Masked Hard-Attention Transformers and Boolean RASP Recognize Exactly the Star-Free Languages

Dana Angluin, David Chiang, Andy Yang

How’s My LISP Syntax?

```
> (defun even(num) (= (mod num 2) 0))
> (filter '(6 4 3 5 2) #'even)
> (6 4 2)
```

How’s My LISP Syntax?

```
(defun even(num)(= (mod num 2) 0))
(filter '(6 4 3 5 2) #'even)
```

hello

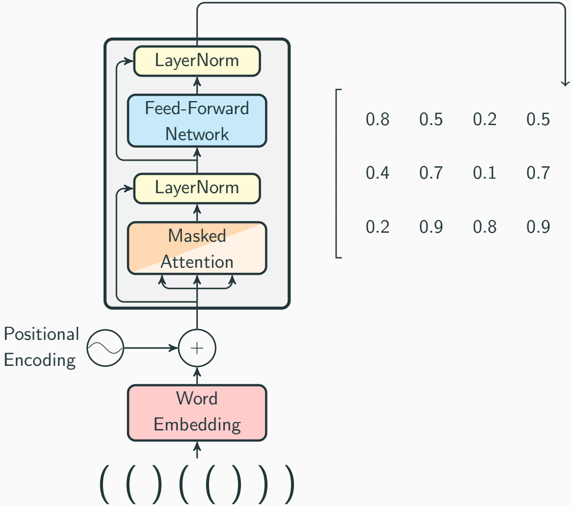
How's My LISP Syntax?

(()(()))

(())

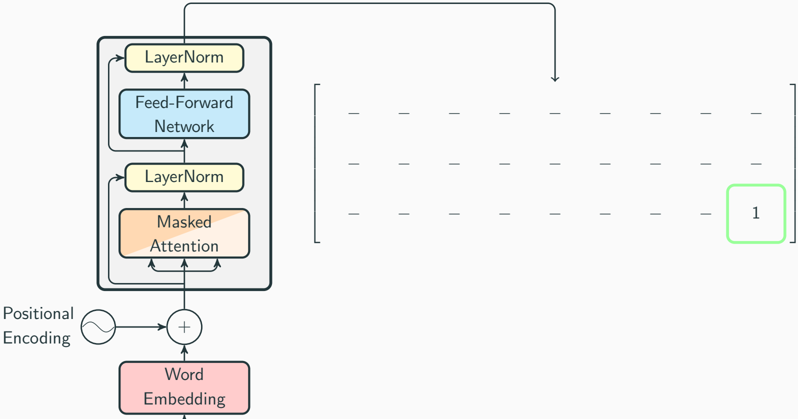
hello

Transformer Encoders



$$\begin{bmatrix} 0.8 & 0.5 & 0.2 & 0.5 & 0.2 & 0.1 & 0.6 & 0.2 \\ 0.4 & 0.7 & 0.1 & 0.7 & 0.5 & 0.8 & 0.3 & 0.9 \\ 0.2 & 0.9 & 0.8 & 0.9 & 0.7 & 0.4 & 0.7 & 0.8 \end{bmatrix}$$

Transformer Encoders as Formal Language Recognizers



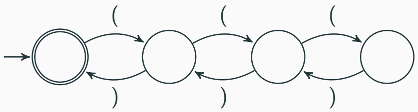
(( ) (( ))) · EOS

Bounded-Depth Dyck Language

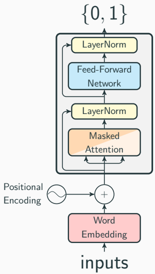
Dyck-1 of depth 3
= strings of parentheses, balanced and nested up to 3 deep
= strings where the number of ('s is equal to the number of)'s, and every prefix contains 0–3 more ('s than)'s

Examples:

- accepted: ()() ✓
- accepted: (())() ✓
- accepted: (()(())) ✓
- rejected: (((()))) ✗
- rejected: ()()(✗



The Big Picture: Expressivity and Logic



$\longleftrightarrow$

$$(\forall i)(Q_{\mathrm{a}})$$
$$(\forall i)(\forall j)(i < j \rightarrow Q_{\mathrm{a}}(i) \wedge Q_{\mathrm{b}}(j))$$
etc.

What languages are recognized by transformer encoders?

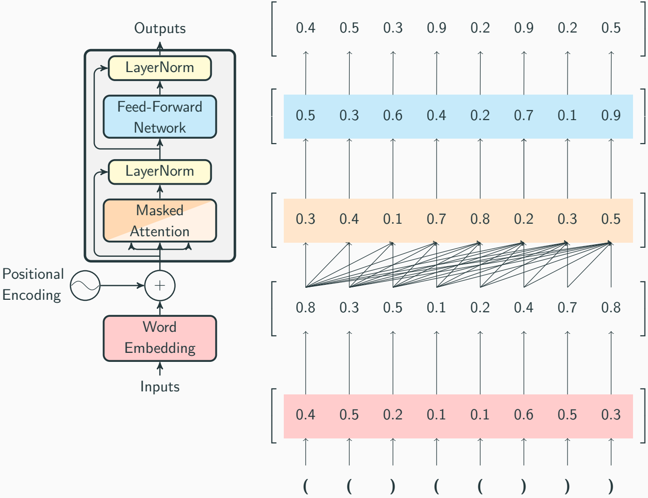
What languages are defined by logical formulas?

For a survey of papers in this area (including this one): Strobl et al. [4], "Transformers as Recognizers of Formal Languages: A Survey on Expressivity"

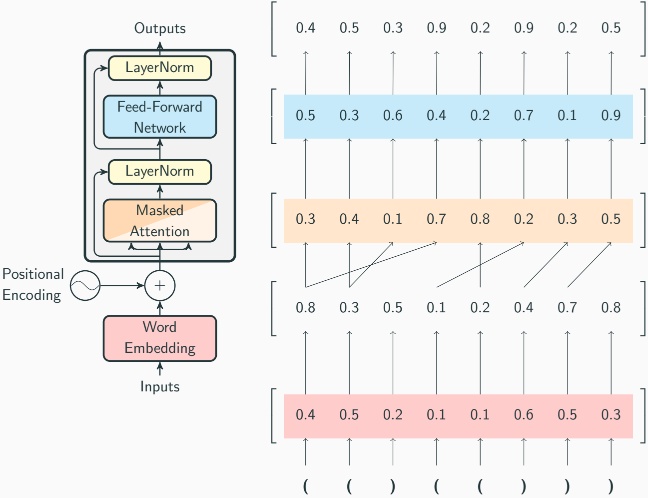
Questions to Consider

- Expressivity – what can transformers do under perfect conditions?
- Learnability – what can transformers learn to do in real life?
- Interpretability – how can we know what transformers have learned?
- Improvements – how can we augment the architecture?

Standard Attention



Hard Attention Simplifies Our Analysis

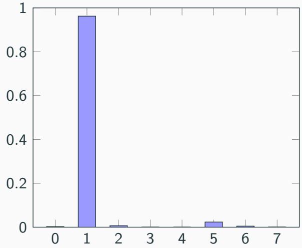


More on Hard Attention

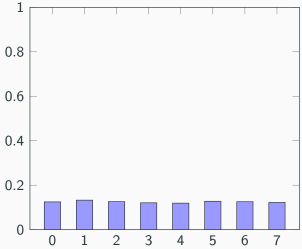
We can think of SOFTMAX as parameterized with a "temperature". At low temperature, it closely approximates ARGMAX - selecting a single input – which is much easier to manage!

$$\textsc{softmax}(x_i, \tau) = \frac{e^{x_i/\tau}}{\sum_{j=0}^{n} e^{x_j/\tau}}$$

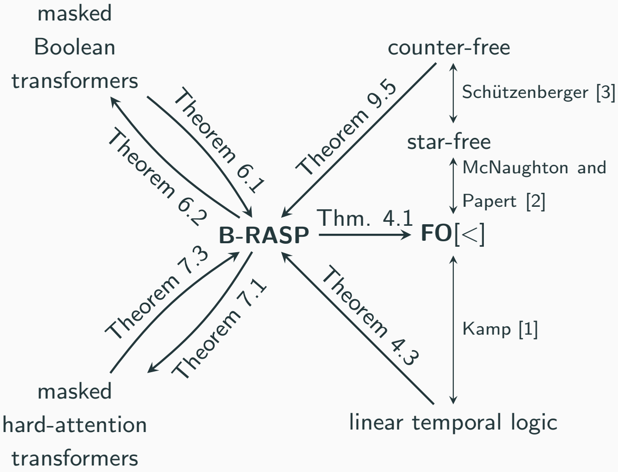
Low temperature $\tau \to 0$



High temperature $\tau \to \infty$



Our Results



masked Boolean transformers
counter-free
Schützenberger [3]
star-free
McNaughton and Papert [2]
Theorem 6.1
Theorem 6.2
Theorem 9.5
B-RASP
Thm. 4.1
FO[<]
Theorem 7.3
Theorem 7.1
Theorem 4.3
Kamp [1]
masked hard-attention transformers
linear temporal logic

Star-Free Languages

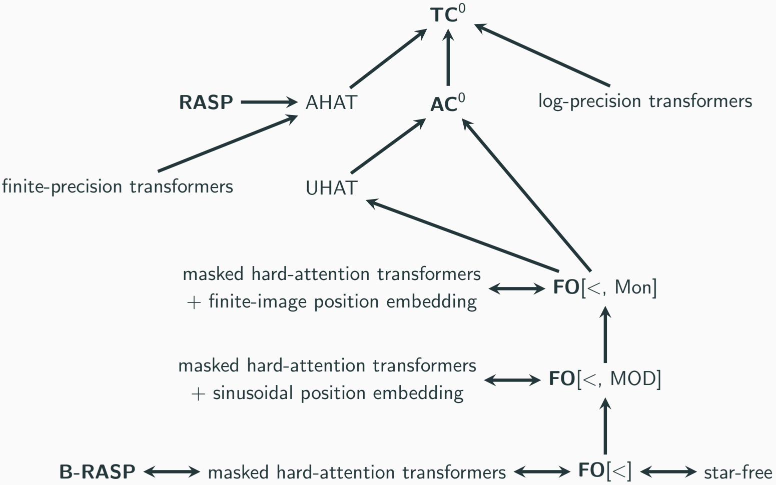
Formula	Language
Dyck-1 of Depth 3	$(), (()), ((())), ()(), (())(), \dots$
$(ab)^*$	$\epsilon, ab, abab, ababab, \dots$
$\overline{\Sigma^* aa \Sigma^*}$	$\epsilon, a, ab, ba, bb, abb, bab, bba, bbb, \dots$
$a\Sigma^* b$	$ab, aab, abb, aaab, aabb, \dots$

FO[<] and Star-Free

Using a theorem of McNaughton and Papert [2], the Star-Free languages are exactly those described by First-Order Logic

Formula	Language
$\exists x. Q_1(x)$	Strings containing a 1
$\forall x. \neg Q_0(x)$	Strings not containing any 0's
$\exists x. Q_0(x) \wedge \forall y. y > x$	String's starting with a 0
$\exists x. \exists y. \exists z. x < y < z$ $\wedge Q_0(x) \wedge Q_0(y) \wedge Q_1(z)$	String's containing the subsequence 001

Contextualizing Our Results

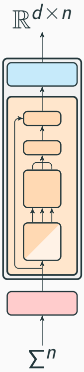


Thank You

Stephen Bothwell, Darcey Riley, Ken Sible,

Aarohi Srivastava, Lena Strobl, and Chihiro Taguchi!

Questions?



- With some generous assumptions, we can prove what transformers are capable of perfect conditions!
- Masked hard-attention transformer as a "base case" to build upon
- Very rich computational equivalences
- Ultimate goal: understand rigorously the capabilities and limitations of transformers

References

[1] Johan Anthony Willem Kamp. *Tense Logic and the Theory of Linear Order*. PhD thesis, University of California, Los Angeles, 1968. URL https://www.proquest.com/docview/302320357.

[2] Robert McNaughton and Seymour Papert. *Counter-Free Automata*. Number 65 in M.I.T. Press Research Monographs. The M.I.T. Press, 1971. ISBN 9780262130769. URL https://archive.org/embed/CounterFre_00_McNa.

[3] M.P. Schützenberger. On finite monoids having only trivial subgroups. *Information and Control*, 8(2):190–194, 1965. DOI 10.1016/S0019-9958(65)90108-7.

[4] Lena Strobl, William Merrill, Gail Weiss, David Chiang, and Dana Angluin. Transformers as recognizers of formal languages: A survey on expressivity. *arXiv preprint arXiv:2311.00208*, 2023.