УДК 004.8

Восстановление знаков препинания в неструктурированном тексте с использованием сверточных нейронных сетей.

Тирских Д.В.[1]
Научный руководитель – к.ф.-м.н., Бернгардт О. И.[1,2]
[1]Иркутский государственный университет
[2]Институт солнечно-земной физики СО РАН

В данной работе рассмотрены несколько архитектур сверточных нейронных сетей для решения задачи восстановления знаков препинания в неструктурированном тексте, при моделировании задачи как проблемы многоклассовой классификации.
Ключевые слова: Машинное обучение, пунктуация, обработка текстов, сверточные нейронные сети.

Введение

Важность решения задачи расстановки знаков препинания в сплошном неструктурированном тексте заключается в практической пользе, которое оно принесет для различных направлений обработки естественных языков. К примеру, в настоящее время задача распознавания человеческой речи решается множеством различных способов, но большинство из них на выходе выдают сырой сплошной текст, состоящий из набора распознанных слов. Такоq текст трудно читаем для человека, привыкшего к тексту, разбитому на отдельные предложения. Характерным примером такого сервиса является Youtube, который способен распознавать произносимый на видео текст, но не расставляет знаки препинания. К тому же отсутствие структуры в распознанном тексте затрудняет работу с ним систем автоматического перевода.

Постановка задачи

Текст разбивается на отдельные токены, каждому из которых присваивается класс, соответствующий знаку пунктуации, идущему после него. Имея на входе набор из n токенов, нужно предсказать знак препинания, стоящего по середине. То есть предсказания делаются не для каждого отдельного, а лишь для слова стоящего в середине окна, поданного на вход как показано на рис. 1. Для предсказания знаков для всего текста используется обход скользящим окном.

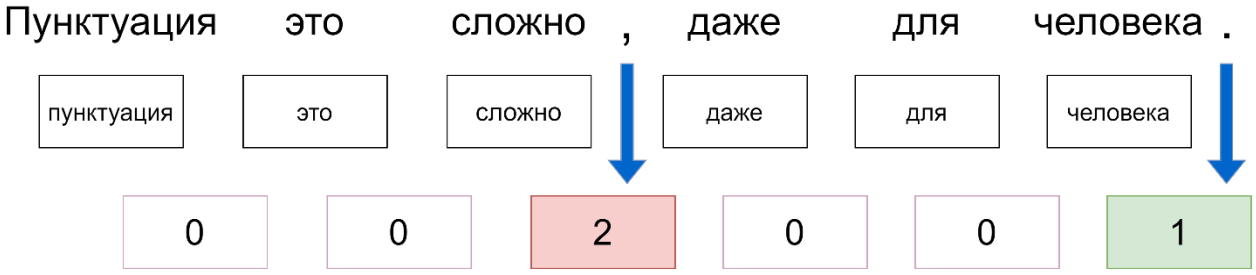

Рис.1. Постановка задачи

Подготовка данных

В качестве данных, на которых будет производится обучение сети, был выбран корпус SynTagRus (*Syntactically Tagged Russian text corpus).* Данный корпус был выбран из-за наличие в нем текстов разной длинны и жанров, что позволит избежать подстройки сети под какой-то определенный тип текстов. На данный момент корпус состоит из 87336 предложений [1]. Для создания датасета тексты были разбиты на отдельные токены, каждому из которых был присвоен соответствующий класс. Размер датасета составил 1203755 токенов. Так же, чтобы окончательно убедиться в работоспособности предложенного решения и

протестировать его работу на текстах определенного жанра, был собран дополнительный датасет из примерно 6000 тысяч новостных текстов с портала лента.ру.

Постановка задачи приведенная выше неизбежно ведет к проблеме несбалансированности классов. В особенности это заметно для классов вопросительного и восклицательного знака, их суммарное количество составляет меньше одного процента от общего объема данных, что делает затруднительным их предсказание с помощью машинного обучения. Поэтому было решено объединить все классы, означающие конец предложения в один общий класс END – конец предложения. Распределение классов в итоговом датасете показано на рис. 2.

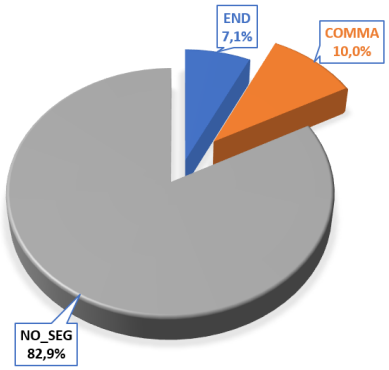

Рис. 2. Распределение классов в датасете

Архитектура и обучение нейронной сети

Задача решается методом переноса знаний от обобщенной языковой модели и построением оптимального классификатора, прогнозирующего необходимый знак препинания.

С помощью модели BERT, а именно, тренированной на русской части википедии и новостных текстах сети – RuBert [2], токены из датасета преобразуются в векторные представления (эмбеддиги) длинной в 768 значений. В качестве эмбеддингов была использована сумма выходов последних четырех слоев сети. Токены разбиваемые сетью на несколько частей были усреднены. Использование сети BERT позволяет учитывать контекст при создании эмбеддингов, что решает проблему кодирования многозначных слов [3].

Для вынесения предсказаний были созданы и протестированы две архитектуры сверточных сетей – вертикальная и горизонтальная. Идея первой заключалась в обработке входного текста как набора n-gram слов группой сверток различных размеров. Идея второй архитектуры была в том, чтобы вместо n-грамм слов производить свертки проходя по измерению эмбеддингов слева на право, получая при этом свертки различных комбинаций частей эмбеддингов сразу для всего окна. Каждое значение в эмбеддинге кодирует какой-то аспект значения слова, следовательно беря различные части этого вектора мы рассматриваем отдельные оттенки его смысла. Идея заключалась в том, что, рассмотрев различные комбинации смыслов входной последовательности, сети будет проще определится с предсказанием знака. Обе архитектуры представлены на рисунке 3.

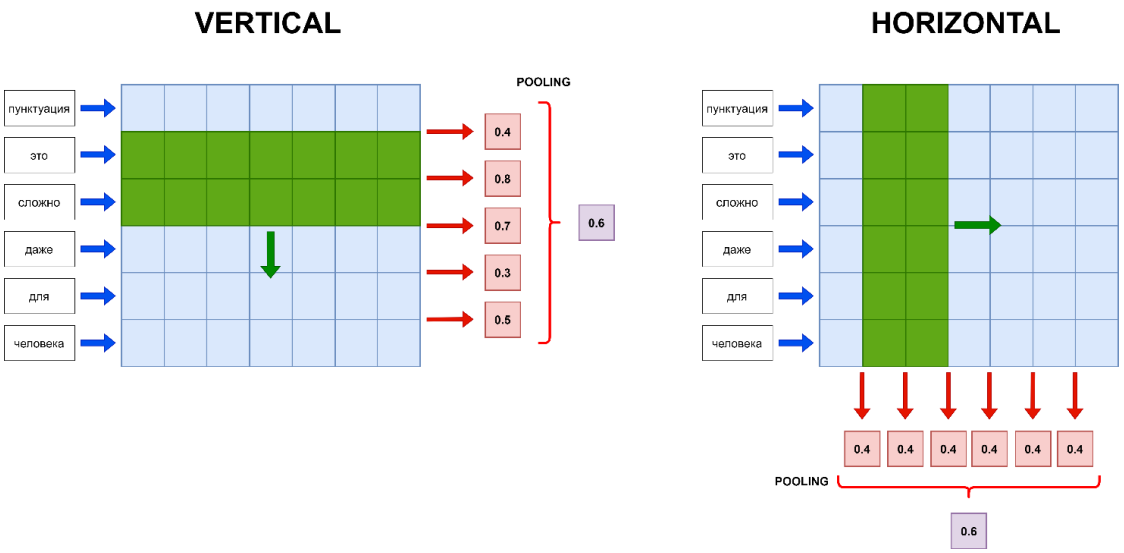

Рис. 3. Вертикальная и горизонтальная архитектуры

При использовании вертикального подхода получилось обучить первый вариант модели, но скорость обучения была низкая, а сам процесс нестабильным. Порой сеть отказывалась обучаться вовсе. Горизонтальный подход же в полной мере оправдал себя, так как при подборе оптимальных размеров и шага сверток, удалось достичь лучших по сравнению с изначально архитектурой результатов за меньшее время. Сеть стала работать лучше, быстрее, а процесс обучения стабилизировался. В последствии горизонтальная архитектура была улучшено добавлением трех независимых наборов сверток для начала, середины и конца предложения как показано на рисунке 4.

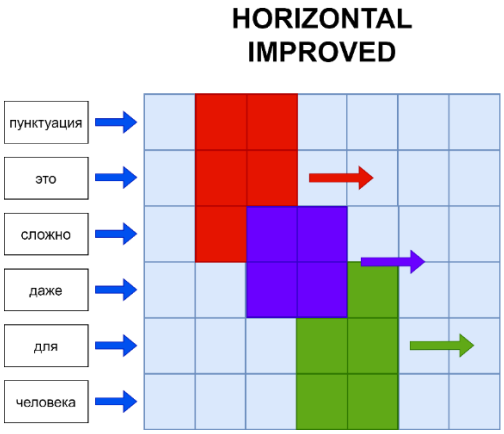

Рисунок 4. Улучшенная горизонтальная архитектура

Окончательная архитектура оптимального классификатора выглядела следующим образом. На вход сети подается окно длиною в N=6 слов. Начало, середина и конец окна параллельно обрабатываются набором сверток, которые движутся слева на право. Всего в каждом наборе по 7 сверток с различными размерами ядра и шага. После этого результаты сверток попадают на пулинг слой, в результате чего из каждого набора сверток получается по 175 значений. Далее выходы всех трех блоков конкатенируются и подаются на полносвязные решающие слои. Количество обучаемых параметров в полученной нейронной сети – 429 103. Вес получившийся модели – 1,7 мегабайт.

Метрики

В качестве метрик, используемых для оценивания качества работы модели и выбора оптимальной модели на стадии обучения были выбраны: Precision, Recall, F1-score. Основной метрикой стала F1 – т. к. она совмещает в себе Precision и Recall, что важно для решаемой задачи т.к. полученный классификатор должен сохранять баланс между точностью предугадывания знаков препинания и количеством расставленных им знаков.

Результаты работы сети

Используя описанную ранее архитектуру, удалось добиться высоких значений метрики F1 порядка 0.81 на тестовой и валидационной выборках. Значения метрик классификации полученных на различных датасетах (обучающем, тестовом и новостных текстах с lenta.ru) представлены в таблице. Класс «NOSEG» соответствует отсутствию знака, «COMMA» – знаку запятой, «END» – классу конца предложения.

Таблица

Показатели метрик

Датасет	**Класс**	**Precision**	**Recall**	**F1**	**Количество**
Test	NOSEG	0.96	0.97	0.96	98824
	COMMA	0.76	0.71	0.73	12412
	END	0.74	0.71	0.73	8631
	Macro avg	0.82	0.80	0.81	
Validation	NOSEG	0.96	0.97	0.97	103916
	COMMA	0.78	0.73	0.75	12573
	END	0.74	0.74	0.74	8623
	Marco avg	0.83	0.81	0.82	
Lenta	NOSEG	0.98	0.98	0.98	913689
	COMMA	0.88	0.76	0.82	83244
	END	0.79	0.89	0.84	71625
	Macro avg	0.88	0.88	0.88	

Интересно на этом фоне выглядят результаты полученные на датасете состоящем из новостных текстов lenta.ru. Наблюдаемые показатели метрик на нем примерно на 10 процентов выше, что свидетельствует о лучшей работе классификатора на новостных текстах.

Совершаемые сетью ошибки можно разделить на три типа: отсутствие знака, постановка лишнего знака, постановка неверного знака. Все это проиллюстрировано на рисунке 4. (Красный цвет – отсутствие знака; Зеленый цвет – неверный знак; Желтый цвет – лишний знак).

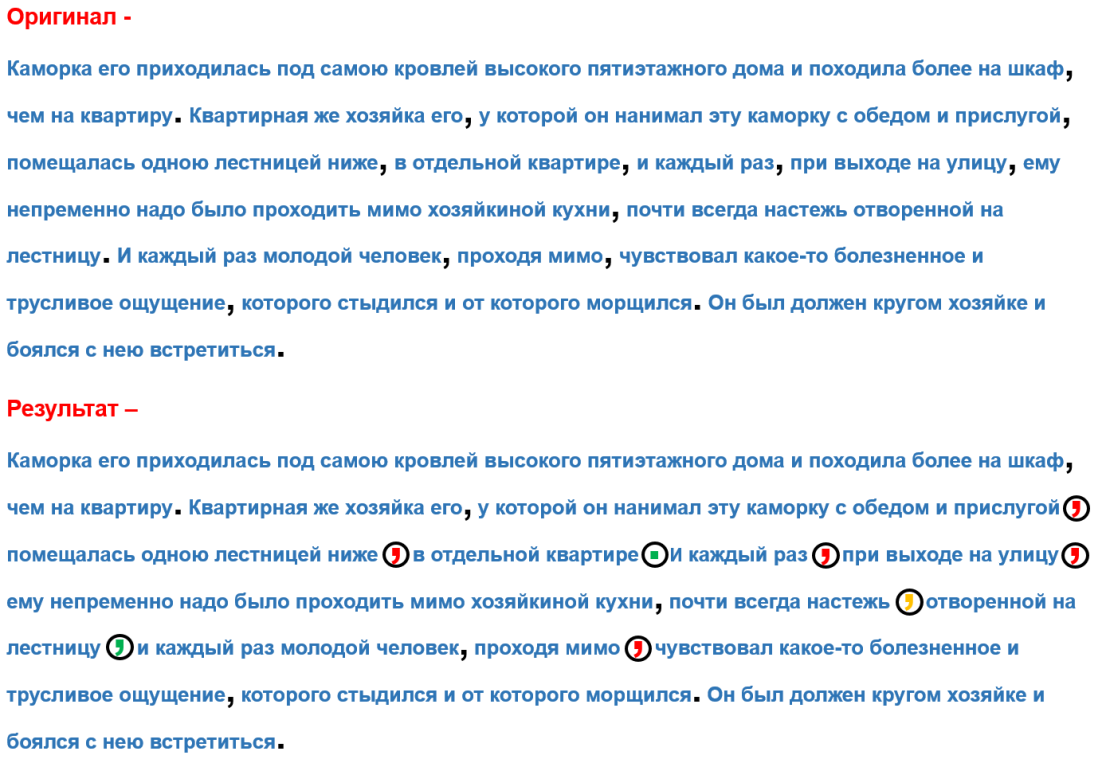

Рис. 4. Демонстрация работы сети.

Стоит обратить внимание что перед одним и тем же сочетанием слов – «и каждый раз», сеть в двух разных случаях поставила разные знаки. Это свидетельствует о том, что получившаяся нейронная сеть не просто заучила сочетания слов, перед которыми обычно стоят знаки, а анализирует текст и выставляет знаки основываясь именно на контексте.

Скорость роботы сети

На тексте диной в 120 слов (средний темп речи), с использование графического ускорителя Tesla T4, полный цикл работы сети занимает 40 мс., из которых 13 мс. затрачиваются на вынесение предсказаний.

Выводы.

Применение методов машинного обучение позволило получить одно из возможных решение поставленной задачи. Результаты работу могут быть применены для улучшения работы систем автоматического распознания речи и систем машинного перевода. В дальнейшем полученное решение может быть улучшено для повышения качества работы или для повышения скорости предсказаний, что позволить обрабатывать текст в режиме реального времени.

Литература

1. Universal Dependencies Russian SynTagRus [Электронный ресурс]. – Режим доступа: https://universaldependencies.org/treebanks/ru_syntagrus/index.html (дата обращения: 12.04.2022).
2. BERT in DeepPavlov [Электронный ресурс]. – Режим доступа: http://docs.deeppavlov.ai/en/master/features/models/bert.html (дата обращения: 13.04.2022).
3. Andreas Pogiatzis NLP: Contextualized word embeddings from BERT [Электронный ресурс]. – Режим доступа: https://towardsdatascience.com/nlp-extract-contextualized-word-embeddings-from-bert-keras-tf-67ef29f60a7b (дата обращения: 13.04.2022).