Charformer

Работа посвящена модификации классических моделей трансформеров для работы с входными данными как с последовательностью символов, а не токенов.
Существующие модели легко перестраиваются под работу с символами (аналогично сопоставляя их некоторым обучаемым векторам и подавая в модель), однако есть проблемы с производительностью (большое число входных векторов) и с достигаемым качеством (модели сложно работать с очень небольшими кусками текста, которые не имеют смысла по отдельности). Основной вклад статьи в том, что предлагается способ сжатия последовательности векторов символов с помощью специального токенизатора, по сути являющегося частью итоговой нейросети, который позволяет не терять производительность как при наивном подходе, а также даёт лучшее качество.

Статья появилась на архиве ещё в 2021, опубликована на ICLR 2022. 8 авторов из Google Research и DeepMind, у всех ранее были статьи по NLP, у части авторов статьи про некоторые проблемы, где предложенный в Charformer метод может быть полезен (например в мультиязычных моделях).

Предложенный метод можно рассматривать как закономерное улучшение модели ByT5 [3], так как все основные идеи по архитектуре/процессу обучения остаются прежними (каждый байт считается символом, увеличивается число параметров encoder части трансформера, предобучение на предсказании нескольких последовательных байт) и к ним добавляется идея группировки входных векторов в меньшую последовательность. В оригинальной статье никакая группировка не использовалась и векторов было столько же, сколько и символов в тексте.
Сама идея группировки тоже не новая и использовалась в модели CANINE [4], однако у Charformer предлагается более эффективный метод и есть множество других отличий от данной модели. Так например, в модели CANINE на вход подаются не байты, а символы юникода, из-за чего требуется хеширование для сопоставления векторов.
Также, статья [2] вероятно оказала влияние на появление идеи Charformer: согласно описанным сравнениям у языков моделей параметры векторов представлений токенов занимают до 21% всех обучаемых параметров, а в случае мультиязычных значительно возрастает число токенов, а параметры для их векторов начинают занимать до 71%. Соответственно встаёт вопрос об уменьшении параметров на хранение представлений, который может решаться с помощью подхода Charformer (необходимо всего 256 векторов для соответствия каждому возможному значению байта).

Статья достаточно новая, поэтому пока нет прямых продолжений или значимых цитирований. Вероятно будет полезно ознакомиться с архитектурой Perceiver IO [1] от DeepMind, предназначеной для работы с произвольным данными. В статье приводится эксперимент, где модель работает с текстом как с последовательностью байт аналогично Charformer и также показывает лучшие результаты по сравнению с CANINE[4].

У предложенного метода есть ряд достоинств и недостатков.

Так, авторам удаётся снизить размер словаря для хранения векторов представлений во много раз (и вообще говоря зафиксировать его размер 256 для любого языка текста), однако сэкономленные на этом параметры приходится “переносить” в encoder часть трансформера. Кроме того, из-за отказа от большого словаря, для получения значимых представлений слов/предложений нельзя просто усреднить векторы токенов, а необходимо запускать некоторую часть модели, что может быть слишком затратно в ряде задач.
Достоинством является отказ от предобработки входного текста, так как это является относительно сложной инженерной задачей и значительно усложняется при работе с несколькими языкими и в других нестандартных случаях.
Наконец, в статье утверждается, что Charformer достигает лучшего качества, чем аналогичные модели без токенов, а также лучшего качества чем классические языковые модели в некоторых задачах. Данные утверждения однако требуют больших подтверждений, в частности необходимо больше сравнений с аналогичными подходами и с видоизменениями предложенного, так как предложен сложный метод группировки векторов, многие детали которого неочевидно почему выбраны именно так, а не иначе.

В статье указывается, что используемые свёртки и другие локальные механизмы делают предположение о смысловой близости подряд идущих символов, что хотя и кажется очевидным, однако верно не для всех языков, например для таких языков как Арабский из-за явления неконкатенативной морфологии. Подобные проблемы могут возникать и из-за других языковых особенностей, что даёт большой простор для дальнейших исследований мультиязычных моделей.

1) https://arxiv.org/pdf/2107.14795.pdf
2) https://arxiv.org/pdf/2010.12821.pdf
3) https://arxiv.org/pdf/2105.13626.pdf
4) https://arxiv.org/pdf/2103.06874.pdf