Thinking like transformers

(Gail Weiss, Yoav Goldberg, Eran Yahav)

1. Содержание и вклад.
   В статье предложен язык обработки последовательностей RASP в качестве модели для энкодера трансформера. Приведены некоторые примеры кода для задач, которые мог бы решать трансформер. Утверждается, что RASP может помочь понять, как работает трансформер.

2. Сильные стороны.
   - Предложена новая интересная идея понимания трансформеров с помощью языка программирования. Аналогов у работы нет.
   - Подробное описание RASP, примеры решаемых задач(гистограммы подсчета частот токенов, правильные скобочные последовательности из разных типов скобок(shuffle-dyck-2)).
   - В статье делается попытка установить связь между операциями на языке RASP и трансформерами. Это может помочь проанализировать минимально необходимое количество слоев и верхнюю границу на количество attention heads для решения задачи трансформером.

3. Слабые стороны.
   - Язык подходит для кодирования только части задач. Как отмечают сами авторы, “we do not expect any researcher to implement, e.g., a strong language model or machine-translation system in RASP—these are not realizable in any programming language”. Это значит, что интуиция получается не для всех задач, которые могут решать трансформеры.
   - Введение нового языка программирования не обосновано. Предложенные базовые функции присутствуют(или легко реализуемы) в других языках программирования. Дополнительные ограничения можно ввести.
   - Эксперименты были добавлены после комментариев на openreview.
   - В статье утверждается, что операции “select” и “aggregate” связаны с матрицами внимания, однако в секции экспериментов есть только одно полное совпадение, в остальных слчаях совпадение либо частичное, либо вовсе отсутствует.
   - Эксперименты не вполне подтверждают качество оценки RASP на количество слоев/attention heads. В эксперименте с сортировкой лучшие результаты получились не для значений полученных их RASP. В эксперименте Dyck-2 разница в качестве(accuracy) между RASP архитектурой и архитектурой меньшего размера менее 1%.
   - Также все задачи достаточно маленькие и им требуется всего 1-2 слоев/attention heads, то есть вариантов архитектур для сравнения мало.

4. Качество текста.
   Статья написана достаточно понятным языком.
5. Воспроизводимость.

- Есть репозиторий https://github.com/tech-srl/RASP и много примеров кода на RASP - решения задач, а так же код для экспериментов https://github.com/tech-srl/RASP-exps.
- В статье даны значения параметров для экспериментов.
- Есть только описание “компиляции” языка в архитектуру трансформера, автоматически это сделать не получится.

6. Оценки:

Статья 5
Уверенность 4