Thinking Like Transformers

📊Докладчик: Александра Сендерович👩‍💻
📜Рецензент: Анастасия Дроздова🦀
🔍Практик-исследователь: Дарья Сапожникова🕵️‍♀️
💻Хакер: Полина Гусева🥷

https://arxiv.org/pdf/2106.06981.pdf

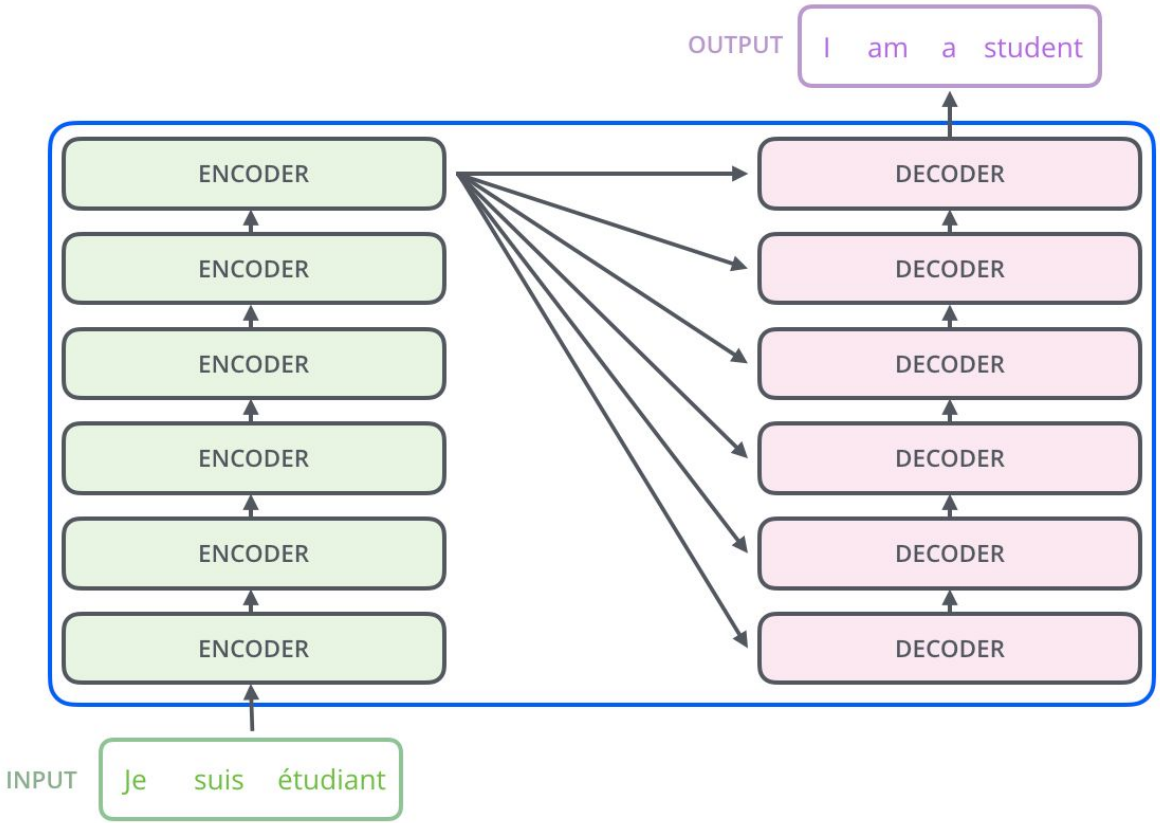
Thinking Like RNNs

Ранее -- RNN:

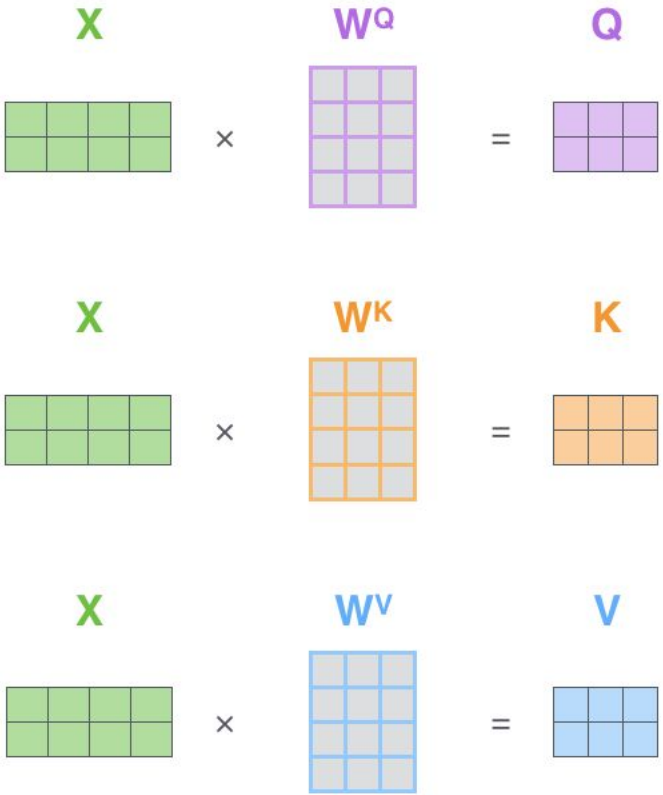
- RNN хорошо распознают грамматическую структуру
- Формальный язык -- множество конечных строк над конечным алфавитом
- Рекуррентные нейронные сети аналогичны конечным автоматам

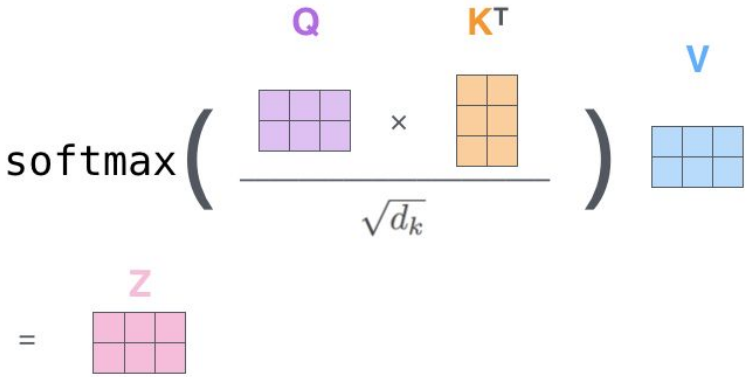
- Основная цель -- интерпретируемость: приближаем сеть простой моделью

Трансформер



Внимание

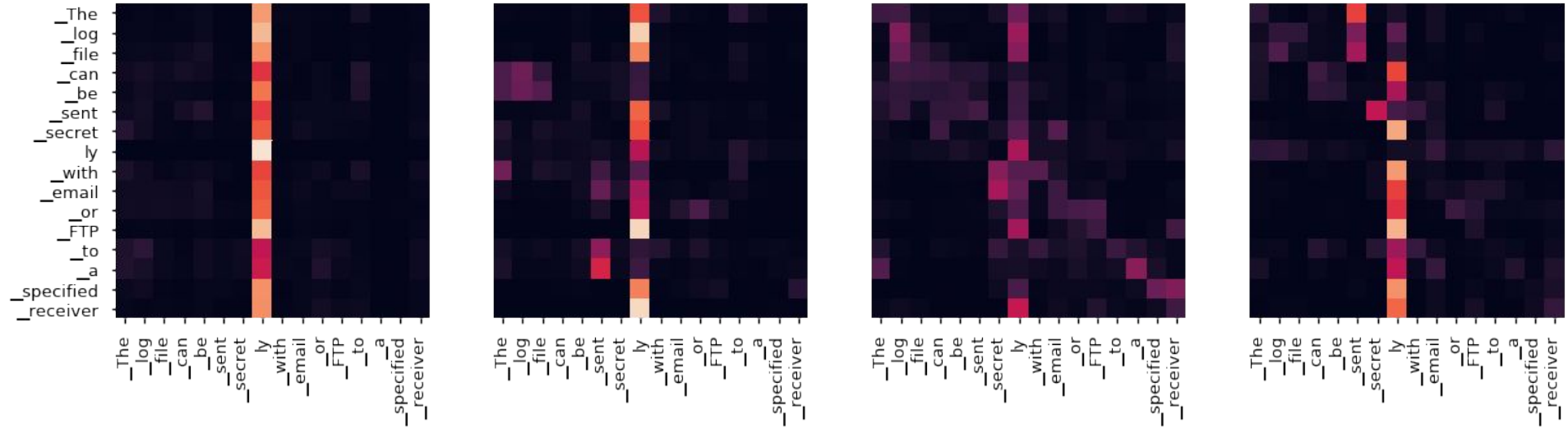




$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^{\text{T}}}{\sqrt{d_k}}\right)V$$

Тепловые карты Self-Attention

$$\text{softmax}\left(\frac{QK^{\mathrm{T}}}{\sqrt{d_k}}\right)$$



Thinking Like Transformers

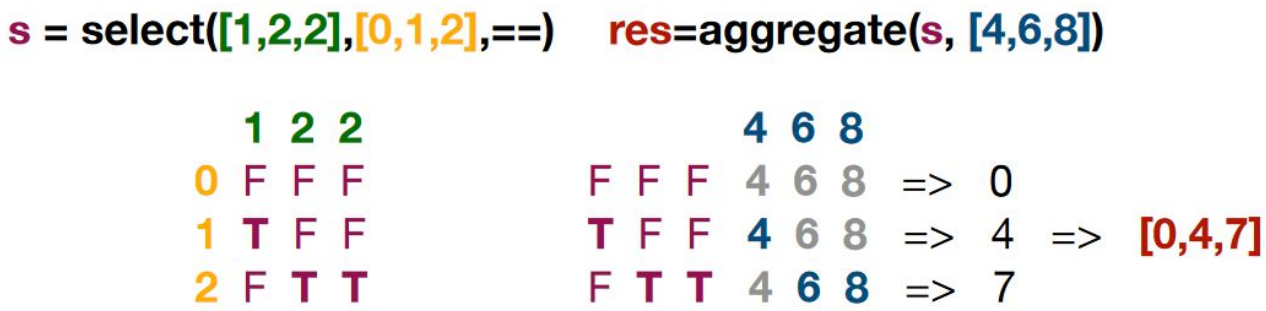
- Предлагается простая модель для кодировщика трансформера
- Язык программирования RASP (Restricted Access Sequence Processing Language)
- Абстракция: вместо слоёв нейронной сети -- операции над последовательностями
- С помощью решающей задачу программы можно построить и обучить трансформер для той же задачи
- Нельзя запрограммировать перевод и другие сложные задачи

Язык RASP

- Вход: (последовательность длины n, range(n))
- Для входа длины n оперирует последовательностями длины n и бинарными матрицами размера n x n, выход -- последовательность
- Константа -- последовательность длины n, заполненная одним и тем же числом
- 3 типа операций:
  - Вспомогательные: length = [n] * n
  - Моделирующие линейные слои и активации: поэлементные (+, + const, pow(,const), >, *, …)
  - Моделирующие механизм внимания

Язык RASP. Моделирование внимания

- Все матрицы -- бинарные!
- **select:** (последовательности k, q; предикат p) -> матрица S: $S_{[i][j]} = p(k_{[i]}, q_{[j]})$
- **aggregate**: (матрица S; последовательность v) -> последовательность s, где i-ый элемент -- среднее элементов, выбранных из v i-ой строкой матрицы S
- **selector_width**: (матрица S) -> последовательность s, где i-ый элемент -- число выбранных элементов в i-ой строке матрицы S
- Булева логика для бинарных матриц



Язык RASP

- Ограничения:
  - Нет циклов: обычный трансформер не может повторять операции произвольное число раз
  - В select выбор всегда взаимный -- как в механизме внимания
- Программа, разворачивающая строку:

```
1  reverse = aggregate(
2      select(indices,
3             length-indices-1,==)
4      tokens );
```

Язык RASP. Связь с трансформером

- Вспомогательные операции:
  - подаваемые на вход индексы range(n) -- позиционное кодирование
  - длину можно посчитать с помощью **aggregate** и **select**
- Операции, моделирующие линейные слои и активации: нет ограничений на операции, любая функция приближается нейросетью глубины 2 с сигмоидой
- Операции, моделирующие внимание:
  - **select** -- матрица внимания, аналогия $\mathrm{softmax}\left(\frac{QK^{\mathrm{T}}}{\sqrt{d_k}}\right)$
  - **aggregate** -- применение внимания, аналогия $\mathrm{softmax}\left(\frac{QK^{\mathrm{T}}}{\sqrt{d_k}}\right)V$

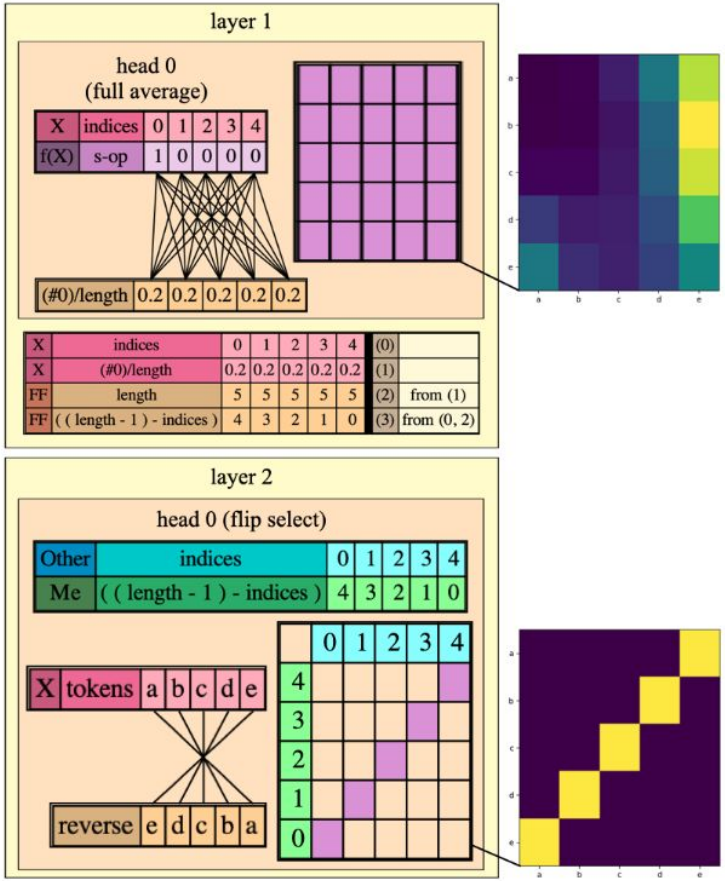
Язык RASP. Трансформер

- Отслеживается граф вычислений, по нему определяется порядок слоёв
- 1 **aggregate** == 1 голова внимания
- Поэлементные операции превращаются в линейный слой с активацией
- При обучении в трансформер добавляется attention supervision: считается MSE между находящейся внутри трансформера и полученной из программы на RASP тепловой картой внимания (attention heatmap)

Эксперименты

- Пример -- программа, разворачивающая строку
- 3 набора экспериментов:
  - Верна ли полученная из программы верхняя граница на число слоёв и голов
  - Насколько она точна
  - Использование attention supervision

```
1  opp_index = length - indices - 1;
2  flip = select(indices, opp_index,==);
3  reverse = aggregate(flip, tokens);
```



Эксперименты. 1 набор

Верна ли полученная из программы верхняя граница на число слоёв и голов

Language	Layers	Heads	Test Acc.	Attn. Matches?
Reverse	2	1	99.99%	✓✗
Hist BOS	1	1	100%	✓
Hist no BOS	1	2	99.97%	✓✗
Double Hist	2	2	99.58%	✓✗
Sort	2	1	99.96%	✗
Most Freq	3	2	95.99%	✗
Dyck-1 PTF	2	1	99.67%	✓✗
Dyck-2 PTF [8]	3	1	99.85%	✗

Эксперименты. 2 набор

Насколько полученная оценка точна; L -- число слоёв, H -- число голов

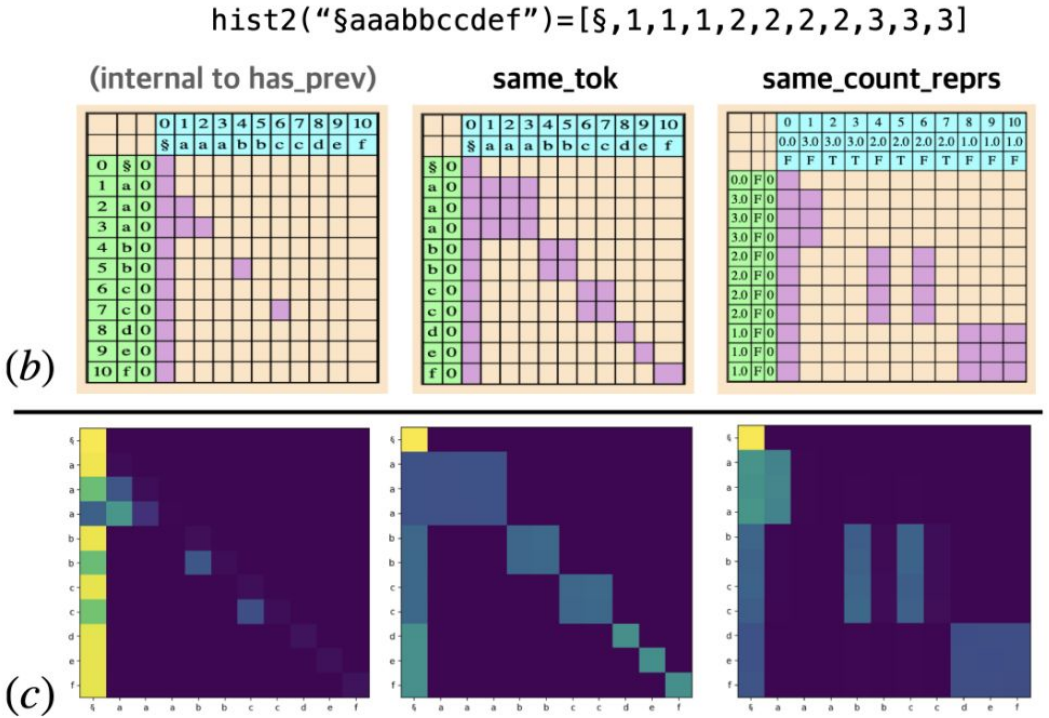
Language	RASP	Average test accuracy (%) with...			
	L, H	L, H	$H-1$	$L-1$	$L-1, 2H$
Reverse	2, 1	**99.9**	-	23.1	41.2
Hist	1, 2	**99.9**	91.9	-	-
2-Hist	2, 2	**99.0**	73.5	40.5	83.5
Sort	2, 1	99.8	-	99.0	**99.9**
Most Freq	3, 2	**93.9**	92.1	84.0	90.2
Dyck-1	2, 1	**99.3**	-	96.9	96.4
Dyck-2	3, 1	**99.7**	-	98.8	94.1

Эксперименты. 3 набор

Использование attention supervision

```
1  same_tok = select(tokens,tokens,==);
2  hist = selector_width(
3              same_tok,
4              assume_bos = True);
5
6  first = not has_prev(tokens);
7  same_count = select(hist,hist,==);
8  same_count_reprs = same_count and
9      select(first,True,==);
10
11 hist2 = selector_width(
12             same_count_reprs,
13             assume_bos = True);
```

(a)



Следствия и новая интуиция о трансформерах

- Ограничение внимания: если внимание использует меньше n log(n) операций (где n -- длина входа), то невозможно обучить трансформер на сортировку
- Порядок слоёв: лучше, если сначала будет внимание, а потом линейные слои с активациями
- k-язык Дика (скобочная последовательность с k видами скобок): задача решается трансформером с фиксированным числом слоёв и голов для любого k
- Решение логических задач: можно написать код на RASP и посмотреть на получившуюся структуру

Thinking Like Transformers

Gail Weiss, Yoav Goldberg, Eran Yahav

From the depths of Hell itself: **RASP**

★ ★ ★ ★ ★

"like Matlab, but made by Satan" - Omri Gilad

The *R*estricted *A*ccess *S*equence *P*rocessing Language

- Think in Symbolic Code!
- Effortlessly follow the information flow constraints of a Transformer!
  (You don't have a choice!)
- Analyse your programs for number of layers and heads they need!

all in exchange for…

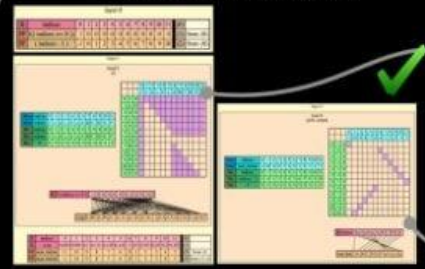
~~Your soul~~ Nothing, it's free online: github.com/tech-srl/RASP

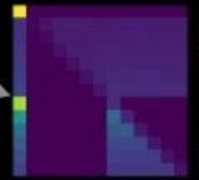
Solve a task in RASP!

```
def with_bos_selector_width(s) {
    s = s or select(indices,0,==);
    inverted_width =
        aggregate(s,indicator(indices==0));
    return round((1/inverted_width)-1);
}
```

```
earlier = select(tokens,tokens,<) or
          (select(tokens,tokens,==) and
           select(indices,indices,<));
num_smaller = with_bos_selector_width(earlier);
target_pos = 0 if (indices == 0) else
             num_smaller + 1;
sel_new_val = select(target_pos,indices,==);
sort = aggregate(sel_new_val,tokens);
```

Compile it to a transformer-encoder architecture!





Bully an actual transformer into realising your solution!

RASP Primitives

All examples on input *"RASP"*

Base Sequences

```
tokens  = [R,A,S,P]
indices = [0,1,2,3]
length  = [4,4,4,4]
```

Elementwise Operations

```
indices+2 = [2,3,4,5]
indices*length = [0,4,8,12]
tokens if indices%2==1 else "a" = [a,A,a,P]
```

Non-Elementwise Operations

```
flip = select([3,2,1,0],[0,1,2,3],==)      reverse=aggregate(flip, [R,A,S,P])
   3 2 1 0                                    R A S P
0  F F F T                                    F F F T  R A S P  => P
1  F F T T                                    F F T T  R A S P  => S  => [P,S,A,R]
2  F T T T                                    F T T T  R A S P  => A
3  T T T T                                    T T T T  R A S P  => R
```

(selector_width)

```
selector_width(flip) = [1,1,1,1]
selector_width(select(indices,indices,<=)) = [1,2,3,4]
histogram = selector_width(select(tokens,tokens,==))
   on input "Hello": histogram = [1,1,2,2,1]
```

Never once worry about what weights your solution requires in practice!

Рецензия

Сильные стороны

- Предложена новая интересная идея понимания трансформеров с помощью языка программирования
- Подробное описание RASP, примеры решаемых задач
- В статье делается попытка установить связь между операциями на языке RASP и трансформерах

Слабые стороны

- Язык подходит для кодирования только части задач
- Введение нового языка программирования не обосновано
- Нет заявленной связи с матрицами внимания
- Связь с матрицами внимания экспериментально не подтверждена
- Качество оценки RASP на количество слоев/attention heads не вполне подтверждается экспериментами

Воспроизводимость

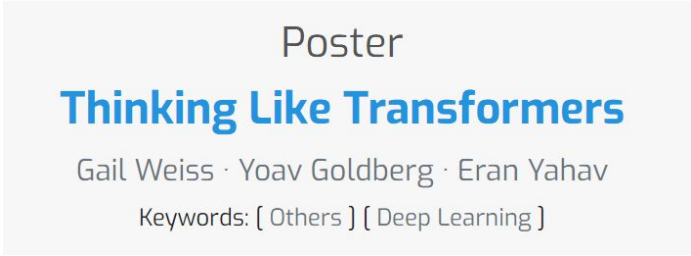
- Есть репозитории с интерпретатором RASP[1] и с кодом экспериментов [2], примеры кода на RASP
- Даны значения параметров для экспериментов
- Есть только описание “компиляции” языка в архитектуру трансформера

[1] https://github.com/tech-srl/RASP
[2] https://github.com/tech-srl/RASP-exps

Контекст работы

Информация о публикации

- Статья опубликована 19.07.21
- Статья была представлена на ICML 21 в виде постера
- Изначально статья подавалась на ICLR2021 (первая версия статьи датируется 28.09.20)



Информация об авторах

- Gail Weiss (PhD студент в Технионе), предыдущие работы посвящены конечным автоматам и формальным иерархиям над RNN и формальными языками
- Yoav Golderg (профессор в Университете Бар Илан), научный руководитель Weiss
- Eran Yahav (профессор в Технионе), научный руководитель Weiss

На что опирается работа

- Работа ключевым образом основывается на научных интересах Weiss и ее предыдущих работах
- Есть работы, которые показывают, что поставленные в данной статье перед авторами задачи могут быть решены с теоретической точки зрения, как, например, [1], [2] и [3]

[1] On the Ability and Limitations of Transformers to Recognize Formal Languages, Satwik Bhattamishra, Kabir Ahuja, Navin Goyal
[2] Are Transformers Universal Approximators Of Sequence-To-Sequence Functions? Chulhee Yun, Srinadh Bhojanapalli, Ankit Singh Rawat, Sashank J. Reddi, Sanjiv Kumar
[3] Attention is Turing Complete, Jorge P´erez, Pablo Barcel´o. Javier Marinkovic.

Цитирование

На данную статью ссылаются только 3 другие работы:

- On the Power of Saturated Transformers: A View from Circuit Complexity William Merrill, Y.Goldberg, R. Schwartz, N. Smith
- The Neural Data Router: Adaptive Control Flow in Transformers Improves Systematic Generalization. Róbert Csordás, Kazuki Irie, Jürgen Schmidhuber
- Learning Adaptive Control Flow in Transformers for Improved Systematic Generalization. Róbert Csordás, Kazuki Irie, Jürgen Schmidhuber

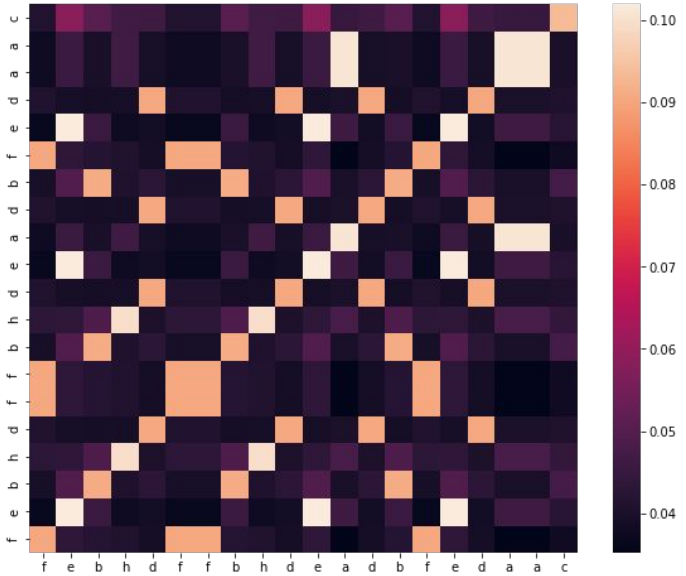
Хакерство

Репликация

- Код авторов очень сложный для прочтения
- При запуске с параметрами из статьи результаты воспроизводятся
- К оценкам на размеры трансформера из статьи возникли вопросы...

Задача о гистограммах

- В статье рассмотрено две вариации — со спецтокенами и без
- Утверждается, что без спецтокенов требуется две головы
- Идеально обучить трансформер с одной головой можно
- Выводы о роли спецтокенов не аргументированы



Language	RASP	Average test accuracy (%) with...			
	L, H	L, H	$H-1$	$L-1$	$L-1, 2H$
Hist	$1, 2$	**99.9**	91.9	-	-

Подсчет бóльших токенов

- Достаточно одного слоя и одной головы
- Карта внимания совпадает с ожиданиями: большие токены имеют большие веса