико и степень атрибута высока, прямое удаление атрибута может очень плохо сказаться на результатах работы алгоритма.

3. большое количество отсутствующих значений и высокая важность атрибутов: Основные используемые методы - интерполяция и моделирование.

(1) Методы интерполяции в основном включают случайную интерполяцию, множественную интерполяцию, интерполяцию тепловой платформы, интерполяцию Лагранжа и интерполяцию Ньютона.

(2) Метод случайной интерполяции - из общей совокупности случайным образом отбирается определенное количество образцов вместо недостающих образцов

(3) Метод множественной интерполяции - недостающие данные прогнозируются с помощью взаимосвязи между переменными, генерируется несколько полных наборов данных с использованием методов Монте-Карло, эти наборы данных анализируются, и, наконец, результаты анализа агрегируются

(4) Интерполяция на горячей платформе -- означает поиск выборки, аналогичной выборке, в которой находится недостающее значение (согласованная выборка), в наборе данных без пропусков и использование наблюдений в ней для интерполяции недостающего значения.

Плюсы: простота и легкость выполнения, высокая точность

Недостатки: когда число переменных велико, обычно трудно найти выборку, которая точно совпадает с выборкой, подлежащей интерполяции. Однако мы можем стратифицировать данные по определенным переменным и интерполировать недостающие значения в страте с помощью практических средств

4. метод разностей Лагранжа и метод интерполяции Ньютона

Далее описываются методы моделирования.

Недостающие данные можно предсказать с помощью таких моделей, как регрессия, Байесовская, случайный лес, дерево решений и т.д. Например, используя атрибуты других данных в наборе данных, можно построить дерево решений для прогнозирования значения отсутствующих значений.

В целом, не существует единого процесса обработки отсутствующих значений данных, и метод должен быть выбран на основе фактического распределения данных, степени перекоса, процента отсутствующих значений и т.д. В моем процессе предварительной обработки данных, в дополнение к использованию простого метода заполнения снаружи с удалением, чаще использовать метод моделирования для заполнения, в основном потому, что метод моделирования для прогнозирования неизвестного значения на основе существующего значения, коэффициент точности выше. Однако метод моделирования может также привести к увеличению корреляции ме

3.3 Аномальные данные

Существует множество методов определения выбросов на основе статистического контекста, отличных от визуального анализа (общие графики линий ящиков), и визуальное наблюдение не подходит для ситуаций с большим количеством данных.

3.1 Простой статистический анализ

Этот шаг выполняется в EDA и может быть достигнут простым использованием метода description в pandas для обнаружения наличия необоснованных значений, т.е. выбросов, с помощью описательной статистики набора данных

3.2 3∂ Принцип - обнаружение выбросов на основе нормального распределения

Если данные подчиняются нормальному распределению, согласно принципу 3∂, выброс - это значение в наборе измеренных значений, которое отклоняется от среднего более чем в 3 раза от стандартного отклонения. Если данные подчиняются нормальному распределению, то вероятность появления значения за пределами 3∂ от среднего составляет P(|x-u| > 3∂) <= 0,003, что является очень индивидуальным маловероятным событием. Если данные не подчиняются нормальному распределению, их также можно описать тем, во сколько раз стандартное отклонение от среднего.

3.3 Обнаружение на основе модели

Сначала постройте модель данных, и аномалии - это те объекты, которые не соответствуют модели идеально; если модель представляет собой набор кластеров, аномалии - это объекты, которые не принадлежат в значительной степени ни к одному кластеру; при использовании регрессионной модели аномалии - это объекты, которые относительно далеки от предсказанных значений

3.4 Основанные на расстоянии

Определяя меры близости между объектами, аномальные объекты - это те, которые находятся далеко от других объектов

Преимущества: простота и удобство в использовании

Недостатки: временная сложность O(m^2), не подходит для ситуаций с большими наборами данных, более чувствительный выбор параметров, не может работать с наборами данных с различными областями плотности, поскольку использует глобальные пороги и не может учесть такие вариации плотности

3.5 На основе плотности

Классифицирует точку как выброс только в том случае, если ее локальная плотность значительно ниже, чем у большинства ее ближайших соседей. Подходит для неравномерно распределенных данных.

Плюсы: дает количественную оценку того, когда объект является выбросом, и хорошо работает, даже если данные имеют различные области

Недостатки: временная сложность O(m^2); сложный выбор параметров, хотя алгоритм решает проблему, рассматривая различные значения k и получая максимальную оценку выбросов, ему все равно необходимо выбрать верхнюю и нижнюю границы для этих значений.

3.6 Основанные на кластеризации

Кластерный выброс: объект является кластерным выбросом, если он не принадлежит ни к одному кластеру. Влияние выбросов на первоначальную кластеризацию: если выбросы обнаруживаются при кластеризации, возникает проблема: является ли структура достоверной или нет, поскольку выбросы влияют на кластеризацию. Чтобы справиться с этой проблемой, можно использовать следующий метод: объект кластеризуется, удаляется выброс, и объект снова кластеризуется.

Преимущества:

(1) Методы кластеризации на основе линейной и почти линейной сложности (k-means) для обнаружения провалов могут быть весьма обоснованными Определение кластеров обычно дополняет провалы, поэтому можно обнаружить как кластеры, так и провалы

Недостатки:

(1) полученный набор выбросов и их оценки могут сильно зависеть от количества используемых кластеров и наличия выбросов в данных

(2) Качество кластеров, производимых алгоритмом кластеризации, очень сильно влияет на качество выбросов, производимых алгоритмом

Методы работы с выбросами.

1. удалите выбросы ---- очевидно, что они являются аномальными, и их количество невелико, могут быть удалены непосредственно

2. не иметь дело с - если алгоритм не чувствителен к промахам, то он не может иметь дело с ними, но если алгоритм чувствителен к промахам, то лучше не использовать этот метод, например, некоторые алгоритмы, основанные на вычислении расстояния, включая kmeans, knn и так далее.

3. средняя замена ---- теряет мало информации и является простой и эффективной.

4. рассматриваются как отсутствующие значения ---- могут обрабатываться так же, как и отсутствующие значения

3.4 Дедупликация

Для определения дубликатов основной идеей является "сортировка и объединение": сначала записи в наборе данных сортируются по определенным правилам, а затем определяется, являются ли записи дубликатами, путем сравнения сходства соседних записей. На самом деле это две операции, одна из которых - сортировка, а другая - вычисление сходства. В настоящее время мы в основном используем метод дублирования для вынесения суждений, а затем дублирующие образцы просто удаляются.

3.5 Обработка шумов (данных о помехах)

Шум - это случайная ошибка или дисперсия измеряемой величины, которая в основном отличается от выбросов. По формуле: Измерение = Истинные данные + Шум. Выбросы - это наблюдения, которые могут быть либо порождены истинными данными, либо шумом, но в целом это наблюдения, которые значительно отличаются от большинства наблюдений. Шум включает ошибочные значения или отдельные значения точек, которые отклоняются от ожидаемых, но нельзя сказать, что зашумленные точки содержат промахи, хотя большинство методов поиска данных отбрасывают промахи как шум или аномалии. Однако в некоторых приложениях (например, при обнаружении мошенничества) анализ выбросов или поиск аномалий проводится для точек выбросов. И некоторые точки являются отклонениями локально, но нормальными глобально.

Обработка шума в основном осуществляется методом split-box и методом регрессии.

1.Метод раздельных коробок.

Подход split-box сглаживает упорядоченные значения данных, рассматривая

"ближайших соседей" данных. Эти упорядоченные значения распределяются по нескольким "ведрам" или бинам. Поскольку метод split-box рассматривает значения ближайших соседей, он выполняет локальное сглаживание.

1.1 Сглаживание с помощью ящичных средних: каждое значение в ящике заменяется средним значением ящика.

1.2 Сглаживание с помощью медианы ящика: каждое значение в ящике заменяется медианой ящика.

1.3 Сглаживание с границами ящика: максимальное и минимальное значения в ящике одинаково считаются границами. Каждое значение в ячейке заменяется ближайшим граничным значением.

В целом, чем больше ширина, тем эффективнее сглаживание. Ячейки также могут быть одинаковой ширины, когда интервальный диапазон каждого значения ячейки является константой. В качестве метода дискретизации можно также использовать разделительные ящики.

2. метод регрессии

Для сглаживания данных можно использовать функцию. Линейная регрессия предполагает нахождение "лучшей" линии, соответствующей двум признакам (или переменным), чтобы один из них мог предсказать другой. Множественная линейная регрессия - это расширение линейной регрессии, которая включает более двух признаков и подгоняет данные к многомерной поверхности. Использование регрессии для нахождения математического уравнения, которое соответствует данным, может помочь устранить шум.

Глава 4. Инструменты и принципы разделения слов

В приведенном выше разделе мы кратко опишем необходимость сегментации слов и предложим еще несколько полезных инструментов сегментации слов, здесь я продолжу объяснять различные инструменты сегментации слов.

4.1 Инструмент для разделения слов на английском (русском) языке –NLTK

NLTK - это ведущая платформа для создания программ на языке Python для работы с данными о человеческом языке. Она предоставляет простые в использовании интерфейсы к более чем 50 корпорациям и лексическим ресурсам, таким как WordNet, а также набор библиотек обработки текста для классификации, токенизации, стемминга, тегирования, синтаксического анализа и семантических рассуждений, обертки для промышленных библиотек NLP и активный дискуссионный форум.

Благодаря практическому руководству, знакомящему с основами

программирования наряду с темами вычислительной лингвистики, а также исчерпывающей документации по API, NLTK подходит как для лингвистов, инженеров, студентов, преподавателей, исследователей, так и для промышленных пользователей. NLTK доступен для Windows, Mac OS X и Linux. Самое главное, что NLTK - это бесплатный проект с открытым исходным кодом, управляемый сообществом.

NLTK называют "прекрасным инструментом для обучения и работы в области вычислительной лингвистики с использованием Python" и "удивительной библиотекой для игры с естественным языком"[9].

Поскольку NLTK является инструментарием обработки естественного языка общего назначения, нам не нужно использовать все его возможности, нам просто нужно использовать функцию "word_tokenize" из "nltk.tokenize" для преобразования предложений в слова.

Если вы хотите использовать NLTK как инструмент для разбиения слов, вам просто нужно установить "pip3 install NLTK" в python и вызвать этот API. использовать NLTK в python очень удобно, но нужно учитывать другой случай - проблему деактивированных слов.

Деактивация слов - это большая проблема, но есть шаги, которые может выполнить каждый. Основная идея, однако, заключается в том, что компьютеры просто не понимают слова напрямую. Если мы хотим подражать тому, как люди читают и понимают текст, нам нужен метод, максимально приближенный к этому. В целом, компьютеры используют числа для представления всего, но мы часто видим непосредственно в программировании использование двоичных сигналов (True или False, которые могут быть непосредственно преобразованы в 1 или 0, непосредственно из наличия (True, 1) или отсутствия (False, 0) электрических сигналов). Для этого нам нужен способ преобразования слов в числовые или сигнальные шаблоны. Процесс преобразования данных в то, что может понять компьютер, называется "предварительной обработкой". Одной из основных форм предварительной обработки является отфильтровывание бесполезных данных. В обработке естественного языка бесполезные слова (данные) называются стоп-словами.

Мы можем сразу понять, что некоторые слова более значимы, чем другие. Мы также видим, что некоторые слова бесполезны и являются словами-заполнителями. Например, в английском языке мы используем их для заполнения предложений, чтобы они не звучали так странно. Одним из самых распространенных, неофициальных примеров бесполезных слов является слово umm. Люди часто используют umm, чтобы заполнить больше, чем другие слова. Для большинства аналитиков эти слова бесполезны.

Мы не хотим, чтобы эти слова занимали место в нашей базе данных или отнимали ценное время обработки. Поэтому мы называем эти слова "деактивированными словами", так как они бесполезны, и мы хотим оставить их в покое. Или, если выразить это в более письменной форме: мы деактивируем слова, приведенные выше.

Например, если вы найдете слова, которые часто используются в сарказме,

вы можете немедленно прекратить их употребление. Саркастическое слово или фраза будет варьироваться от слова к слову и от корпуса к корпусу. На данный момент мы будем относиться к стоп-словам так, как будто они не содержат никакого смысла, и мы собираемся их удалить.

Вы можете легко реализовать его, сохранив список слов, которые вы считаете стоп-словами. NLTK использует кучу слов, которые они считают стоп-словами, и получить доступ к ним можно через корпус NLTK по адресу.

«from nltk.corpus import stopwords»

API деактиватора NLTK можно вызвать с помощью этого оператора python.

4.2 Инструмент для разделения китайских слов – jieba

jieba - компонент для разделения китайских слов, размещенный на GitHub, разработанный и поддерживаемый fxsjy, использующий двойные деревья Trie, динамическое программирование и модель HMM, и является самым эффективным компонентом для разделения китайских слов, разработанным с использованием Python.[10]

Поскольку я использую два набора данных, китайский и английский. Для китайского набора данных мне нужно использовать специальный инструмент для разделения китайских слов - jieba. Если ваша задача не на китайском языке, вы можете пропустить эту часть. Из-за разницы в языке, это просто разница в инструменте разделения, но следующие шаги в основном одинаковы. Далее я излагаю основные принципы использования jieba для разделения слов.

(1)Эффективное сканирование графа слов на основе структуры дерева тройки, создавая направленный ациклический граф (DAG) всех возможных случаев словообразования для китайских слов в предложении.jieba поставляется со словарем dict.txt, который содержит более 20 000 слов, включая количество вхождений слов (это количество получено в результате собственного обучения автора на основе корпуса китайских новостей и газет) и лексические свойства. В этой первой статье о структуре дерева тройки сканирования графа слов, говорится, что 20 000 слов, помещенных в дерево тройки, и дерево тройки является хорошо известным префиксным деревом, то есть слово с одинаковыми первыми несколькими словами, означает, что они имеют одинаковый префикс, вы можете использовать дерево тройки для хранения, с преимуществом быстрой скорости поиска.

Направленный ациклический граф DAG - это направленный ациклический граф всех возможных случаев словообразования в предложении, порожденный после предложения, то есть, задав предложение, вы хотите разделить слово, то есть, задав предложение, которое нужно разделить, породите направленный ациклический граф этого предложения.

Например: {0:[1,2,3]} является простой DAG, что означает, что символы между начальными позициями 0~1, 0~2, 0~3 являются словами в dict.txt.

(2)Динамическое программирование используется для нахождения пути с максимальной вероятностью, для нахождения максимальной комбинации

разрезов на основе частоты слов

В коде jieba говорится, что словарь преобразует количество вхождений каждого слова в частоту при генерации дерева. Объяснить про частоту и вероятность: по определению, частота также является десятичным числом между 0 и 1, и представляет собой число случаев наступления события/общее число опытов, поэтому частота приблизительно равна вероятности, если число опытов достаточно велико, или предел частоты равен вероятности. Однако его часто путают с частотой, часто приравнивая частоту к количеству повторений события, как в данном случае количество повторений слова.

В динамическом программировании мы сначала находим слово, которое было нарезано и порезано кубиками в делимом предложении, и находим частоту этого слова (число/общее количество), если такого слова нет (а оно должно быть, так как основано на словаре), мы берем частоту слова с наименьшей частотой в словаре в качестве частоты этого слова, что означает P(a word)=FREQ.get('a word',min_freq), затем в соответствии с методом динамического программирования для нахождения пути максимальной вероятности, предложение справа налево, чтобы обратить максимальную вероятность (в некоторых учебниках может быть слева направо, здесь обратное, потому что центр тяжести китайских предложений часто отстает, то есть справа, потому что обычно Поэтому при расчете справа налево правильная ставка выше, чем при расчете слева направо, что похоже на обратное максимальное соответствие), P(NodeN)=1.0, P(NodeN-1)=P(NodeN)*Max(P(первое слово отсчета))... и так далее, и в итоге получаем максимум путь вероятности, чтобы получить комбинацию баллов среза с максимальной вероятностью.

(3)Для незарегистрированных слов используется модель HMM, основанная на способности китайских иероглифов к словообразованию, и алгоритм Витерби.

Незарегистрированное слово - это слово, которое не записано в словаре dict.txt.

Китайские слова маркируются в соответствии с четырьмя состояниями BEMS, B - начальная позиция, E - конечная позиция, M - средняя позиция, S - певучая позиция, нет ни до, ни после. Другими словами, jieba использует четыре состояния (B, E, M, S) для обозначения китайских слов.

(4) основная таблица вероятностей jieba (вы можете увидеть эти три файла в исходном коде jieba на python)

prob_trans.py

Вероятности позиционного перехода, т.е. вероятности перехода для четырех состояний B (начало), M (середина), E (конец), S (независимое словообразование).

{'B': {'E': 0.8518218565181658, 'M': 0.14817814348183422},
'E': {'B': 0.5544853051164425, 'S': 0.44551469488355755},
'M': {'E': 0.7164487459986911, 'M': 0.2835512540013088},
'S': {'B': 0.48617017333894563, 'S': 0.5138298266610544}

P(E|B) = 0,851, P(M|B) = 0,149, что указывает на то, что когда мы находимся в начале слова, вероятность того, что следующее слово является окончанием

намного выше, чем вероятность того, что следующее слово является средним словом, в соответствии с нашей интуицией, поскольку слова из двух слов встречаются чаще, чем многословные слова.

prob_emit.py

Вероятность расположения к стрельбе одного слова, например, P("and"|M) указывает на вероятность того, что слово "and" встречается в середине слова.

prob_start.py

Это начальный вектор, который является начальным состоянием модели системы HMM

Фактически, переход между BEMS в некоторой степени похож на бинарную модель, которая представляет собой переход между двумя словами

Бинарная модель учитывает вероятность того, что за словом последует другое слово, и является одной из моделей N-грамм.

Вкратце о шагах разделения jieba:

1. загружаем словарь, генерируем триевое дерево

2. дано предложение, которое нужно разделить, использовать регулярные для получения последовательных китайских и английских символов, разрезать на список фраз, использовать DAG (словарь) и динамическое программирование для каждой фразы, получить путь максимальной вероятности, для тех слов в DAG, которые не найдены в словаре, объединить их в новый фрагмент фразы, и использовать модель HMM для разделения фразы.

3. использовать синтаксис yield в python для генерации генератора слов, возвращая слово за словом.

Глава 5. GloVe и Bret

Прежде чем говорить о GloVe, необходимо понять word2vec, поскольку алгоритм GloVe фактически можно рассматривать как оптимизированный алгоритм word2vec, а алгоритм GloVe реализован поверх алгоритма word2ve.

5.1 Объяснение принципа word2vec

Мы можем догадаться по контексту слов, например, я ем виноград и яблоки. Мы не можем знать, что и виноград, и яблоки - это фрукты, но по этим двум словам мы можем сказать, что и виноград, и яблоки - это еда. Если есть еще несколько примеров, например, Этот виноград такой сладкий, Яблоко очень сладкое, Есть немного винограда в фруктовой тарелке, Тарелка содержит много

яблок и т.д., мы можем грубо догадаться, что виноград и яблоки очень похожи. Таким образом, анализируя несколько предложений с близким контекстом, мы можем получить сходство этих двух слов, и чем больше предложений, тем точнее полученное сходство.

Word2Vec основан на предположении - дистрибутивном предположении - что слова со схожими контекстами также семантически схожи.

Word2vec имеет две модели: CBOW и Skip-Gram.

1) CBOW предсказывает w, когда известен контекст(w).
2) Skip-Gram предсказывает контекст(w), когда w известен.

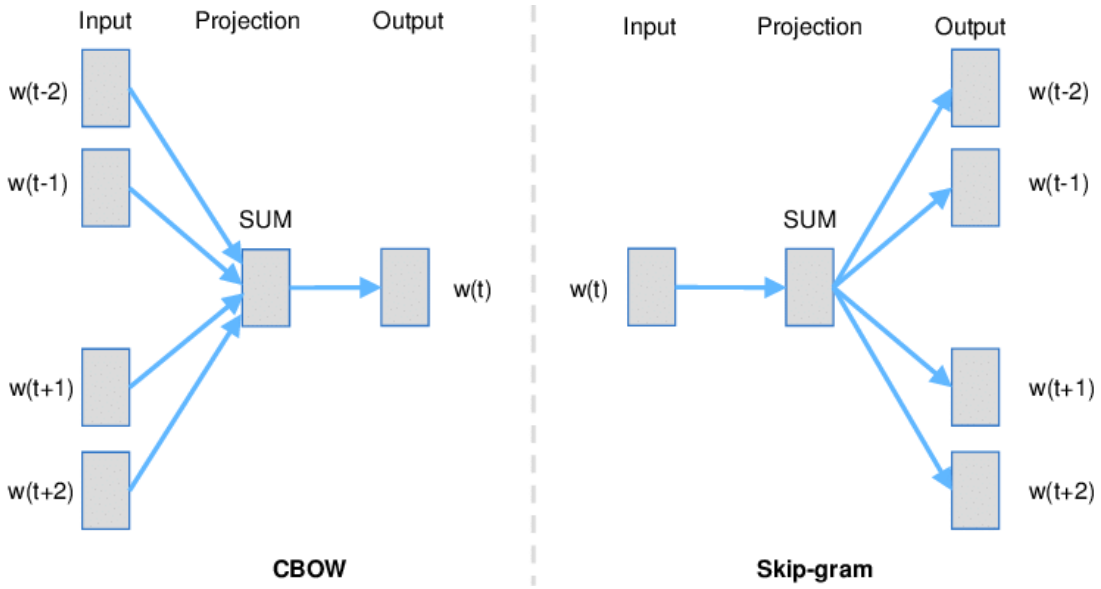


Рис.7

CBOW(Continuous Bag-of-Words)

Обычная модель CBOW предсказывает центральные слова по контексту, отбрасывая информацию о порядке слов; обратите внимание, что разные документы имеют разные описания, но суть одна и та же. Процесс обучения заключается в следующем.

1) One-Hoth-кодированный словарный вектор контекстных слов, V - количество слов в словаре и C - количество контекстных слов. "На столе есть яблоко" используется в качестве обучающих данных, здесь C=4, поэтому вход в модель является One-Hot-кодированным словесным вектором (is,an,on,the) 4 слов.
2) Инициализируем весовую матрицу Wv*n, затем слева умножаем эту матрицу на все входные One-Hot-кодированные словесные векторы, чтобы получить векторы w1,w2,w3,...wn размерности N. Здесь N задается самостоятельно в соответствии с потребностями задачи.
3) Полученные векторы w1,w2,w3,...wn суммируются и усредняются как вектор скрытого слоя h
4) Инициализируем другую весовую матрицу Wnm, слева умножаем Wnv

на вектор скрытого слоя h, а затем обрабатываем его функцией активации для получения V-мерного вектора y. Каждый элемент y представляет собой вероятностное распределение каждого соответствующего ему слова.

5) Слово, указанное элементом с наибольшей вероятностью в y, является прогнозируемым промежуточным словом (target word) по сравнению с One-Hot-кодированным словесным вектором истинной метки, а две весовые матрицы обновляются в соответствии с погрешностью.

Перед обучением необходимо определить функцию потерь (обычно это кросс-энтропийная функция стоимости) и обновить W и W', используя алгоритм градиентного спуска. После обучения один горячий вектор каждого отдельного слова во входном слое умножается на матрицу W, чтобы получить вектор слова, которое мы хотим представить, с помощью Распределенного Представления, также называемого встраиванием слова.

Примечание: Так как только один элемент One-Hot кодированного словарного вектора равен 1, а остальные 0, то i-ый словарный вектор, умноженный на матрицу W, является i-ой строкой матрицы, поэтому эта матрица также называется look up table, с помощью look up table можно обойтись без тренировочного процесса и напрямую искать вверх по таблице, чтобы получить словарный вектор слов.

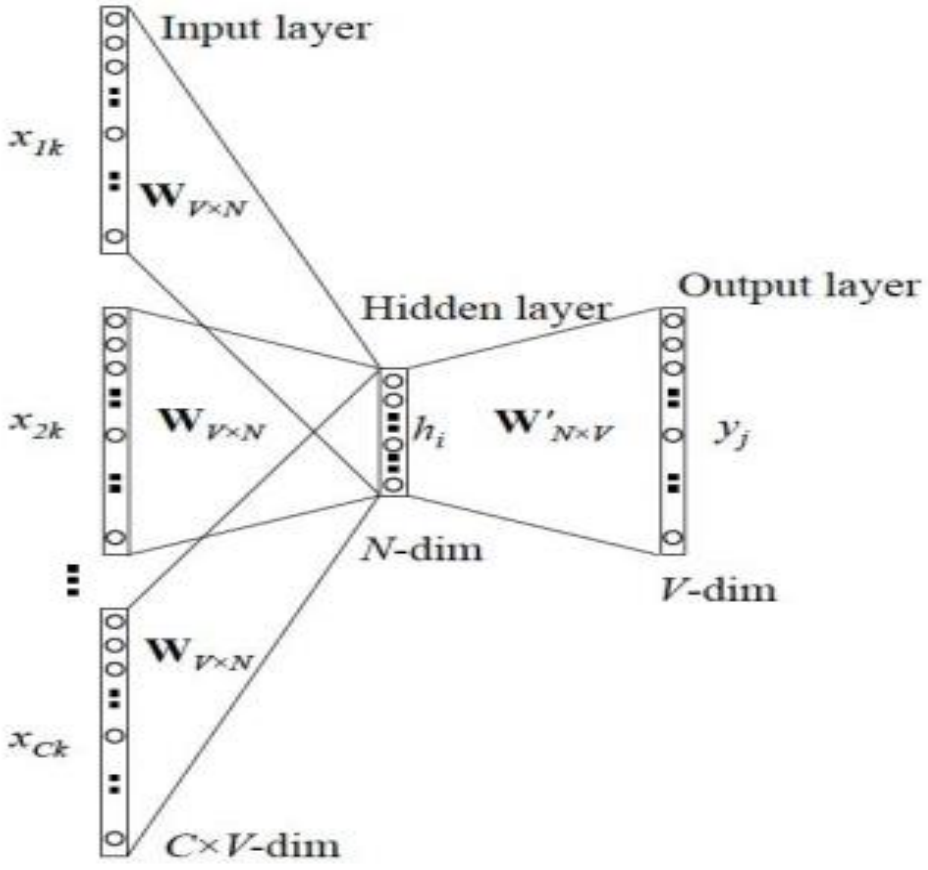


рис. 8

Skip-gram

Общий принцип метода Skip-gram заключается в том, что слово в контексте, в котором встречается слово, берется за выход, а само слово берется за вход, т.е., учитывая слово, можно надеяться, что оно будет предсказано в том контексте, в котором оно может появиться.

Весовая модель от входного до неявного слоя получается путем обучения на большом корпусе. Контекстуальные слова для слова " apple" ('there' , 'is' , 'an' , 'on','the','table') . Затем в качестве входного вектора слова "One-Hot" яблока выводится ('there' , 'is' , 'an' , 'on','the','table') вектора одного горячего слова. По окончании обучения вес каждого измерения каждого слова к неявному слою получается как вектор каждого слова (как в CBOW).

Сначала мы выбираем слово из середины предложения "На столе есть яблоко" в качестве входного слова, например, мы выбираем "яблоко" в качестве входного слова.

После того, как у нас есть входное слово, мы определяем параметр под названием skip_window, который представляет собой количество слов, которые мы выбираем вокруг текущего входного слова. Если мы установим значение skip_window=2, то словами, которыми мы заканчиваем в окне (включая входное слово) будут ['is','an','apple ', 'on', 'the']. skip_window=2 означает, что 2 слова слева от левого входного слова и 2 слова справа выбраны для входа в наше окно, так что все окно Другой параметр называется num_skips, который представляет собой количество различных слов, которые мы выбираем из всего окна в качестве нашего выходного слова. когда skip_window=2 и num_skips=2, мы получим два сета (входное слово, выходное слово) тренировочные данные формы ("apple", "an"), ("apple", "on").

Нейронная сеть получает статистику, основанную на количестве вхождений каждой пары слов в эти обучающие данные, и выводит вероятностное распределение, которое показывает, насколько вероятно, что каждое слово будет происходить одновременно с входным словом по нашей лексике.

Например, если в нейросетевую модель ввести слово "Китай", то вероятность конечного выхода модели будет значительно выше для родственных слов типа "Великобритания" и "Россия", чем для неродственных слов типа "яблоко" и "хлеб". будет намного выше, чем вероятность таких не относящихся к делу слов, как "яблоко" и "хлеб". Это связано с тем, что в окне "Китай" в тексте скорее всего появятся "Великобритания" и "Россия".

Наконец, как и в CBOW, матрица W обновляется по градиентному спуску и обратному распространению, а вектор строки в W - это Word,

встраивающее представление каждого слова.

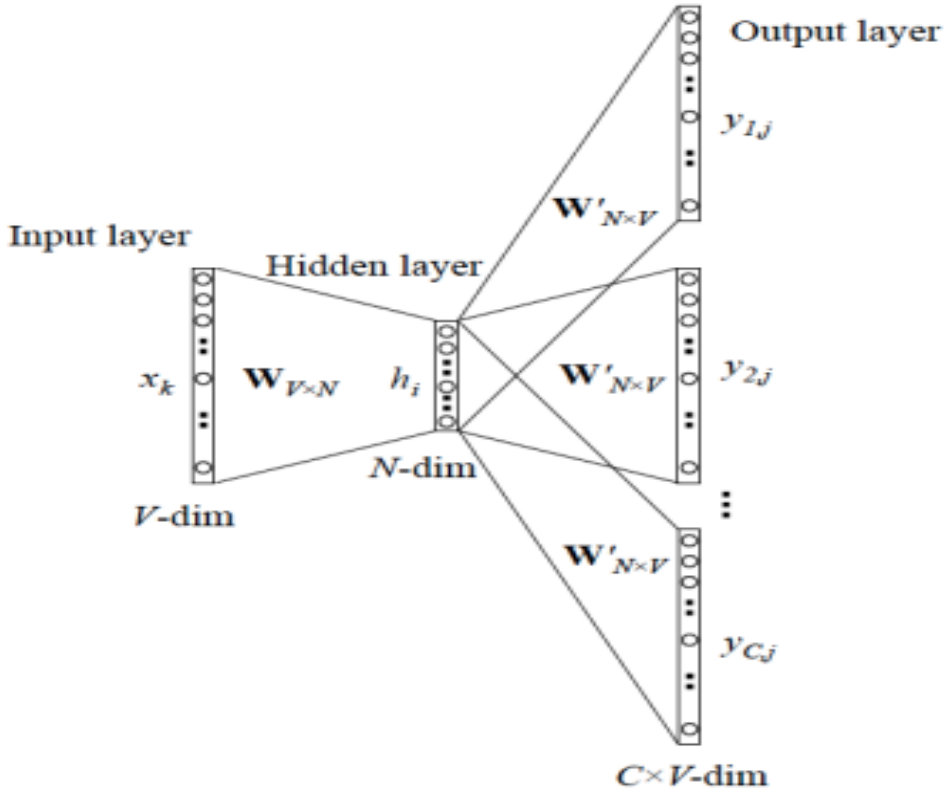


рис. 9

5.2 Объяснение принципа GloVe

GloVe - это инструмент для характеризации слов, основанный на глобальной статистике частоты слов.

Процесс создания GloVe:

1) Из корпуса строится матрица кооперации, и каждый элемент Xij матрицы представляет собой количество вхождений слова i и контекстного слова j вместе в контекстном окне определенного размера.
2) Построим приближенную связь между вектором слов и матрицей совпадений, целевая функция которой：

$$J = \sum_{i,j=1}^{V} f\left(X_{ij}\right)\left(w_i^T \tilde{w}_j + b_i + b_j - \log X_{ij}\right)^2$$

Основной формой этой функции потерь является простейший средний квадратный убыток, за исключением того, что к нему добавляется функция взвешивания f(xij) :

$$f(x) = \{ (x/x_{\max})^{\alpha} \quad \text{if } x < x_{\max} b1 \quad \text{otherwise.}$$

На основе экспериментов было установлено, что значение Xmax не сильно влияет на результаты, оригинальные авторы использовали Xmax=100, а результаты лучше для α=3/4, чем для α=1. Ниже представлено изображение функции f(x) для α=3/4, которое показывает, что для меньшего xij, веса также меньше. Изображение этой функции показано ниже:

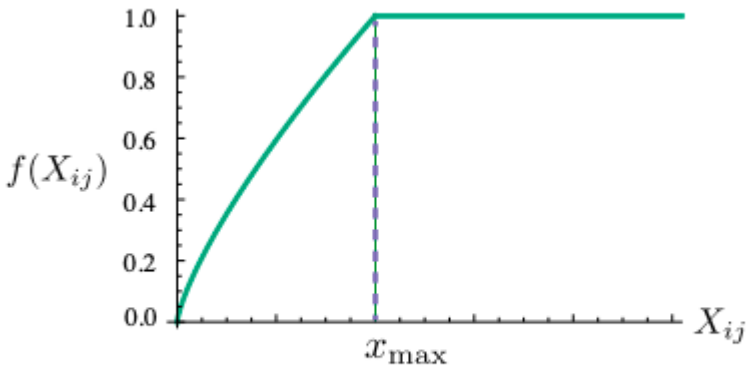


Figure 1: Weighting function f with $\alpha = 3/4$.

рис. 10

процесс обучения GloVe:

1) По существу, это все еще контролируемое обучение: хотя GloVe и не нужно вручную маркировать как неконтролируемое обучение, суть существования ярлыка заключается в log(xij).

2) Векторы ω и ω~ являются параметрами обучения и по сути аналогичны методу обучения под наблюдением, использующему алгоритм градиентного спуска АдаГрада, который случайным образом отсортировывает все ненулевые элементы матрицы X. Кривизна обучения установлена на 0,05, а для векторов размером менее 300 итераций - 50 раз, а для векторов других размеров - 100 раз. до конвергенции.

3) Окончательное изучение заключается в том, что двумя словарными векторами являются ω и ω~ , так как X симметричен, поэтому в принципе ω и ω~ , также симметричны, единственное различие между ними заключается в том, что инициализированные значения различны, а результирующие конечные значения не одинаковы. Таким образом, они фактически эквивалентны, и оба могут быть использованы в качестве конечного результата. Но для улучшения робастности в конечном итоге мы выбираем сумму двух ω + ω~ в качестве конечного вектора (разная инициализация двух эквивалентна добавлению разных случайных шумов, поэтому улучшает робастность).

5.3 Объяснение принципа Bert

Обратите внимание, что здесь мы говорим о модели встраивания слов в структуру Bert, а не о всей структуре Bert.

На входе Bert использует добавление трех вкраплений по сравнению с другими моделями, добавляя три вектора вкраплений Token, вкраплений Segment и вкраплений Position, с целью предварительного обучения и предсказания следующего слова(рис.11).

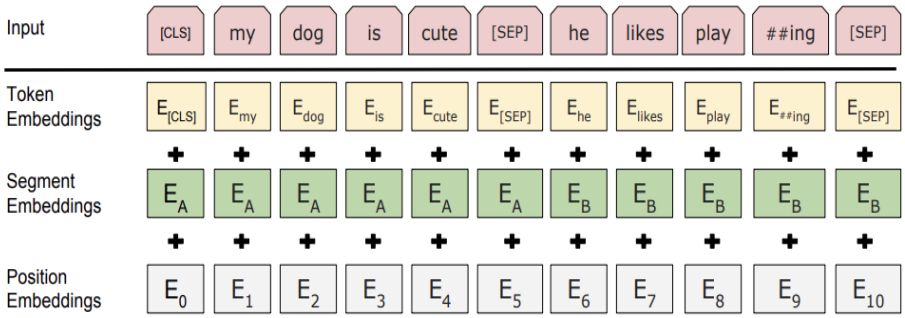


рис. 11

Как показано выше, входные данные Берта состоят из двух предложений: " my dog is cute" и " he likes playing". Во-первых, в начале первого предложения добавляется специальный маркер [CLS] для обозначения начала предложения и [SEP] для обозначения конца предложения.

Затем для каждого токена выполняется 3 вкрапления: вкрапление слова (Token Embeddings), вкрапление позиции (Position Embeddings) и вкрапление предложения (Segment Embeddings). Наконец, сумма трех вкраплений подается на следующий слой.

Вкрапления токенов

Преобразует каждое слово в одномерный вектор путем построения таблицы векторов слов в качестве входных данных модели. В частности, английские слова нарезаются с более мелкой зернистостью, например, play или cut на play и ##ing, в то время как китайский язык в настоящее время не нарезает входной текст, а непосредственно формирует настоящую входную единицу для одиночных слов. Разбиение слов на более мелкие части - это распространенный подход к решению проблемы незарегистрированных слов.

Предположим, вводится текст " I like dog". На следующем рисунке показана реализация слоя Token Embeddings. Перед подачей в слой Token Embeddings входной текст подвергается токенизации, и два специальных токена вставляются в начало [CLS] и конец [SEP] текста.

Слой Token Embeddings преобразует каждое слово в 768-мерный вектор,

в примере 5 Tokens преобразуются в (6, 768) матрицу или (1, 6, 768) тензор.

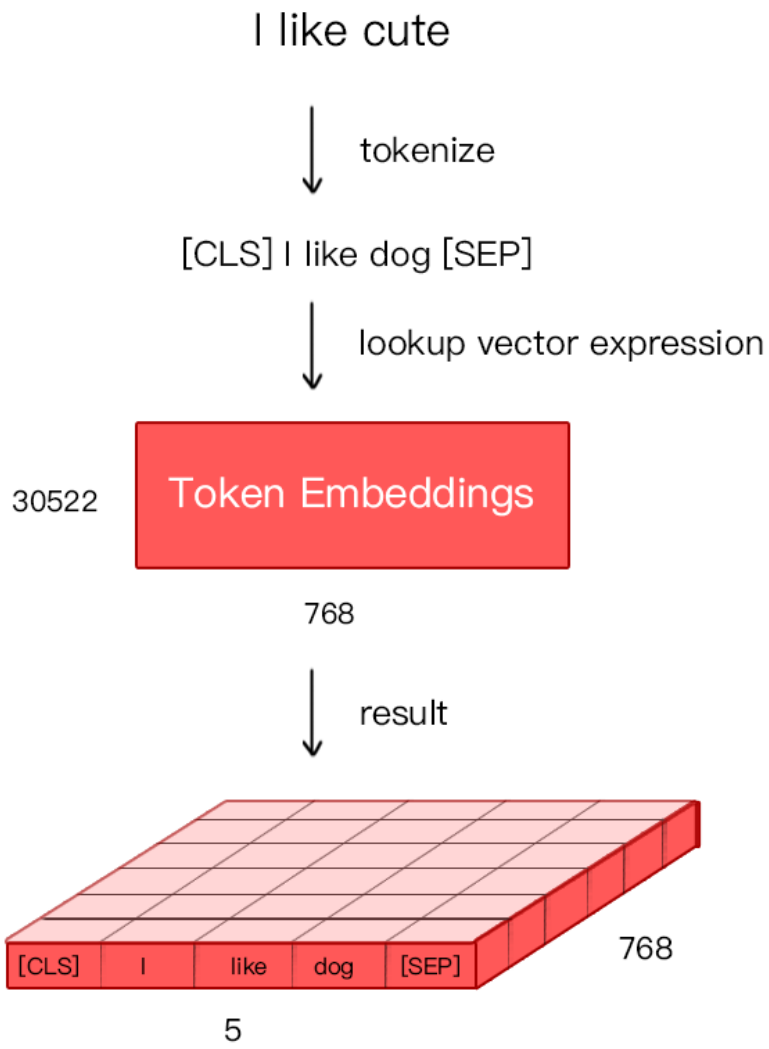


рис. 12

Встраивание сегментов

Bert способен справиться с задачей классификации пар предложений, которая заключается в определении того, являются ли два текста семантически похожими. Два предложения в паре предложений просто склеиваются вместе и подаются в модель, как Bert отличает пару предложений от двух предложений? Ответ - сегментные вкрапления. Слой Segment Embeddings имеет два векторных представления, первый вектор присваивает 0 каждому токену первого предложения, а второй вектор присваивает 1 каждому токену, где задача, такая как система вопросов и ответов, должна предсказать следующее предложение, поэтому входными данными являются связанные предложения. Если классификация текста содержит только одно предложение, то вкрапления Segement - это все 0.

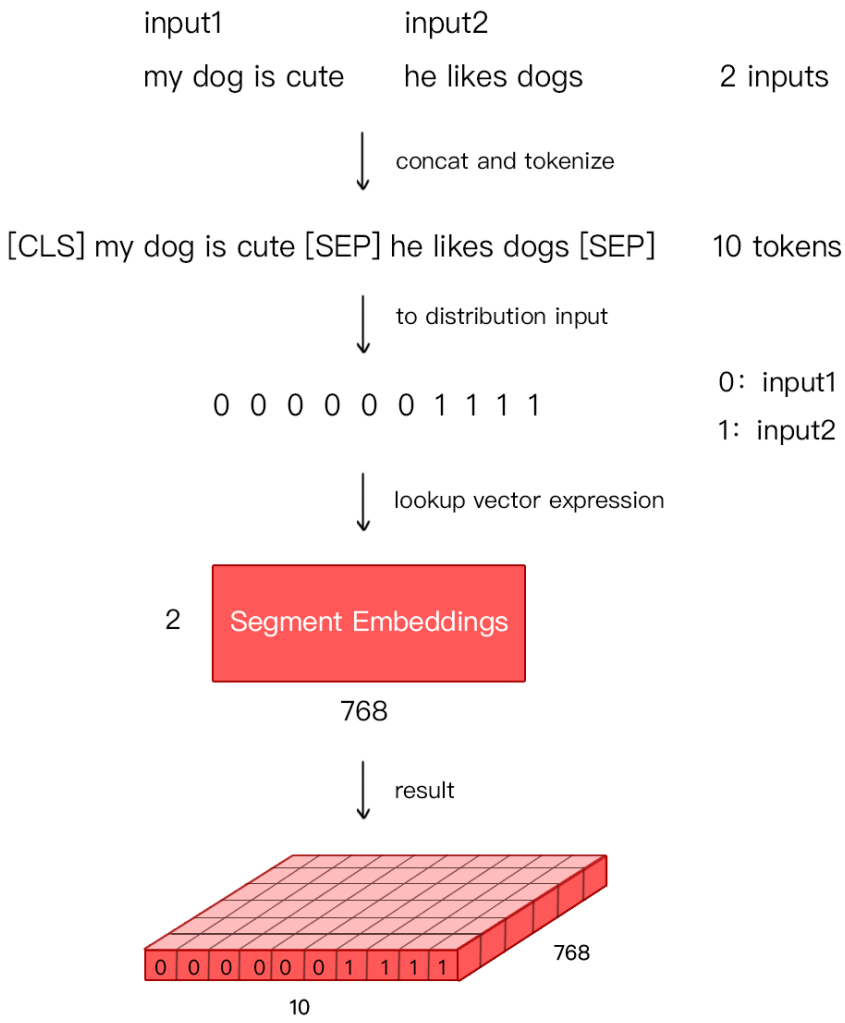


рис. 13

Встраивание позиции

Поскольку существует разница в семантической информации, которую несут слова/фразы, появляющиеся в разных позициях текста (например, "ты любишь меня" и "я люблю тебя"), хотя и ты, и я близки к слову любовь, они находятся в разных позициях и представляют разные значения.

В RNN второе "I" и первое "I" не выражают один и тот же смысл, потому что у них разные скрытые состояния. Для второго "I" скрытое состояние проходит через три слова " I think therefore" и содержит информацию предыдущих трех слов, в то время как первое "I" - это просто начальное значение. Таким образом, скрытое состояние RNN гарантирует различные представления выходного вектора для одних и тех же слов в разных позициях.

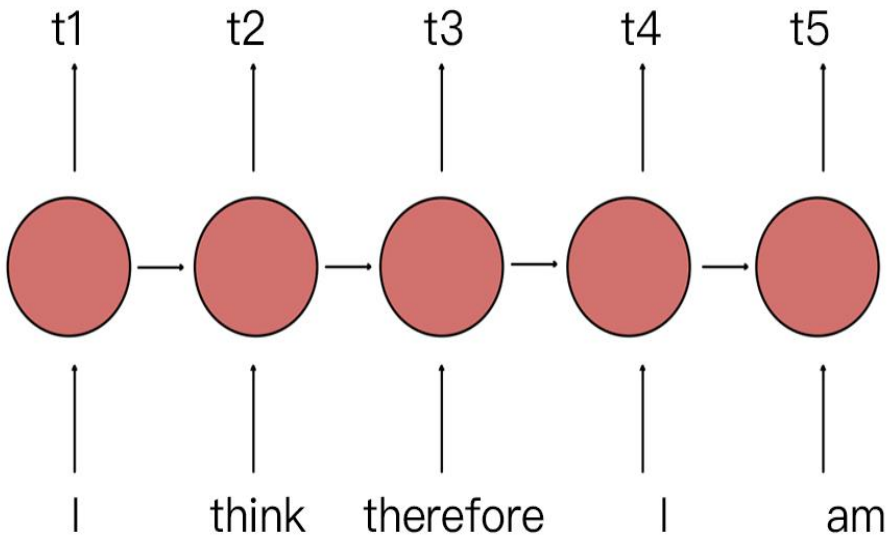


рис. 14

В этой задаче я буду использовать bert-as-service, чтобы помочь нам получить вектор слов Берта. Автором bert-as-service является доктор Сяо Хань из Китая. В настоящее время он является старшим научным сотрудником Tencent AI Lab и президентом Немецко-китайской ассоциации искусственного интеллекта. Набор данных Fashion-MNIST Сяо Ханя стал эталонным набором для машинного обучения с более чем 4,4 тыс. звезд на Github и более 300 научных ссылок за один год. Использование bert-as-service может очень легко помочь нам получить векторы[11] слов,сгенерированные Bert, а также использовать векторы слов, сгенерированные Bert, для следующего шага.

Глава 6. Конволюционная нейронная сеть – CNN

Глубокое обучение уже некоторое время добивается заметных успехов как в обработке изображений, так и в задачах НЛП. Как правило, задача обработки изображений решается с помощью CNN, чья уникальная конволюционная, объединяющая структура способна извлекать различные степени текстуры, структуры в изображении, и в конечном итоге объединяться с полностью связанными сетями для достижения агрегирования и вывода информации. В задачах анализа настроений (или задачах классификации коротких текстов) ограниченная длина предложения, компактная структура и способность выражать смысл независимо друг от друга позволяют CNN решать этот класс задач, и основная идея состоит в том, чтобы объединить модель n-грамм, о которой мы говорили выше, со сверточными операциями.

В задачах анализа настроений входными данными CNN являются уже не пиксели изображения, а слова предложения или предложения, представленные в виде матриц. Каждая строка матрицы соответствует слову

или предложению. То есть каждая строка представляет собой вектор слов, обычно что-то вроде вектора слов Bert или GloVe. В задачах с изображениями ядро свертки скользит по "блоку" изображения, но в области естественного языка мы обычно используем ядро свертки для скольжения по "строке" матрицы (слову).

Статья "Convolutional Neural Networks for Sentence Classification" (автор Yoon Kim) представляет собой попытку решения подобной задачи.[12]

В задаче анализа настроений (классификации текста) модель CNN состоит из четырех частей: входного слоя,сверточного слоя, слоя объединения и полностью подключенного слоя + softmax.

Ниже я по очереди опишу каждую часть.

6.1 Входной слой CNN

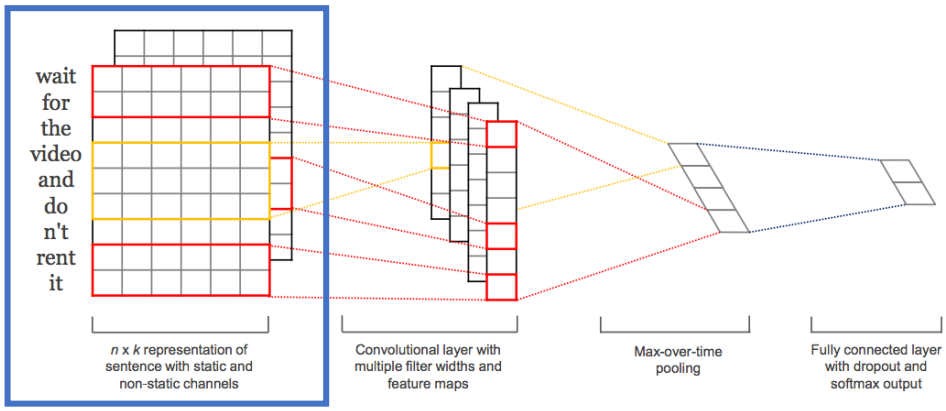


рис. 15

Входной слой CNN - это область синего квадрата на рисунке 15, входной слой представляет собой матрицу векторов слов, соответствующих словам в предложении, расположенных последовательно (сверху вниз). Если предположить, что предложение состоит из n слов, а векторы имеют размерность k, то эта матрица имеет размерность n × k (в CNN ее можно рассматривать как набор изображений высотой n и шириной k).

Тип этой матрицы может быть как статическим, так и динамическим. Статический означает, что вектор слов фиксирован, и в этой задаче мы используем вектор слов, сгенерированный GloVe и Bert, который будет представлять собой статический тип матрицы.

Динамическая, с другой стороны, это когда векторы слов также рассматриваются как оптимизируемые параметры в процессе обучения модели, и этот процесс обратного распространения ошибки, приводящий к изменению медианных значений векторов слов, обычно называют тонкой настройкой. При выполнении задач анализа настроений тонкая настройка,

как правило, невозможна. Это связано с тем, что его процесс является неконтролируемым (иногда приносящим зашумленные данные), а также с потреблением вычислительных ресурсов.

6.2 Конволюционный слой CNN

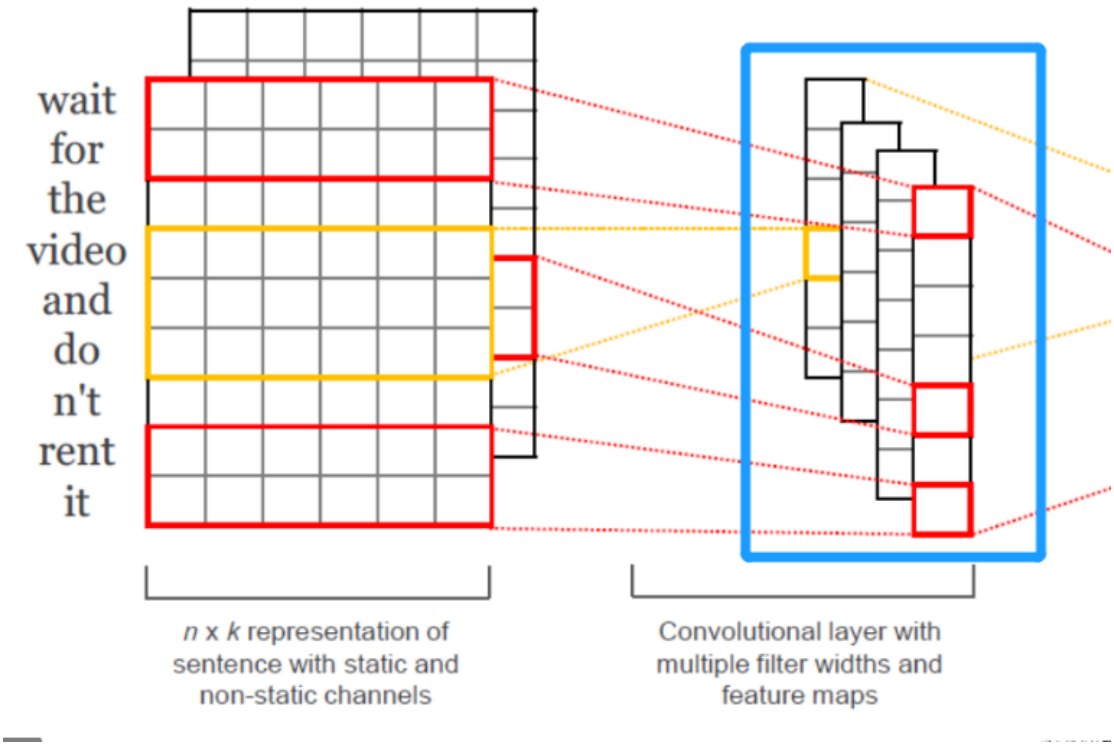


рис. 16

Конволюционный слой CNN показан в синей рамке на рисунке 16.

Входной слой получает несколько карт характеристик путем сверточных операций со сверточным окном размером h × k , где h обозначает количество продольных слов, а k - размерность вектора слов. При таком большом окне свертки получается множество "Карт характеристик" с номером столбца 1. (Этот принцип похож на принцип модели n-грамм, которую я упоминал выше в языковой модели). Модель n-грамм языка просто означает, что вероятность появления слова связана только с n-1 словами, предшествующими ему, с учетом ассоциаций между словами. В общем, вычислить 3-кортежную языковую модель в большом списке слов, не говоря уже о 4 или даже 5, будет непросто. Благодаря небольшому количеству параметров, CNN с легкостью выполняют подобные вычисления.

Размер ядра свертки: поскольку свертка выполняется в целых строках, необходимо только определить количество строк для каждой свертки, которое в модели n-грамм равно n. Вообще говоря, n обычно принимается равным 2, 3, 4, что также соответствует значению n в модели n-грамм.

В этой задаче классификации настроений я установил размер ядра свертки равным 5.

6.3 Объединяющий слой CNN

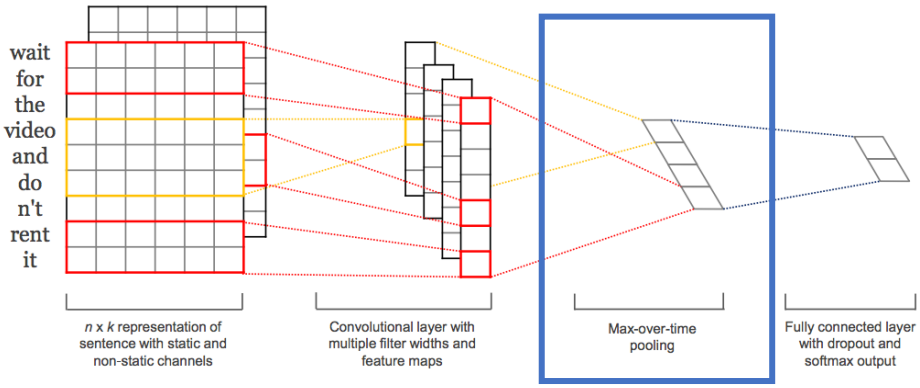


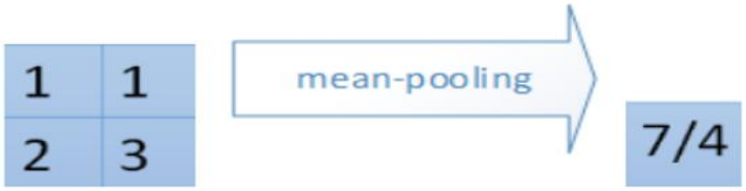
рис. 17

Принцип слоя объединения заключается в понижении дискретизации каждой непересекающейся области n*n, методами объединения являются MaxPooling, среднее объединение и т.д., из которых MaxPooling является наиболее широко используемым. Поскольку размер ядра свертки может быть уменьшен после максимального объединения с сохранением соответствующих признаков, оно в основном используется для уменьшения размерности.

Здесь я более подробно остановлюсь на двух способах объединения.

1) Среднее объединение.

Усреднение небольшой области, при условии, что размер окна объединения равен 2x2, представляет собой понижающую выборку 2x2 выхода предыдущего конволюционного слоя без перекрытия для получения среднего объединенного значения (как показано на рисунке 18).

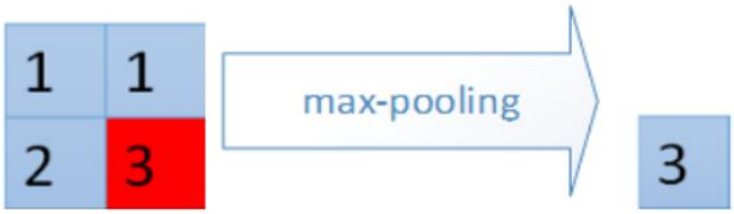


Усредненная площадь 2x2:

рис. 18

2) Максимальное объединение.

То есть, максимальное значение берется для небольшой области, предполагая, что размер окна объединения равен 2x2, то есть, выход предыдущего конволюционного слоя понижается путем взятия максимального значения 2x2 без перекрытия, и получается максимальное объединенное значение(как показано на рисунке 19).



Область 2x2 принимает максимальное значение:

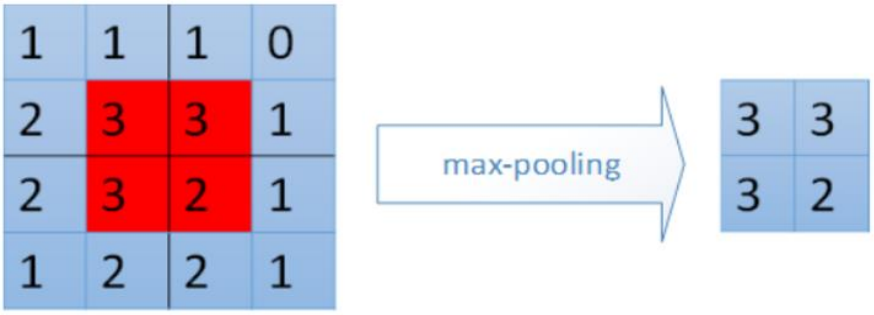


рис. 19

В этой задаче я также использовал максимальное объединение. Это связано с тем, что, согласно соответствующим исследованиям, эффект максимального объединения всегда превосходит эффект среднего объединения в результате многочисленных итераций экспериментов и сравнения влияния различных гиперпараметров на структуру модели CNN с точки зрения производительности и стабильности [13].

Уровень объединения выполняет следующие действия:

(1) Можно упомянуть о фиксации размеров выходной матрицы.

(2) Уменьшает размерность вывода, но сохраняет важные характеристики.

6.4 Полностью связанный слой CNN + слой softmax

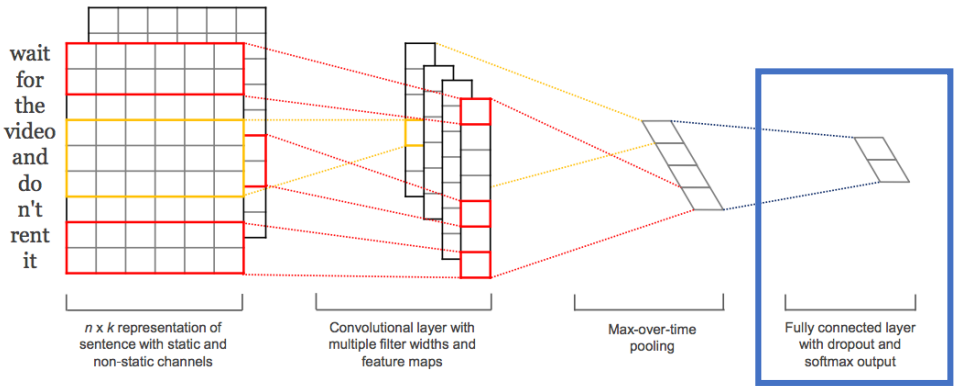


рис. 20

Полностью связанный слой CNN показан в синей рамке на рис.20.

Для того чтобы преобразовать векторы, выводимые со слоя объединения, в нужные нам прогнозы, нам необходимо добавить полностью подключенный слой с функцией активации Softmax и вывести вероятности для каждой категории с помощью функции активации Softmax. Поскольку набор данных, используемый в эксперименте, относительно мал, может легко произойти перебор, поэтому я добавил в программу метод Dropout, чтобы уменьшить перебор.

В своих экспериментах я пробовал dropout с разными параметрами, и лучшим оказался 0,5, так что если у вас, как и у меня, ограниченные вычислительные ресурсы, 0,5 может быть хорошим выбором.

В настоящее время отсев является наиболее популярным методом регуляризации сверточных нейронных сетей. Такой подход не позволяет нейронам вместе адаптироваться к признаку и заставляет их изучать полезные признаки по отдельности.

6.5 Функции потерь и оптимизаторы

Для обучения модели требуется функция потерь и оптимизатор. Поскольку это задача пяти классификаций и модель выводит значения вероятности (полностью подключенный слой с использованием функции активации softmax), я использую функцию потерь категориальная перекрестная энтропия.

Существуют две популярные функции потерь: категориальная перекрестная энтропия и бинарная перекрестная энтропия, категориальная перекрестная энтропия для мультиклассификации и бинарная перекрестная энтропия для биклассификации.

Конечно, существуют и другие функции потерь, например, вы можете выбрать mean_squared_error (средняя квадратичная ошибка). Однако в целом

категориальная перекрестная энтропия больше подходит для вероятности - она измеряет "расстояние" между вероятностными распределениями.

Я выбрал оптимизатор Adam, алгоритм, который выполняет оптимизацию градиента первого порядка для стохастической объективной функции, основанный на адаптивной оценке момента низшего порядка. Алгоритм Adam прост в реализации, имеет высокую вычислительную эффективность и низкие требования к памяти. Диагональное масштабирование градиента алгоритма Adam является инвариантным, поэтому он хорошо подходит для решения задач с большими данными или параметрами. Алгоритм также подходит для решения нестационарных задач с большим шумом и разреженными градиентами.

Сравнение алгоритма Адама с некоторыми другими подобными алгоритмами представлено в работе "Adam: A Method for Stochastic Optimization", где анализируется теоретическая сходимость алгоритма Адама, приводятся интервалы для скорости сходимости и доказывается, что скорость сходимости оптимальна в онлайновом выпуклом случае. оптимизация достигается оптимальным образом, и окончательные результаты также показывают, что алгоритм Адама лучше других алгоритмов оптимизации.[14]

Выводы

По предложению доцент Блеканова Ивана Станиславовича я использовал два набора данных, английский пятикатегорийный набор данных и китайский пятикатегорийный набор данных, соответственно. Каждый набор данных был обучен с помощью GloVe+CNN и Bert+CNN соответственно.

Английский набор данных с пятью классификациями.

1) GloVe+CNN:

```
282/282 [==============================] - 24s 86ms/step - loss: 0.9835 - acc: 0.5825 - val_loss: 0.9857 - val_acc: 0.5739
Epoch 22/25
282/282 [==============================] - 24s 86ms/step - loss: 0.9685 - acc: 0.5889 - val_loss: 0.9838 - val_acc: 0.5776
Epoch 23/25
282/282 [==============================] - 24s 86ms/step - loss: 0.9709 - acc: 0.5901 - val_loss: 0.9809 - val_acc: 0.5796
Epoch 24/25
282/282 [==============================] - 24s 86ms/step - loss: 0.9690 - acc: 0.5872 - val_loss: 0.9825 - val_acc: 0.5796
Epoch 25/25
282/282 [==============================] - 24s 86ms/step - loss: 0.9720 - acc: 0.5882 - val_loss: 0.9777 - val_acc: 0.5821
```

рис. 21

2) Bert+CNN:

```
Epoch 1/10
282/282 [==============================] - 1688s 6s/step - loss: 1.7723 - acc: 0.4602 - val_loss: 1.7151 - val_acc: 0.4731
Epoch 2/10
282/282 [==============================] - 1699s 6s/step - loss: 1.6983 - acc: 0.4758 - val_loss: 1.6469 - val_acc: 0.4731
Epoch 3/10
282/282 [==============================] - 1698s 6s/step - loss: 1.6332 - acc: 0.4708 - val_loss: 1.5862 - val_acc: 0.4731
Epoch 4/10
282/282 [==============================] - 1700s 6s/step - loss: 1.5747 - acc: 0.4689 - val_loss: 1.5325 - val_acc: 0.4731
Epoch 5/10
282/282 [==============================] - 1696s 6s/step - loss: 1.5238 - acc: 0.4770 - val_loss: 1.4851 - val_acc: 0.4731
Epoch 6/10
282/282 [==============================] - 1700s 6s/step - loss: 1.4784 - acc: 0.4711 - val_loss: 1.4432 - val_acc: 0.4731
Epoch 7/10
282/282 [==============================] - 1770s 6s/step - loss: 1.4375 - acc: 0.4727 - val_loss: 1.4064 - val_acc: 0.4731
Epoch 8/10
282/282 [==============================] - 1857s 7s/step - loss: 1.4028 - acc: 0.4733 - val_loss: 1.3739 - val_acc: 0.4731
Epoch 9/10
282/282 [==============================] - 1716s 6s/step - loss: 1.3708 - acc: 0.4741 - val_loss: 1.3454 - val_acc: 0.4731
Epoch 10/10
282/282 [==============================] - 1702s 6s/step - loss: 1.3421 - acc: 0.4744 - val_loss: 1.3203 - val_acc: 0.4731
```

рис. 22

Как видно на рисунке 21, после 25 циклов обучения точность GloVe+CNN достигает 58.21%.

Как показано на рисунке 22, при использовании Bert+CNN точность составляет всего 47.31%.

Команда, которая достигла относительно высокой точности в проблеме классификации пяти настроений до 2020 года, - это команда Цзыцзюнь Суня, которая использовала самообъясняющиеся структуры для улучшения NLP-моделей [15] в проблеме классификации пяти настроений и

использовала самую большую версию предварительного обучения вектора слов Берта. модель. Наибольшая точность их модели составила 59,1%.

Причина точности всего 58% заключается в том, что в наборе данных много нерелевантных данных и много предложений, которые не точно выражают настроение.

Например: 《yous think at responding to texts》.

Эта фраза означает: 《вы думаете о том, как отвечать на сообщения》.

Это предложение имеет оценку 1, что является самым низким показателем, указывающим на гнев. Но из этого предложения мы не можем точно определить настроение предложения.

Другая причина заключается в том, что чем больше разнообразие классификаций, тем большее количество признаков требуется; например, дихотомическая классификация более точна, чем квинтуплетовская (72% точности для одной и той же модели).

Эти два пункта также объясняют, почему точность составляет всего 47.31% в модели Bert+CNN. Как я уже говорил, когда представлял Bert, размерность Bert составляет фиксированные 768 измерений, что означает, что вектор слов, обученный Bert, несет больше информации (например, положение слова в предложении и т.д.). И в процессе, недействительные (или бесполезные) данные усиливаются, помните принцип max-pooling, который я показал в слое объединения CNN? Каждый раз, когда вы играете в пул, вы выбираете максимальное значение. Другими словами, он отбрасывает некоторые значения, и в процессе CNN не может определить, какие данные полезны, а какие бесполезны. Отсюда также ясно, что Bert не подходит для объединения с базовой моделью CNN. Или же следует искать новое решение при использовании Bert+CNN, например, добавить в модель механизм внимания.

Чтобы проверить свой вывод, я использовал китайский набор данных из пяти категорий. Китайский язык, или современный китайский стандартный мандарин (самый распространенный стандартный язык в Китае сегодня), имеет очень простую грамматику, что означает, что слово в китайском языке должно выражать больше информации. Обычно размерность вектора слов на китайском языке составляет от 150 до 300 измерений, в то время как размерность вектора слов на английском языке обычно составляет от 50 до 100 измерений.

Высокая размерность является характеристикой Bert, другими словами, это означает, что векторы слов, полученные при обработке китайского языка с помощью Bert, будут содержать более эффективные характеристики.

Китайский набор данных по пяти категориям

1) GloVe+CNN:

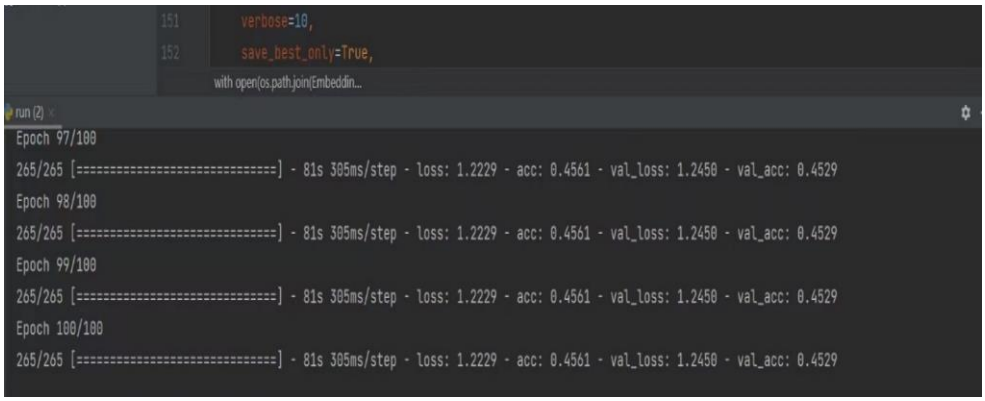


рис. 23

2) Bert+CNN:

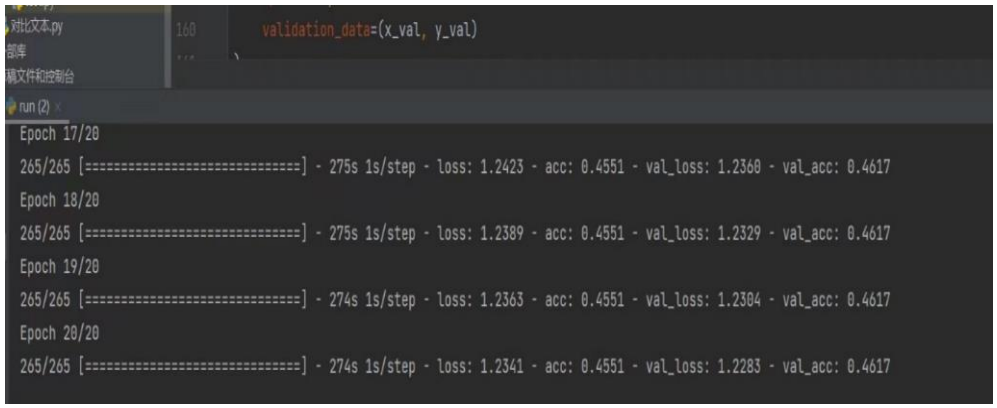


рис. 24

Используя китайский набор данных для классификации пяти слов, мы видим, что модель Bert+CNN примерно на 1% точнее модели Glove+CNN, что подтверждает связь между размерностью вектора слов, максимальным слоем объединения CNN и показателем точности. Более низкая скорость Bert (39 часов для обучения вектора слов из 40 000 слов, 1794 секунды на цикл) также может быть продемонстрирована с точки зрения времени выполнения, требуя больше ресурсов, чем GloVe.

Заключение

В данной работе проводится исследование пяти классификаций для анализа настроений с использованием GloVe, Bert и CNN.

Результаты показывают, что комбинированная модель GloVe+CNN является лучшим выбором, чем комбинированная модель Bert+CNN для решения задачи классификации по пяти признакам для анализа настроений с ограниченными вычислительными ресурсами.

Я добавил больше слоев объединения в комбинированную модель Bert+CNN с целью снижения вычислительной нагрузки, но это также влечет за собой проблему снижения точности.

В своей экспериментальной работе я обнаружил, что решение проблемы мультиклассификации настроений на основе бесполезных данных является сложной задачей.

В настоящее время у меня есть два способа решения проблемы бесполезных данных.

1) Используйте ручной подход для очистки набора данных.

2) Ввести в CNN некоторый механизм, например, механизм "внимания".

Второй способ - это то, над чем я собираюсь начать работать, механизм внимания - это способ быстро отфильтровать ценную информацию из большого количества информации, используя ограниченные ресурсы. Механизм внимания в глубоком обучении опирается на человеческое мышление внимания и широко используется в различных типах задач глубокого обучения, таких как обработка естественного языка, классификация изображений и распознавание речи, с замечательными результатами.

В ходе экспериментов по обработке сырых данных я обнаружил, что многие пользователи любят использовать смайлики (или картинки) для выражения своих эмоций, но в силу своей собственной неспособности (я не занимался программированием и алгоритмами машинного обучения до поступления в Санкт-Петербургский государственный университет), я могу использовать только очень базовые алгоритмы для построения моделей (моя модель построена на основе модели для классификации текстов) и не обладаю способ использования этой информации об изображении. Но благодаря этому исследованию я понял, где я хочу продолжить свои исследования. Причина, по которой я выбрал модель CNN, заключается также в том, что я надеюсь в будущем использовать эмодзи в качестве признаков для задач анализа настроения.

Список литературы

1. Сергеев С.Л. Архитектуры вычислительных систем. БХВ-Петербург, 2010. 240.
2. Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. ICLR Workshop, 2013
3. T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. Distributed Representations of Words and Phrases and their Compositionality. NIPS 2013
4. Le Q, Mikolov T. Distributed representations of sentences and documents[C]//International Conference on Machine Learning. 2014: 1188-1196.
5. Goldberg Y, Levy O. word2vec Explained: deriving Mikolov et al.'s negative-sampling word-embedding method[J]. arXiv preprint arXiv:1402.3722, 2014.
6. Rong X. word2vec parameter learning explained[J]. arXiv preprint arXiv:1411.2738, 2014.
7. Pennington J , Socher R , Manning C . Glove: Global Vectors for Word Representation[C]// Conference on Empirical Methods in Natural Language Processing. 2014.
8. Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova . BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv preprint arXiv:1810.04805, 11 Oct 2018 (v1), last revised 24 May 2019 (v2)
9. Bird, Steven, Edward Loper and Ewan Klein (2009).Natural Language Processing with Python. O'Reilly Media Inc.
10. Fxsjy, jieba, https://github.com/fxsjy/jieba
11. xiao2018bertservice,bert-as-service,XiaoHan,https://github.com/hanxiao/bert-as-service,2018
12. Kim, Y. (2014). Convolutional Neural Networks for Sentence Classification. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014), 1746–1751.
13. Zhang, Y., & Wallace, B. (2015). A Sensitivity Analysis of (and Practitioners' Guide to) Convolutional Neural Networks for Sentence Classification
14. Diederik P. Kingma, Jimmy Ba, 2014/12/22,Adam: A Method for Stochastic Optimization
15. Zijun Sun, Chun Fan, Qinghong Han, Xiaofei Sun, Yuxian Meng, Fei Wu, Jiwei, 3 Dec 2020, Self-Explaining Structures Improve NLP Models