Characterizing the Expressivity of Local Attention in Transformers

Jiaoda Li **Ryan Cotterell**
{jiaoda.li, ryan.cotterell}@inf.ethz.ch
ETH zürich

Abstract

The transformer is the most popular neural architecture for language modeling. The cornerstone of the transformer is its global attention mechanism, which lets the model aggregate information from all preceding tokens before generating the next token. One common variant of attention is called local attention, which restricts each token to aggregating information from a bounded window of predecessors, reducing the quadratic cost of global attention to linear. Although this restriction is usually motivated by efficiency, it has also been found to improve model quality, a phenomenon that has so far lacked a satisfactory explanation. We provide a formal account of this phenomenon in terms of recognizer expressivity. It has been shown that fixed-precision transformers with global attention correspond to a fragment of linear temporal logic containing a single past operator. We additionally prove that adding local attention introduces a second temporal operator, strictly enlarging the class of recognizable regular languages. Moreover, global and local attention are expressively complementary: neither subsumes the other, and combining them yields the richest fragment. Experiments on formal language recognition and natural language modeling corroborate the theory, showing that hybrid global–local transformers outperform their global-only counterparts.

1 Introduction

The transformer (Vaswani et al., 2017) is the dominant neural architecture in modern language modeling (Radford et al., 2018, 2019; Brown et al., 2020; OpenAI, 2023). At its core lies the attention mechanism (Schmidhuber, 1992; Bahdanau et al., 2015), which allows the model to aggregate information from all previously generated tokens before generating the next one. *Local attention* (Luong et al., 2015; Child et al., 2019; Beltagy et al., 2020; Zaheer et al., 2020) is a popular attention variant that restricts the model to aggregating over a fixed-size window of preceding tokens.

Local attention is typically motivated by efficiency. Standard *global attention* aggregates over *all* previous positions, incurring quadratic cost in sequence length; local attention reduces this to linear. At first glance, restricting the range of tokens that a transformer can attend to appears to be a strict weakening—a trade-off of expressivity for speed. Surprisingly, however, local attention has been repeatedly observed to yield empirical improvements, both in machine translation (Luong et al., 2015) and in language modeling (Child et al., 2019; Beltagy et al., 2020; Zaheer et al., 2020)—even when computational budgets are held constant. We resolve this puzzle by analyzing which formal languages different attention patterns can recognize, a standard lens in the formal study of neural models (Delétang et al., 2023; Butoi et al., 2025). This lens also has practical relevance: modern language models (OpenAI, 2023; DeepSeek-AI, 2025; Team et al., 2025) are routinely used to perform computation via prompting (Wei et al., 2022; Li et al., 2024; Merrill and Sabharwal, 2024). In this vein, Li and Cotterell (2025) show that fixed-precision transformer language models with global attention correspond exactly to the fragment of linear temporal logic with a single past operator. Expanding their work, we prove that adding local attention introduces a new temporal operator, enabling the recognition of a strictly larger class of regular languages. These include the locally testable languages, a family closely connected to n-gram models and extensively studied in formal language theory (McNaughton and Papert, 1971, Ch. 2). Crucially, local attention is complementary rather than a drop-in replacement for global attention: with purely local attention, the computation depends only on a fixed-length suffix of the input, collapsing expressivity to the smaller class of definite languages (Kleene, 1956, §5).

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 37485–37507

Empirically, we test these predictions on formal language recognition with length generalization. We compare transformers with (i) global attention, (ii) local attention, and (iii) a hybrid that mixes global and local attention. The results align with the theory: hybrid models recognize strictly more languages than either extreme and generalize more reliably to longer sequences. Varying local attention's window size further shows that a window of size one is consistently the strongest choice in the local-attention family, both for purely local models and for hybrid models. Finally, we evaluate models equipped with two popular positional encodings—sinusoidal (`SiPE`) and rotary (`RoPE`)—and find that neither compensates for the absence of local attention. We further show that these findings extend to natural language: on WikiText-2, hybrid attention with a window of size one consistently achieves the lowest perplexity across positional encodings.

2 Linear Temporal Logic

In this section, we introduce linear temporal logic, the main tool we use to characterize the expressive power of transformers under different types of attention. We examine the fragments relevant to our analysis, prove the separations between them, and connect them to classical characterizations from algebraic formal language theory.

2.1 Preliminaries

We now introduce **linear temporal logic** (LTL; Kamp, 1968; Pnueli, 1977), which has become a popular tool in the analysis of transformers (Yang et al., 2024, 2026; Li and Cotterell, 2025).

Tokens, Strings and Languages. An **alphabet** is a finite, non-empty set of **tokens**. Fix an alphabet Σ. A **string** over Σ is a finite sequence of tokens drawn from Σ. The **length** of a string $\mathtt{w} = \mathtt{w}_1 \cdots \mathtt{w}_N$, denoted $|\mathtt{w}| = N$, is the number of tokens it contains. We write Σ^* for the set of all strings over Σ, and call any subset $L \subseteq \Sigma^*$ a **language**.

Linear Temporal Logic. The syntax and semantics of linear temporal logic ($\mathbf{LTL}[\mathbf{P}, \mathbf{Y}, \mathbf{S}, \mathbf{U}]$) as well as the definition of operator depth are recalled in §A. We also give a definition of what it means for a formula in LTL to define a language; see §A.3. Given a formula ψ, we write $\mathbb{L}(\psi)$ for the set of all strings satisfying ψ.

Fragments of $\mathbf{LTL}[\mathbf{P}, \mathbf{Y}, \mathbf{S}, \mathbf{U}]$. For a set of temporal operators $\mathcal{O} \subseteq \{\mathbf{P}, \mathbf{Y}, \mathbf{S}, \mathbf{U}\}$, we write $\mathbf{LTL}[\mathcal{O}]$ for the corresponding fragment in which only operators from $\mathcal{O}$ are allowed.

Definition 2.1. *Let $\mathcal{O} \subseteq \{\mathbf{P}, \mathbf{Y}, \mathbf{S}, \mathbf{U}\}$ be a set of temporal operators. A language L is **definable** by $\mathbf{LTL}[\mathcal{O}]$ if there exists a formula ψ in $\mathbf{LTL}[\mathcal{O}]$ such that $\mathbb{L}(\psi) = L$.*

Gabbay et al. (1980) show that every language definable by $\mathbf{LTL}[\mathbf{P}, \mathbf{Y}, \mathbf{S}, \mathbf{U}]$ is also definable by both $\mathbf{LTL}[\mathbf{S}]$ and $\mathbf{LTL}[\mathbf{U}]$; in the paper, we write $\mathbf{LTL}[\mathbf{S}]$ for full linear temporal logic. The fragments $\mathbf{LTL}\,[\mathbf{P}]$ and $\mathbf{LTL}\,[\mathbf{Y}]$ are strictly less expressive, i.e., there exist languages definable by $\mathbf{LTL}[\mathbf{S}]$ that are not definable by either $\mathbf{LTL}\,[\mathbf{P}]$ or $\mathbf{LTL}\,[\mathbf{Y}]$.

2.2 Characterizing $\mathbf{LTL}\,[\mathbf{Y}]$

We first characterize the expressive power of $\mathbf{LTL}\,[\mathbf{Y}]$ by relating it to a classical class of regular languages. Because formulas in $\mathbf{LTL}\,[\mathbf{Y}]$ can only look back a bounded number of steps, we show they naturally correspond to languages determined by a bounded suffix—the definite languages.

For a string $\mathtt{w}$ of length N and an integer $m \leq N$, an **m-factor** of $\mathtt{w}$ is any contiguous substring of $\mathtt{w}$ of length m. The **m-prefix** of $\mathtt{w}$ is its initial substring of length m, and the **m-suffix** of $\mathtt{w}$ is its final substring of length m. We write $\mathbb{I}_m(\mathtt{w})$ for the set of all m-factors of $\mathtt{w}$, and $P_m(\mathtt{w})$ and $S_m(\mathtt{w})$ for the m-prefix and m-suffix of $\mathtt{w}$, respectively. If $m > |\mathtt{w}|$, we set $P_m(\mathtt{w}) = S_m(\mathtt{w}) = \mathtt{w}$. Formally, a language $L \subseteq \Sigma^*$ is **m-definite**, for some $m \geq 0$, if L is a Boolean combination of languages of the form $\{\mathtt{w} \in \Sigma^* \mid S_m(\mathtt{w}) = \mathtt{u}\}$. A language is **definite** if it is m-definite for some $m \geq 0$.

We next relate $\mathbf{LTL}\,[\mathbf{Y}]$ to definite languages, for which we also provide algebraic and automata-theoretic characterizations. The relevant background, including definitions of the syntactic semigroups and deterministic finite automata (DFAs), are recalled in §B.

Theorem 2.2. *Let $L \subseteq \Sigma^*$ be a regular language, let $\mathbb{S}$ be its syntactic semigroup, and let $\mathcal{A}$ be the minimal DFA recognizing L. The following are equivalent:*

1. *L is definable in $\mathbf{LTL}\,[\mathbf{Y}]$;*
2. *L is definite;*
3. *every idempotent $e \in \mathbb{S}$ is a right zero, i.e., for all $s \in \mathbb{S}$, $se = e$;*
4. *every transformation induced by a non-empty string is non-permutational.*

Proof. See §C.1. ∎

2.3 LTL [P] and LTL [Y] are Incomparable

Li and Cotterell (2025) give a rich characterization of LTL [P]. Building on this and §2.2, we now relate LTL [P] and LTL [Y]. However, we first discuss what it means to compare two sets of operators for linear temporal logic. Fix $\mathcal{O}_1, \mathcal{O}_2 \subseteq \{\mathbf{P}, \mathbf{Y}, \mathbf{S}, \mathbf{U}\}$. We say $\mathbf{LTL}[\mathcal{O}_1]$ is more expressive than $\mathbf{LTL}[\mathcal{O}_2]$, written in symbols $\mathbf{LTL}[\mathcal{O}_2] \subseteq \mathbf{LTL}[\mathcal{O}_1]$, if every language definable in $\mathbf{LTL}[\mathcal{O}_2]$ is also definable in $\mathbf{LTL}[\mathcal{O}_1]$. We say $\mathbf{LTL}[\mathcal{O}_1]$ is *strictly* more expressive than $\mathbf{LTL}[\mathcal{O}_2]$, written in symbols as $\mathbf{LTL}[\mathcal{O}_2] \subset \mathbf{LTL}[\mathcal{O}_1]$, if $\mathbf{LTL}[\mathcal{O}_2] \subseteq \mathbf{LTL}[\mathcal{O}_1]$ and there exists a language definable in $\mathbf{LTL}[\mathcal{O}_1]$ but not in $\mathbf{LTL}[\mathcal{O}_2]$. We call $\mathbf{LTL}[\mathcal{O}_1]$ and $\mathbf{LTL}[\mathcal{O}_2]$ **incomparable** if neither is more expressive than the other.

Proposition 2.3. *There exists a language definable in* LTL [P] *that is not definable in* LTL [Y].

Proof. Consider the language $\mathtt{a}\Sigma^*$, i.e., the set of strings whose first token is $\mathtt{a}$. It is definable in LTL [P] by $\mathbf{P}(\pi_{\mathtt{a}} \wedge \neg\mathbf{P}\top)$, i.e., $\mathbb{L}(\mathbf{P}(\pi_{\mathtt{a}} \wedge \neg\mathbf{P}\top)) = \mathtt{a}\Sigma^*$. However, it is not definite, and therefore not definable in LTL [Y] by Theorem 2.2. ■

Proposition 2.4. *There exists a language definable in* LTL [Y] *that is not definable in* LTL [P].

Proof. Consider the language $\Sigma^*\mathtt{a}$, i.e., the set of strings whose last token is $\mathtt{a}$. It is definable in LTL [Y] by $\mathbf{Y}\pi_{\mathtt{a}}$, i.e., $\mathbb{L}(\mathbf{Y}\pi_{\mathtt{a}}) = \Sigma^*\mathtt{a}$. However, it is not definable in LTL [P]: Li and Cotterell (2025) show that LTL [P] defines exactly the class of left-deterministic polynomials, and $\Sigma^*\mathtt{a}$ is not a left-deterministic polynomial. See §B for an exposition of the relevant technical definitions. ■

Corollary 2.5. *The fragments* LTL [P] *and* LTL [Y] *are incomparable.*

Proof. This is a direct implication of both Propositions 2.3 and 2.4. ■

2.4 Characterizing LTL [P, Y]

We give another implication of Corollary 2.5 below.

Corollary 2.6. $\mathbf{LTL}[\mathbf{Y}], \mathbf{LTL}[\mathbf{P}] \subset \mathbf{LTL}[\mathbf{P}, \mathbf{Y}]$.

Proof. Note that LTL [P, Y] trivially contains both P and Y; it subsumes both fragments. However, because LTL [P] and LTL [Y] are incomparable by Corollary 2.5, there exists a language definable in LTL [P, Y] that is not definable in LTL [P] as well as a language definable in LTL [P, Y] that is not definable in LTL [Y]. This renders both subsumptions strict, as we sought to show. ■

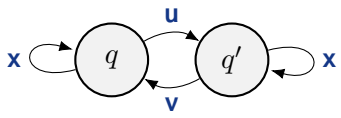


Figure 1: Forbidden configuration in the minimal DFAs of LTL [P, Y]-definable languages.

To further characterize the expressive power of LTL [P, Y], we relate it to standard algebraic and automata-theoretic notions. The relevant definitions, including locally $\mathcal{R}$-trivial semigroups and configurations of DFAs, are recalled in §B.

Theorem 2.7. *Let $L \subseteq \Sigma^*$ be a regular language, $\mathbb{S}$ its syntactic semigroup, and $\mathcal{A}$ the minimal DFA recognizing L. The following are equivalent:*

1. *L is definable in* LTL [P, Y];
2. *$\mathbb{S}$ is locally $\mathcal{R}$-trivial;*
3. *$\mathcal{A}$ contains no configuration of the form shown in Fig. 1.*

Proof. See §C.2. ■

Next, we give two examples that respectively illustrate a language definable in LTL [P, Y] and a language beyond its expressive power.

2.4.1 The Locally Testable Languages

The addition of Y enables the definition of classes of languages that are linguistically relevant yet lie beyond the expressive power of LTL [P] alone. As a representative example, we consider the locally testable languages.

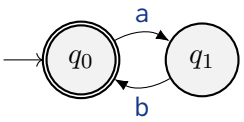
A language $L \subseteq \Sigma^*$ is **locally m-testable**, for some $m > 0$, if it is a Boolean combination of languages of the following three forms:

$$\{\mathtt{w} \in \Sigma^* \mid \mathtt{u} \in \mathbb{I}_m(\mathtt{w})\}, \qquad \mathtt{u} \in \Sigma^m, \tag{1a}$$

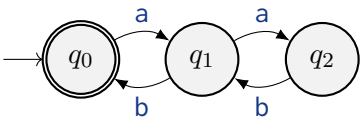
$$\{\mathtt{w} \in \Sigma^* \mid P_{m-1}(\mathtt{w}) = \mathtt{u}\}, \qquad \mathtt{u} \in \Sigma^{m-1}, \tag{1b}$$

$$\{\mathtt{w} \in \Sigma^* \mid S_{m-1}(\mathtt{w}) = \mathtt{u}\}, \qquad \mathtt{u} \in \Sigma^{m-1}. \tag{1c}$$

A language is **locally testable** if it is locally m-testable for some $m > 0$. Li and Cotterell (2025) show that LTL [P] cannot define many simple locally testable languages, e.g., $\Sigma^*\mathtt{a}$ and $\Sigma^*\mathtt{ab}\Sigma^*$, because they are not right-deterministic. In contrast, as we show in the subsequent theorem, all locally testable languages are definable in LTL [P, Y], yielding a witness to the separation in expressive power between LTL [P] and LTL [P, Y].



(a) $(\mathsf{ab})^*$



(b) $(\mathsf{a}(\mathsf{ab})^*\mathsf{b})^*$

Figure 2: Minimal DFAs. Nodes represent states and arrows represent transitions. The initial state is indicated by an incoming arrow with no source node, and accepting states are shown with double circles.

Theorem 2.8. *Any locally testable language is definable in* $\mathbf{LTL}[\mathbf{P}, \mathbf{Y}]$.

Proof. See §C.3. ■

2.4.2 A Bounded Dyck Language

Despite this gain in expressivity, $\mathbf{LTL}[\mathbf{P}, \mathbf{Y}]$ still defines a limited class of languages. For instance, it does not define all star-free languages, which are exactly the languages definable in $\mathbf{LTL}[\mathbf{S}]$. One illustrative family is the bounded Dyck languages, consisting of well-balanced parentheses with a fixed upper bound on nesting depth.[1] The bounded Dyck language of depth k consists of all well-balanced strings whose maximum nesting depth is at most k.[2] For example, over $\{\mathsf{a}, \mathsf{b}\}$, where a is an opening parenthesis and b is a closing parenthesis, the languages

$$(\mathsf{ab})^* \qquad \text{and} \qquad (\mathsf{a}(\mathsf{ab})^*\mathsf{b})^* \tag{2}$$

correspond to maximum nesting depth 1 and 2.

The minimal DFAs for these languages are shown in Fig. 2. The depth-1 language $(\mathsf{ab})^*$ is recognizable by $\mathbf{LTL}[\mathbf{P}, \mathbf{Y}]$, because its minimal DFA avoids the forbidden configuration in Fig. 1. In contrast, the depth-2 language $(\mathsf{a}(\mathsf{ab})^*\mathsf{b})^*$ is not $\mathbf{LTL}[\mathbf{P}, \mathbf{Y}]$-definable: its minimal DFA contains the configuration from Fig. 1, witnessed by $q = q_0$, $q' = q_1$, $\mathsf{u} = \mathsf{a}$, $\mathsf{v} = \mathsf{b}$, and $\mathsf{x} = \mathsf{ab}$.

2.5 Characterizing $\mathbf{LTL}\left[\mathbf{P}, \mathbf{Y}^{\leq k}\right]$

We now introduce two additional temporal operators derived from $\mathbf{Y}$. The bounded operator $\mathbf{Y}^{\leq k}$ asks whether ψ holds at some position among the previous k steps, while the unbounded operator $\mathbf{Y}^*$ asks whether ψ holds at *any* earlier position. Formally, these two operators are defined as follows

$$\mathbf{Y}^{\leq k}\psi \stackrel{\text{def}}{=} \bigvee_{i=1}^{k} \mathbf{Y}^i\psi = \bigvee_{i=1}^{k} \underbrace{\mathbf{Y}\mathbf{Y}\cdots\mathbf{Y}}_{i \text{ times}}\psi \tag{3a}$$

$$\mathbf{Y}^{*}\psi \stackrel{\text{def}}{=} \bigvee_{i=1}^{\infty} \mathbf{Y}^i\psi = \bigvee_{i=1}^{\infty} \underbrace{\mathbf{Y}\mathbf{Y}\cdots\mathbf{Y}}_{i \text{ times}}\psi \tag{3b}$$

Note that $\mathbf{Y}$ is the special case $\mathbf{Y}^{\leq 1}$, and $\mathbf{Y}^*$ coincides with $\mathbf{P}$. This last fact implies that $\mathbf{LTL}[\mathbf{P}, \mathbf{Y}^*] = \mathbf{LTL}[\mathbf{P}]$, i.e., the fragments are not only comparable, they are equivalent. However, for any $k \geq 1$, the fragment $\mathbf{LTL}\left[\mathbf{Y}^{\leq k}\right]$, as is the case with $\mathbf{LTL}[\mathbf{Y}]$, is incomparable with $\mathbf{LTL}[\mathbf{P}]$. We make this claim precise in the following theorem.

Theorem 2.9. *For any* $k \geq 1$, *the fragments* $\mathbf{LTL}[\mathbf{P}]$ *and* $\mathbf{LTL}\left[\mathbf{Y}^{\leq k}\right]$ *are incomparable.*

Proof. See §C.4. ■

Thus, we arrive at the following corollary.

Corollary 2.10. *The fragment* $\mathbf{LTL}\left[\mathbf{P}, \mathbf{Y}^{\leq k}\right]$ *is strictly more expressive than each of the fragments* $\mathbf{LTL}[\mathbf{P}]$ *and* $\mathbf{LTL}\left[\mathbf{Y}^{\leq k}\right]$.

Proof. See §C.5. ■

We next analyze expressivity for various $k \geq 1$; we find $\mathbf{Y}^{\leq k}$ is maximally expressive at $k = 1$.

Proposition 2.11. *The fragment* $\mathbf{LTL}\left[\mathbf{Y}^{\leq k}\right]$ *is strictly less expressive than* $\mathbf{LTL}[\mathbf{Y}]$ *for* $k > 1$.

Proof. See §C.6. ■

We now show adding $\mathbf{P}$ preserves this strictness.

Proposition 2.12. *The fragment* $\mathbf{LTL}\left[\mathbf{P}, \mathbf{Y}^{\leq k}\right]$ *is strictly less expressive than* $\mathbf{LTL}[\mathbf{P}, \mathbf{Y}]$ *for* $k > 1$.

Proof. See §C.7. ■

The gap in expressivity between $\mathbf{LTL}\left[\mathbf{P}, \mathbf{Y}^{\leq k}\right]$ and $\mathbf{LTL}[\mathbf{P}, \mathbf{Y}]$ can, however, be closed in the presence of suitable numerical predicates. Numerical predicates depend only on positions, and not on the tokens occurring at those positions (Straubing, 1994, II.2). A particularly important family is modular predicates MOD_m^r, with $m > 0$ and $r \geq 0$:

$$\mathbf{w}, n \models \mathrm{MOD}_m^r \quad \text{iff} \quad n \equiv r \pmod{m}. \tag{4}$$

[1] The **nesting depth** of a position is the number of unmatched opening parentheses at that point in the string, and the **maximum nesting depth** of a string is the largest such number attained anywhere in the string.

[2] Bounded Dyck languages are often used as simple proxies for hierarchical structure in syntax (Hewitt et al., 2020).

Proposition 2.13. *For every $k > 1$, if modular predicates MOD^r_m are available for some $m \geq k$, then* **LTL** $\left[\mathbf{Y}^{\leq k}\right]$ *has the same expressive power as* **LTL** $[\mathbf{Y}]$*, and* **LTL** $\left[\mathbf{P}, \mathbf{Y}^{\leq k}\right]$ *has the same expressive power as* **LTL** $[\mathbf{P}, \mathbf{Y}]$.

Proof. See §C.8. ∎

3 The Transformer

In this section, we formally define the transformer architecture we consider. Note that we analyze the transformer as a language *recognizer*.

3.1 Preliminaries

All arithmetic in this section is carried out over a set of representable numbers $\mathbb{F}$; we defer the precise formalization to §3.4. For now, the reader may assume that all relevant operations (addition, multiplication, exp, etc.) are well-defined over $\mathbb{F}$.

Definition 3.1 (Recognition)**.** *Fix an alphabet Σ. A* ***language recognizer*** *is a function $o\colon \Sigma^* \to \{0,1\}$. A language $L \subseteq \Sigma^*$ is* ***recognized*** *by o if $o(\mathtt{w}) = 1$ for all $\mathtt{w} \in L$ and $o(\mathtt{w}) = 0$ for all $\mathtt{w} \notin L$.*

We can also define string recognition over an EOS-extended alphabet $\overline{\Sigma} \stackrel{\text{def}}{=} \Sigma \cup \{\text{EOS}\}$, where $\text{EOS} \notin \Sigma$ is an **end-of-sequence** token.

Definition 3.2 (EOS-Recognition)**.** *Fix an alphabet Σ. Let $o\colon \overline{\Sigma}^* \to \{0,1\}$ be a recognizer over the* EOS*-extended alphabet. We say that o* **EOS-recognizes** *a language $L \subseteq \Sigma^*$ if o recognizes the language $L_{\text{EOS}} \stackrel{\text{def}}{=} \{\mathtt{w}\text{EOS} \mid \mathtt{w} \in L\} \subseteq \overline{\Sigma}^*$.*

In other words, Definition 3.2 says that, given an input $\mathtt{w} = \mathtt{w}_1 \cdots \mathtt{w}_N \in \Sigma^*$, the recognizer processes the EOS-padded string $\overline{\mathtt{w}} \stackrel{\text{def}}{=} \mathtt{w}_1 \cdots \mathtt{w}_N\text{EOS}$. This convention mirrors the semantics of LTL (see §A.3): a formula ψ is evaluated at position $N+1$, just beyond the last token of $\mathtt{w}$. In the transformer, position $N+1$ corresponds to EOS.

3.2 Attention

The Basic Definition. Attention can be viewed as a matrix-to-matrix function. Let $\mathbf{H} \in \mathbb{F}^{D \times (N+1)}$ be a matrix. We map $\mathbf{H}$ to another matrix as follows. We first define the linear transformations

$$\mathbf{Q}(\mathbf{H}) \stackrel{\text{def}}{=} \mathbf{\Theta}^{\mathrm{Q}}\mathbf{H}, \tag{5a}$$

$$\mathbf{K}(\mathbf{H}) \stackrel{\text{def}}{=} \mathbf{\Theta}^{\mathrm{K}}\mathbf{H}, \tag{5b}$$

$$\mathbf{V}(\mathbf{H}) \stackrel{\text{def}}{=} \mathbf{\Theta}^{\mathrm{V}}\mathbf{H}, \tag{5c}$$

where $\mathbf{\Theta}^{\mathrm{Q}} \in \mathbb{F}^{D_{\mathrm{K}} \times D}$, $\mathbf{\Theta}^{\mathrm{K}} \in \mathbb{F}^{D_{\mathrm{K}} \times D}$, and $\mathbf{\Theta}^{\mathrm{V}} \in \mathbb{F}^{D_{\mathrm{V}} \times D}$ are parameter matrices. These are often called the **query**, **keys** and **values**, respectively. For integers $n, m \in \{1, \ldots, N+1\}$, we then define the **score** as follows

$$\mathbf{S}_{n,m}(\mathbf{H}) \stackrel{\text{def}}{=} \frac{\mathbf{Q}_{:,n}(\mathbf{H}) \cdot \mathbf{K}_{:,m}(\mathbf{H})}{\sqrt{D_{\mathrm{K}}}} \in \mathbb{F}. \tag{6}$$

Next, we define the **attention weights** as

$$\boldsymbol{\alpha}_{n,m}(\mathbf{H}) \stackrel{\text{def}}{=} \frac{\exp(\mathbf{S}_{n,m}(\mathbf{H}))\mathbf{M}_{n,m}}{\sum_{i=1}^{N+1} \exp(\mathbf{S}_{n,i}(\mathbf{H}))\mathbf{M}_{n,i}}, \tag{7}$$

where $\mathbf{M} \in \{0,1\}^{(N+1)\times(N+1)}$ is a binary matrix called the **mask**. Note that $\boldsymbol{\alpha}_{n,m} \stackrel{\text{def}}{=} 0$ for all m if $\mathbf{M}_{n,i} = 0$ for all i, i.e., if all preceding positions are masked. Finally, we define the resulting matrix

$$\mathbf{O}_{:,n}(\mathbf{H}) \stackrel{\text{def}}{=} \sum_{m=1}^{N+1} \boldsymbol{\alpha}_{n,m}(\mathbf{H})\mathbf{V}_{:,m}(\mathbf{H}), \tag{8}$$

which is the **output** of the function. In summary, we define the **attention mechanism** as follows

$$\begin{aligned} \mathbf{A}\colon \mathbb{F}^{D\times(N+1)} &\to \mathbb{F}^{D\times(N+1)} \\ \mathbf{H} &\mapsto \mathbf{O}(\mathbf{H}). \end{aligned} \tag{9}$$

Masking Patterns. We now give an exposition of several different choices of masks $\mathbf{M}$. First, we define the **global mask**

$$\mathbf{M}^*_{n,m} \stackrel{\text{def}}{=} \begin{cases} 1 & \text{if } m < n, \\ 0 & \text{otherwise.} \end{cases} \tag{10}$$

which corresponds to global attention in the attention mechanism. Next, we define the k**-local mask**:

$$\mathbf{M}^{\leq k}_{n,m} \stackrel{\text{def}}{=} \begin{cases} 1 & \text{if } \max(1, n-k) \leq m < n, \\ 0 & \text{otherwise,} \end{cases} \tag{11}$$

which corresponds to k**-local attention**.

Multi-Head Attention. It is also commonplace to combine attention heads. A **multi-head attention** with H heads $\mathbf{O}^1, \ldots, \mathbf{O}^H$ is defined below

$$\mathbf{A}(\mathbf{H}) \stackrel{\text{def}}{=} \sum_{h=1}^{H} \mathbf{\Theta}^h\mathbf{O}^h(\mathbf{H}), \tag{12}$$

where the heads are combined by a linear projection, and each $\mathbf{\Theta}^h \in \mathbb{F}^{D \times D_{\mathrm{V}}}$ is a parameter. Different heads may use different masks: for instance, some heads may use $\mathbf{M}^*$ while others use $\mathbf{M}^{\leq k}$, giving rise to **hybrid global–local attention**.

3.3 Transformer Architecture

We now define a transformer recognizer.

Definition 3.3 (Transformer)**.** *Fix an alphabet* Σ*. A* (D, L)*-**transformer** is a 4-tuple composed of*

- *a token encoder* $\mathbf{e}\colon \overline{\Sigma} \to \mathbb{F}^D$*,*
- *a family of attention mechanisms* $\{\mathbf{A}^\ell\}_{\ell=1}^L$*,*
- *a family of non-linear functions* $\{\mathbf{F}^\ell\}_{\ell=1}^L$ *where each* $\mathbf{F}^\ell\colon \mathbb{F}^D \to \mathbb{F}^D$*, and*
- *a classification function* $c\colon \mathbb{F}^D \to \{0, 1\}$*.*

A transformer EOS-recognizes a language as follows. Given $\overline{\mathtt{w}} = \mathtt{w}_1 \cdots \mathtt{w}_N\text{EOS}$ where $\mathtt{w}_1, \ldots, \mathtt{w}_N \in \Sigma$, define the base case

$$\mathbf{H}^0(\overline{\mathtt{w}})_{:,n} \stackrel{\text{def}}{=} \mathbf{e}(\mathtt{w}_n), \quad \forall n \in \{1, \ldots, N+1\}. \tag{13}$$

Then, for each layer $\ell = 1, \ldots, L$, compute

$$\mathbf{G}^\ell(\overline{\mathtt{w}}) \stackrel{\text{def}}{=} \text{LN}\left(\mathbf{H}^{\ell-1}(\overline{\mathtt{w}}) + \mathbf{A}^\ell(\mathbf{H}^{\ell-1}(\overline{\mathtt{w}}))\right), \tag{14a}$$

$$\mathbf{H}^\ell(\overline{\mathtt{w}}) \stackrel{\text{def}}{=} \text{LN}\left(\mathbf{G}^\ell(\overline{\mathtt{w}}) + \mathbf{F}^\ell(\mathbf{G}^\ell(\overline{\mathtt{w}}))\right), \tag{14b}$$

where $\mathbf{F}^\ell$ is applied column-wise, and LN is layer normalization (Ba et al., 2016). The transformer's output is then $o(\mathtt{w}) \stackrel{\text{def}}{=} c(\mathbf{H}^L(\overline{\mathtt{w}})_{:,N+1})$.

3.4 Fixed Precision

In the exposition above, we assumed that transformers operate at **fixed precision**, i.e., there exists a finite set $\mathbb{F}$ of numbers—analogous to the floating-point formats used in practice (IEEE, 2019)—such that all parameters and intermediate values lie in $\mathbb{F}$. Every primitive operation, including $\exp$ and the division in softmax, is performed in $\mathbb{F}$ and rounded back into $\mathbb{F}$. We note that fixed-precision arithmetic differs from exact arithmetic in subtle ways: for instance, addition is not generally associative over $\mathbb{F}$, so the order of summation in softmax can affect the result. Nevertheless, the fixed-precision assumption is standard in the analysis of transformers and is part of what makes the connection to finite automata and temporal logic possible.

4 Characterizing Transformers

We now show that global, local, and hybrid attention correspond exactly to the fragments introduced in §2. We first restate an existing result, relating transformers with global attention to **LTL** [**P**].

Theorem 4.1 (Li and Cotterell, 2025)**.** *A language* L *is* EOS*-recognizable by a* (D, L)*-transformer* $\left(\mathbf{e}, \{\mathbf{A}^\ell\}_{\ell=1}^L, \{\mathbf{F}^\ell\}_{\ell=1}^L, c\right)$ *in which every attention head uses the global mask* $\mathbf{M}^*$ *if and only if it is definable in* **LTL** [**P**].

We now consider k-local attention.

Theorem 4.2. *A language* L *is* EOS*-recognizable by a* (D, L)*-transformer* $\left(\mathbf{e}, \{\mathbf{A}^\ell\}_{\ell=1}^L, \{\mathbf{F}^\ell\}_{\ell=1}^L, c\right)$ *in which every attention head uses the* k*-local mask* $\mathbf{M}^{\leq k}$ *if and only if it is definable in* **LTL** $\left[\mathbf{Y}^{\leq k}\right]$.

Proof sketch. The argument follows the same structure as Li and Cotterell (2025) for Theorem 4.1. Because each query attends to at most k predecessors, all summations in the masked softmax and value aggregation reduce to bounded counting, which is expressible in **LTL** $\left[\mathbf{Y}^{\leq k}\right]$ using bounded nesting of $\mathbf{Y}^{\leq k}$. Conversely, by induction on **LTL** $\left[\mathbf{Y}^{\leq k}\right]$ formulas, Boolean connectives are implemented positionwise, and $\mathbf{Y}^{\leq k}\psi$ is realized by a k-local attention head that aggregates information from the last k positions satisfying ψ. The same refinement step as in Li and Cotterell (2025, Lemma B.12) can be applied. ■

Finally, we consider hybrid transformers that mix global and k-local heads.

Theorem 4.3. *A language* L *is* EOS*-recognizable by a* (D, L)*-transformer* $\left(\mathbf{e}, \{\mathbf{A}^\ell\}_{\ell=1}^L, \{\mathbf{F}^\ell\}_{\ell=1}^L, c\right)$ *in which some heads use* $\mathbf{M}^*$ *and the remaining heads use* $\mathbf{M}^{\leq k}$ *if and only if it is definable in* **LTL** $\left[\mathbf{P}, \mathbf{Y}^{\leq k}\right]$.

Proof. The theorem is a direct corollary of Theorems 4.1 and 4.2. ■

4.1 Outlook

We now translate the logical characterizations into concrete consequences for transformers under different attention patterns. An immediate consequence of Theorems 4.1 to 4.3 together with Corollary 2.10 is that transformers with hybrid global-local attention are strictly more expressive than either global-only or k-local-only transformers. Thus, mixing local and global attention is not only computationally attractive (Luong et al., 2015; Child et al., 2019; Beltagy et al., 2020; Zaheer et al., 2020), but can also yield a strict expressive gain. Moreover, 1-local attention is the strongest choice within the local-attention family: by Propositions 2.11 and 2.12 and Theorems 4.2 and 4.3, for every $k > 1$, transformers with 1-local attention are strictly more expressive than those with k-local attention, and the same holds when global heads

are added. This suggests that 1-local attention is a particularly promising choice in practice. We note, however, two important caveats.

Depth Overhead. Although 1-local attention maximizes expressivity, replacing a single k-local head by 1-local heads may incur a depth overhead. At the logical level, expressing $\mathbf{Y}^{\leq k}\psi$ via Eq. (3a) requires a disjunction of formulas of the form $\mathbf{Y}^{i}\psi$ for $1 \leq i \leq k$. Thus, using only $\mathbf{Y}$ increases the operator depth (§A.2) from $\mathrm{od}(\psi)+1$ to as much as $\mathrm{od}(\psi)+k$. In other words, eliminating $\mathbf{Y}^{\leq k}$ in favor of $\mathbf{Y}$ preserves expressivity, but may increase the required depth by an overhead of up to $k-1$. Specifically, in the proof of Theorem 4.2, each temporal operator requires at least one layer to simulate, so this translation may require up to k stacked layers in place of a single layer. Thus, under a fixed depth, transformers with 1-local attention need not be strictly more expressive than transformers with k-local attention. In practice, modern language models are deep (OpenAI, 2023), making this limitation less of a practical detriment and rendering architectures with 1-local heads a plausible way to increase expressivity.

Positional Encodings. The strict separation above holds in the absence of positional encodings. With sufficiently rich positional information, however, the gap can disappear. Intuitively, if the model can always identify the immediate predecessor within a k-window, then k-local attention can simulate 1-local attention. Positional encodings are often viewed as numerical predicates added to the logic (Yang et al., 2024; Li and Cotterell, 2025). By Proposition 2.13, suitable modular predicates suffice to erase the gap. Therefore, if positional encodings that simulate such predicates are available, then a k-local head can recover the immediate predecessor by selecting, at each position, the unique position in the previous k steps. Interestingly, sinusoidal encodings (`SiPE`; Vaswani et al. 2017) and rotary encodings (`RoPE`; Su et al. 2024) can be related to modular predicates (Yang et al., 2025). However, realizing modular predicates in this way requires a rational variant (Chiang et al., 2023), which is not the form typically used in practice.

5 Empirically Verifying the Theory

In this section, we test the expressivity predictions of our theory on formal language recognition tasks.

5.1 Languages

We evaluate a suite of formal languages grouped by definability in temporal-logic fragments, selecting two representative languages from each class:

- **LTL $[\mathbf{Y}]$-definable:** $\Sigma^{*}\mathsf{a}$ (strings ending with a) and $\Sigma^{*}\mathsf{ab}$ (strings ending with ab).
- **LTL $[\mathbf{P}]$-definable:** $\mathsf{a}\Sigma^{*}$ (strings starting with a) and $\Sigma^{*}\mathsf{a}\Sigma^{*}\mathsf{b}\Sigma^{*}$ (strings containing ab as a not-necessarily-contiguous subsequence).
- **LTL $[\mathbf{P}, \mathbf{Y}]$-definable but not in LTL $[\mathbf{P}]$ or LTL $[\mathbf{Y}]$:** $(\mathsf{ab})^{*}$ (strings of repeated ab) and $\Sigma^{*}\mathsf{ab}\Sigma^{*}$ (strings containing ab as a contiguous substring).
- **LTL$[\mathbf{S}]$-definable but not in LTL $[\mathbf{P}, \mathbf{Y}]$:** $\{\mathsf{a},\mathsf{b},\mathsf{d}\}^{*}\mathsf{a}\{\mathsf{c},\mathsf{d}\}^{*}$ (a right-deterministic polynomial) and $(\mathsf{a}(\mathsf{ab})^{*}\mathsf{b})^{*}$ (bounded Dyck with maximum nesting depth 2).

5.2 Experimental Setup

We consider three attention settings, corresponding to the masking patterns defined in §3: *local-only* (all heads use $\mathbf{M}^{\leq k}$), *hybrid* (half the heads use $\mathbf{M}^{\leq k}$, the rest use $\mathbf{M}^{*}$), and *global-only* (all heads use $\mathbf{M}^{*}$). For local-only and hybrid models, we vary the window size $k \in \{1, 2, 4\}$. We also evaluate transformers with two positional encodings: sinusoidal encodings (`SiPE`; Vaswani et al. 2017) and rotary encodings (`RoPE`; Su et al. 2024). The models are trained on strings of length up to 40 and evaluated on lengths 41–500. Each experiment is run with 5 random seeds and 3 learning rates; additional details are given in §D.2. In addition to accuracy, we report the *longest perfect length*, defined as the largest length up to which the model achieves 100% accuracy.

5.3 Results

We summarize performance using a heatmap of the longest perfect length in Fig. 3. Full numerical results, including accuracies and longest perfect lengths, are provided in §D.3.

5.3.1 Results on `NoPE`

We begin with a discussion of the `NoPE` setting, whose results are shown on the left panel of Fig. 3.

Local-only models match LTL $[\mathbf{Y}]$. As predicted, local-only transformers succeed precisely on the LTL $[\mathbf{Y}]$-definable languages: they achieve perfect generalization up to length 500 on the two LTL $[\mathbf{Y}]$ tasks in Fig. 3, while failing to generalize on the remaining languages.

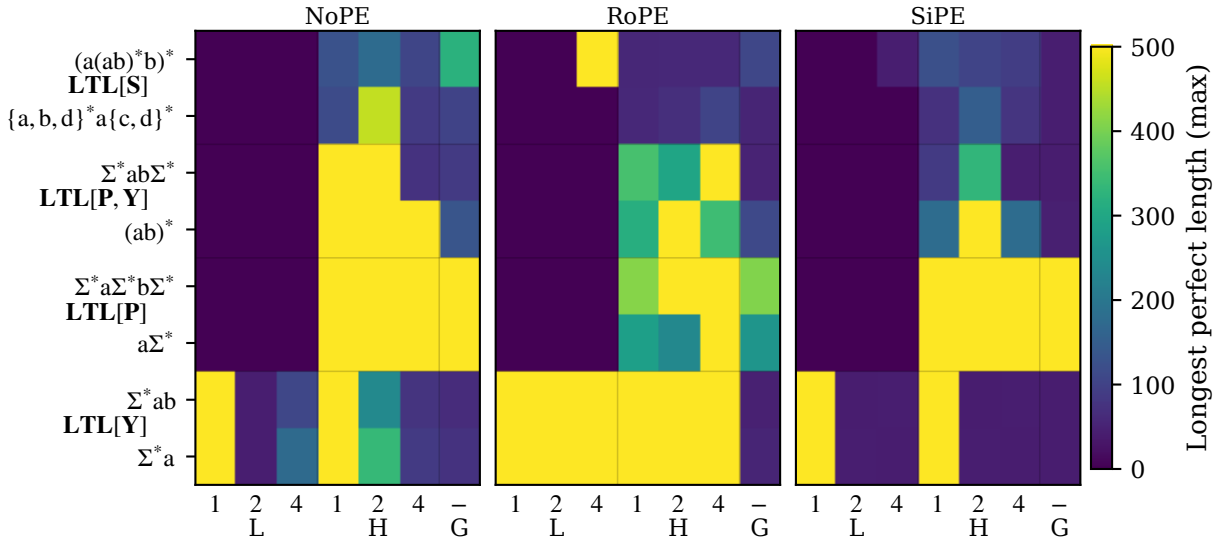


Figure 3: Heatmap of longest perfect lengths (maximum over runs) across formal languages, attention patterns, and positional encodings. Columns denote local (L), hybrid (H), and global (G) attention patterns; for L and H, the labels above columns indicate the local window size k.

Global-only models match LTL [P]. Conversely, global-only models succeed only on the LTL [P]-definable languages and do not exhibit perfect length generalization beyond this class.

Hybrid models match LTL [P, Y]. Hybrid models, especially with $k = 1$, succeed on all languages definable in LTL [P, Y], which subsumes both LTL [P] and LTL [Y]. As visualized in Fig. 3, they generalize perfectly up to length 500 on the bottom six languages.

1-local attention. Among the tested window sizes, $k = 1$ is consistently the strongest performer. Both $k = 2$ and $k = 4$ fail even on definite languages, and increasing the window can hurt: for example, hybrid models with $k = 2$ learn $\Sigma^*\mathtt{ab}\Sigma^*$, whereas those with $k = 4$ do not.

Beyond LTL [P, Y]. As expected, none of the models perfectly generalizes on the two languages in LTL[S] \ LTL [P, Y], namely the right-deterministic polynomial and bounded Dyck. Nevertheless, hybrid attention yields the strongest partial generalization on these tasks.

5.3.2 Positional Encodings

We now turn to the settings with positional encodings (middle and right panels of Fig. 3).

Global-only models. One might hope that positional encodings would allow global attention to approximate locality. In our experiments, however, neither SiPE nor RoPE compensates for the absence of local attention: Positional encodings do not erase the qualitative separation between global-only and hybrid models; on the contrary, they often reduce performance on the tested languages.

Local-only and hybrid models. With RoPE, the gap between 1-local and larger-window local attention becomes smaller. In particular, local-only models with $k = 2$ or $k = 4$ can now recognize $\Sigma^*\mathtt{a}$ and $\Sigma^*\mathtt{ab}$. In hybrid models, RoPE also enables $k = 4$ to recognize $\Sigma^*\mathtt{ab}\Sigma^*$. However, these gains come with tradeoffs, as RoPE degrades performance in several other settings. By comparison, SiPE does not narrow the gap and instead tends to reduce performance across the board.

Summary. Overall, on one hand, the results under NoPE align exactly with our theory. On the other hand, the role of positional encodings is less clear-cut and merits further investigation.

6 An Experiment on Natural Language

We also experiment with whether our theory has something to say about natural language. Thus, we run a lightweight next-token language modeling experiment on WikiText-2.

6.1 Experimental Setup

We train a GPT-2-small-style decoder-only Transformer from scratch, using the standard GPT-2 small architecture (12 layers, hidden size 768, 12 attention heads, and feed-forward size 3072). We keep the architecture and optimization setup fixed across all runs, varying only the attention pattern.

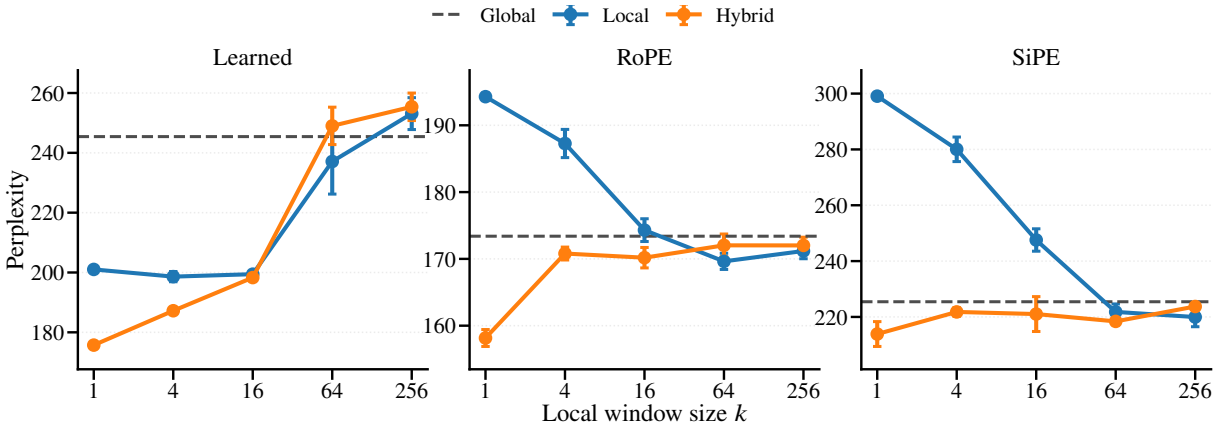


Figure 4: Perplexity on WikiText-2 for local, hybrid, and global attention patterns under different positional encodings. Curves show mean perplexity across runs for local and hybrid models as a function of the local window size k, and error bars indicate standard deviations. The dashed horizontal line shows the corresponding global baseline.

Concretely, we compare global-only, local-only, and hybrid attention, where in the hybrid setting half of the heads are replaced by local heads. To remain consistent with the formal setup in §3, we use strict causal masking throughout. We consider local window sizes $k \in \{1, 4, 16, 64, 256\}$ and three positional encoding schemes: learned absolute positional embeddings, `RoPE`, and `SiPE`. For optimization, we use AdamW (Loshchilov and Hutter, 2019) with learning rate 2.5×10^{-4}, weight decay 0.01, 2000 warmup steps, cosine learning-rate decay, per-device batch size 2, gradient accumulation of 32 steps, and early stopping based on validation loss.

6.2 Results

The results show a clear and consistent pattern. Across all positional encoding choices, the hybrid model is always the strongest, and within the hybrid family, $k = 1$ is always the best setting. Hybrid attention with $k = 1$ reduces perplexity by 69.7, 15.2, 11.5 relative to the global-only baseline under various positional encodings. Increasing the local window generally degrades performance, and with sufficiently large windows the hybrid model can match or underperform the global-only baseline.

We also find that local-only attention can outperform global-only attention, which is surprising but compatible with our theory, since local-only and global-only attention are incomparable in expressive power. One possible explanation is that WikiText-2 relies more heavily on short-range patterns than on the longer-range dependencies. The effect of window size, however, depends on the positional encoding. Under learned positional embeddings, smaller local windows perform better. Under `RoPE` and `SiPE`, local-only models instead tend to prefer larger windows. This is consistent with the theory: although 1-local attention is maximally expressive in the unbounded setting, its advantage may disappear under a fixed depth bound, especially when suitable positional encodings narrow the gap between different local window sizes.

Overall, our results support the main claim of the paper: local attention is complementary to global attention, and the hybrid setting combines the strengths of both. In particular, pairing global attention with 1-local attention yields the largest additional benefit.

7 Conclusion

We have given a formal account of why local attention helps in transformer language models. By connecting attention patterns to fragments of linear temporal logic, we showed that local and global attention are complementary rather than interchangeable, and that adding local attention to a globally attentive transformer strictly increases expressive power. In particular, global attention alone corresponds to the logic $\mathbf{LTL}\,[\mathbf{P}]$, purely k-local attention corresponds to $\mathbf{LTL}\left[\mathbf{Y}^{\leq k}\right]$, and hybrid global–local attention corresponds to the richer logic $\mathbf{LTL}\left[\mathbf{P}, \mathbf{Y}^{\leq k}\right]$. The largest gain arises from 1-local attention, which induces the $\mathbf{Y}$ operator and captures, among other classes, the locally testable languages—a classical family beyond the reach of global-only models. Our formal-language experiments empirically confirm these separations, and a preliminary language-modeling experiment on WikiText-2 shows that hybrid attention with $k{=}1$ consistently achieves the best performance.

Limitations

Due to limited computational resources, we are unable to run large-scale language modeling experiments. Instead, we use a lightweight setup that is intended as a sanity check while remaining manageable in terms of model size, dataset, and training budget. Accordingly, our natural-language results should be interpreted with caution. It is possible that the qualitative picture changes with larger models, longer training, or broader datasets.

Ethical Considerations

We do not foresee any ethical issues.

References

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer normalization. In *NIPS Deep Learning Symposium*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *The Third International Conference on Learning Representations*.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *Computing Research Repository*, arXiv:2004.05150. Version 2.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Janusz Brzozowski, Baiyu Li, and David Liu. 2012. Syntactic complexities of six classes of star-free languages. *Journal of Automata, Languages and Combinatorics*, 17(2):83–105.

Alexandra Butoi, Ghazal Khalighinejad, Anej Svete, Josef Valvoda, Ryan Cotterell, and Brian DuSell. 2025. Training neural networks as recognizers of formal languages. In *The Thirteenth International Conference on Learning Representations*.

David Chiang, Peter Cholak, and Anand Pillay. 2023. Tighter bounds on the expressivity of transformer encoders. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 5544–5562. PMLR.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. 2019. Generating long sequences with sparse transformers. *Computing Research Repository*, arXiv:1904.10509.

Joëlle Cohen, Dominique Perrin, and Jean-Eric Pin. 1993. On the expressive power of temporal logic. *Journal of Computer and System Sciences*, 46(3):271–294.

DeepSeek-AI. 2025. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *Computing Research Repository*, arXiv:2501.12948.

Grégoire Delétang, Anian Ruoss, Jordi Grau-Moya, Tim Genewein, Li Kevin Wenliang, Elliot Catt, Chris Cundy, Marcus Hutter, Shane Legg, Joel Veness, and Pedro A Ortega. 2023. Neural networks and the chomsky hierarchy. In *The Eleventh International Conference on Learning Representations*.

Samuel Eilenberg. 1974. *Automata, Languages, and Machines*. Number pt. 2 in 59/B. Academic Press.

Dov Gabbay, Amir Pnueli, Saharon Shelah, and Jonathan Stavi. 1980. On the temporal analysis of fairness. POPL '80, page 163–173, New York, NY, USA. Association for Computing Machinery.

James A. Green. 1951. On the structure of semigroups. *Annals of Mathematics*, 54(1):163–172.

John Hewitt, Michael Hahn, Surya Ganguli, Percy Liang, and Christopher D. Manning. 2020. RNNs can generate bounded hierarchical languages with optimal memory. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1978–2010, Online. Association for Computational Linguistics.

IEEE. 2019. IEEE standard for floating-point arithmetic. IEEE Std 754-2019.

Johan Anthony Wilem Kamp. 1968. *Tense Logic and the Theory of Linear Order*. Ph.D. thesis, University of California, Los Angeles.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *Computing Research Repository*, arXiv:1412.6980. Version 9.

Stephen C. Kleene. 1956. *Representation of Events in Nerve Nets and Finite Automata*, pages 3–42. Princeton University Press, Princeton.

Jiaoda Li and Ryan Cotterell. 2025. Characterizing the expressivity of fixed-precision transformer language models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Zhiyuan Li, Hong Liu, Denny Zhou, and Tengyu Ma. 2024. Chain of thought empowers transformers to solve inherently serial problems. In *The Twelfth International Conference on Learning Representations*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *The Seventh International Conference on Learning Representations*.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.

Robert McNaughton and Seymour Papert. 1971. *Counter-free Automata*. M.I.T. Press research monographs. M.I.T. Press.

William Merrill and Ashish Sabharwal. 2024. The expressive power of transformers with chain of thought. In *The Twelfth International Conference on Learning Representations*.

OpenAI. 2023. GPT-4 technical report. *Computing Research Repository*, arXiv:2303.08774.

Micha A. Perles, Michael O. Rabin, and Eli Shamir. 1963. The theory of definite automata. *IEEE Transactions on Electronic Computers*, 12:233–243.

Amir Pnueli. 1977. The temporal logic of programs. In *18th Annual Symposium on Foundations of Computer Science (sfcs 1977)*, pages 46–57.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. OpenAI technical report.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. OpenAI technical report.

Jürgen Schmidhuber. 1992. Learning to control fast-weight memories: An alternative to dynamic recurrent networks. In *Neural Computation*, volume 4, pages 131–139. MIT Press.

Howard Straubing. 1994. *Finite Automata, Formal Logic, and Circuit Complexity*. Progress in Theoretical Computer Science. Birkhäuser Boston, Boston, MA.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. 2024. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063.

OLMo Team, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Chang, Khyathi Chandu, Akshita Bhagia, Oyvind Tafjord, and 1 others. 2025. OLMo 2: The best fully open language model to date. *Computing Research Repository*, arXiv:2501.00656.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H Chi, Quoc V Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, and 3 others. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45.

Andy Yang, Michaël Cadilhac, and David Chiang. 2025. Knee-deep in c-RASP: A transformer depth hierarchy. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Andy Yang, David Chiang, and Dana Angluin. 2024. Masked hard-attention transformers recognize exactly the star-free languages. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Andy Yang, Lena Strobl, David Chiang, and Dana Angluin. 2026. Simulating hard attention using soft attention. *Transactions of the Association for Computational Linguistics*, 14.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. 2020. Big bird: Transformers for longer sequences. In *Advances in Neural Information Processing Systems*, volume 33, pages 17283–17297. Curran Associates, Inc.

Table of Contents

A Linear Temporal Logic

We now give the basic definitions of linear temporal logic.

A.1 Syntax

We define the syntax of full linear temporal logic, $\mathbf{LTL}[\mathbf{P}, \mathbf{Y}, \mathbf{S}, \mathbf{U}]$, over an alphabet Σ. The set of **formulas** over Σ is defined inductively:

- **Constants:** $\top$ and $\bot$ are formulas.
- **Atomic predicates:** $\pi_{\mathtt{a}}$ is a formula for $\mathtt{a} \in \Sigma$.
- **Boolean connectives:** if ψ_1 and ψ_2 are formulas, then so are $\neg\psi_1$, $\psi_1 \wedge \psi_2$, and $\psi_1 \vee \psi_2$.
- **Temporal operators:** if ψ and ψ' are formulas, then so are $\mathbf{Y}\psi$ ("yesterday"), $\mathbf{P}\psi$ ("past"), $\psi\mathbf{S}\psi'$ ("since"), and $\psi\mathbf{U}\psi'$ ("until").

A.2 Operator Depth

The **operator depth** of a formula ϕ, denoted $\mathrm{od}(\phi)$, is defined inductively:

$$\mathrm{od}(\phi) \stackrel{\text{def}}{=} \begin{cases} 0 & \text{if } \phi \in \{\top, \bot, \pi_{\mathtt{a}}\} \\ \mathrm{od}(\psi) & \text{if } \phi = \neg\psi \\ \max\left(\mathrm{od}(\psi_1), \mathrm{od}(\psi_2)\right) & \text{if } \phi = \psi_1 \circ \psi_2,\ \circ \in \{\wedge, \vee\} \\ 1 + \mathrm{od}(\psi) & \text{if } \phi = \circ\,\psi,\ \circ \in \{\mathbf{Y}, \mathbf{P}\} \\ 1 + \max\left(\mathrm{od}(\psi_1), \mathrm{od}(\psi_2)\right) & \text{if } \phi = \psi_1 \circ \psi_2,\ \circ \in \{\mathbf{S}, \mathbf{U}\} \end{cases} \tag{15}$$

A.3 Semantics

Given a string $\mathtt{w}$ and a position n, the semantics for the satisfaction relation $\mathtt{w}, n \models \psi$ is defined as:

- $\mathtt{w}, n \models \pi_{\mathtt{a}}$ iff $1 \leq n \leq |\mathtt{w}|$ and the n^{th} token of $\mathtt{w}$ is $\mathtt{a}$;
- $\mathtt{w}, n \models \psi_1 \vee \psi_2$ iff $\mathtt{w}, n \models \psi_1$ or $\mathtt{w}, n \models \psi_2$;
- $\mathtt{w}, n \models \psi_1 \wedge \psi_2$ iff $\mathtt{w}, n \models \psi_1$ and $\mathtt{w}, n \models \psi_2$;
- $\mathtt{w}, n \models \neg\psi$ iff $\mathtt{w}, n \not\models \psi$;
- $\mathtt{w}, n \models \mathbf{Y}\psi$ iff $\mathtt{w}, n-1 \models \psi$;
- $\mathtt{w}, n \models \mathbf{P}\psi$ iff there exists m such that $m < n$ and $\mathtt{w}, m \models \psi$;
- $\mathtt{w}, n \models \psi_1\mathbf{S}\psi_2$ iff there exists m such that $m < n$, $\mathtt{w}, m \models \psi_2$, and for all i such that $m < i < n$, $\mathtt{w}, i \models \psi_1$;
- $\mathtt{w}, n \models \psi_1\mathbf{U}\psi_2$ iff there exists m such that $m > n$, $\mathtt{w}, m \models \psi_2$, and for all i such that $n < i < m$, $\mathtt{w}, i \models \psi_1$.

A string $\mathtt{w}$ of length N **satisfies** a formula ψ, written $\mathtt{w} \models \psi$, if ψ holds at a position outside the string, such as position 0 or $N+1$. In this work, we evaluate formulas at position $N+1$, just beyond the end of the string, since we consider only past-time operators such as $\mathbf{Y}$ and $\mathbf{P}$, i.e.,

$$\mathtt{w} \models \psi \quad \text{iff} \quad \mathtt{w}, N+1 \models \psi. \tag{16}$$

The language **defined** by a formula ψ is the set of all strings that satisfy it, i.e., the set

$$\mathbb{L}(\psi) \stackrel{\text{def}}{=} \{\mathtt{w} \in \Sigma^* \mid \mathtt{w} \models \psi\}. \tag{17}$$

B Additional Background

We collect background on formal language classes, syntactic semigroups, and deterministic finite automata, which we use to characterize transformers under different attention patterns.

B.1 Formal Languages

Regular Languages. A **regular expression** is a declarative description of a language over Σ. It is defined inductively as follows:

- $\emptyset$ and each $\mathtt{a} \in \Sigma$ are regular expressions;
- if α and β are regular expressions, then so are the union $\alpha + \beta$, the concatenation $\alpha\beta$, the Kleene star α^*, and the complement α^{c}.

A language is **regular** if and only if it can be described by a regular expression (Kleene, 1956). A regular language is **star-free** if it can be described by a regular expression that does not use the Kleene star (McNaughton and Papert, 1971). For example, Σ^* is star-free, since it can be described by the regular expression $\emptyset^{\mathsf{c}}$.

Monomials. A **monomial** over Σ is a language of the form $\Sigma_0^*\mathtt{a}_1\Sigma_1^* \cdots \mathtt{a}_n\Sigma_n^*$, where $\mathtt{a}_1, \ldots, \mathtt{a}_n \in \Sigma$ and $\Sigma_0, \Sigma_1, \ldots, \Sigma_n \subseteq \Sigma$. A monomial is called

- **left-deterministic** if for every $k \in \{1, \ldots, n\}$, $\mathtt{a}_k \notin \Sigma_{k-1}$;
- **right-deterministic** if for every $k \in \{1, \ldots, n\}$, $\mathtt{a}_k \notin \Sigma_k$;
- **unambiguous** if every string $\mathtt{w}$ in the monomial admits at most one factorization $\mathtt{w} = \mathtt{w}_0\mathtt{a}_1\mathtt{w}_1 \cdots \mathtt{a}_n\mathtt{w}_n$ with $\mathtt{w}_k \in \Sigma_k^*$ for all $k \in \{0, \ldots, n\}$.

Polynomials. A **polynomial** is a finite union of monomials. It is called left-deterministic (resp. right-deterministic or unambiguous) if it is a finite disjoint union of left-deterministic (resp. right-deterministic or unambiguous) monomials.

B.2 Syntactic Monoids and Semigroups

Semigroups and Monoids. A **semigroup** is a set endowed with an associative binary operation. A **monoid** is a semigroup with an identity element.

$\mathcal{R}$-trivial Monoids. Recall Green's $\mathcal{R}$ relation on a monoid $\mathbb{M}$ (Green, 1951). For $s, t \in \mathbb{M}$, we write $s \mathcal{R} t$ if they generate the same principal right ideal:

$$s\mathbb{M} = t\mathbb{M}. \tag{18}$$

A *monoid* $\mathbb{M}$ is **$\mathcal{R}$-trivial** if $\mathcal{R}$ is equality; equivalently, for all $s, t \in \mathbb{M}$,

$$s \mathcal{R} t \quad \Longrightarrow \quad s = t. \tag{19}$$

Locally $\mathcal{R}$-trivial Semigroups. An element $e \in \mathbb{S}$ is **idempotent** if $e^2 = e$. A *semigroup* $\mathbb{S}$ is **locally $\mathcal{R}$-trivial** if, for every idempotent $e \in \mathbb{S}$, the submonoid

$$e\mathbb{S}e \stackrel{\text{def}}{=} \{ese \mid s \in \mathbb{S}\} \tag{20}$$

with identity element e is $\mathcal{R}$-trivial.

Syntactic Congruence. Let $L \subseteq \Sigma^*$ be a language. Two strings $\mathtt{s}, \mathtt{t} \in \Sigma^*$ are **syntactically equivalent**, denoted $\mathtt{s} \equiv_L \mathtt{t}$, if

$$\forall \mathtt{u}, \mathtt{v} \in \Sigma^* : \qquad \mathtt{usv} \in L \;\Leftrightarrow\; \mathtt{utv} \in L. \tag{21}$$

We write $[\mathtt{w}]$ for the equivalence class of $\mathtt{w}$.

Syntactic Monoid. The quotient monoid

$$\Sigma^* / \equiv_L, \tag{22}$$

whose elements are equivalence classes and whose operation is induced by concatenation, is called the **syntactic monoid** of L.

Syntactic Semigroup. Let Σ^+ denote the set of nonempty strings over Σ. Restricting the syntactic congruence to Σ^+ yields the quotient semigroup

$$\Sigma^+/\equiv_L. \tag{23}$$

This quotient is called the **syntactic semigroup** of L.

B.3 Deterministic Finite Automata

Definition. A deterministic finite automaton (DFA) is a 5-tuple $\mathcal{A} = (\Sigma, Q, q_I, F, \delta)$, where

- Σ is an alphabet;
- Q is a finite set of states;
- $q_I \in Q$ is the initial state;
- $F \subseteq Q$ is the set of accepting states;
- $\delta\colon Q \times \Sigma \to Q$ is the transition function.

We extend δ to $Q \times \Sigma^*$ recursively by

$$\delta(q, \varepsilon) = q \tag{24}$$

for every $q \in Q$, and

$$\delta(q, \mathtt{wa}) = \delta(\delta(q, \mathtt{w}), \mathtt{a}) \tag{25}$$

for every $q \in Q$, $\mathtt{w} \in \Sigma^*$, and $\mathtt{a} \in \Sigma$.

Acceptance. A DFA $\mathcal{A}$ accepts a string $\mathtt{w} = \mathtt{w}_1 \cdots \mathtt{w}_N \in \Sigma^*$ if the unique run $q_0, \ldots, q_N$ defined by $q_0 = q_I$ and $q_{n+1} = \delta(q_n, \mathtt{w}_{n+1})$ for $n = 0, \ldots, N-1$ ends in an accepting state, i.e., $q_N \in F$.

Minimal DFA. A DFA $\mathcal{A}$ **recognizes** a language $L \subseteq \Sigma^*$ if it accepts exactly the strings in L. A DFA recognizing L is **minimal** if it has the smallest number of states among all DFAs recognizing L.

Partially Ordered DFAs. A DFA is **partially ordered** if there exists a partial order $\preceq$ on Q such that for every $q \in Q$ and $\mathtt{a} \in \Sigma$, we have $q \preceq \delta(q, \mathtt{a})$. Equivalently, the state-transition graph has no cycles other than self-loops.

Transformation. For each non-empty string $\mathtt{w} \in \Sigma^+$, the extended transition function induces a transformation $t_{\mathtt{w}}\colon Q \to Q$ defined by

$$q t_{\mathtt{w}} \stackrel{\text{def}}{=} \delta(q, \mathtt{w}). \tag{26}$$

A permutation of Q is a bijection from Q to itself. A transformation $t\colon Q \to Q$ is **permutational** if there exists a subset $P \subseteq Q$ with $|P| \geq 2$ such that the restriction

$$t|_P\colon P \to P \tag{27}$$

is a permutation of P.

Configuration Containment. Let $\mathcal{A} = (\Sigma, Q, q_I, F, \delta)$ be a DFA. Define $G(\mathcal{A})$ to be the directed labeled graph whose vertex set is Q and whose edge set consists of all triples $(q, \mathtt{w}, q')$, such that $q, q' \in Q$, $\mathtt{w} \in \Sigma^+$, and $\delta(q, \mathtt{w}) = q'$. Because Q is finite but Σ^+ is infinite, the graph $G(\mathcal{A})$ has finitely many vertices and, in general, infinitely many labeled edges. If a directed graph whose edges are labeled by non-empty strings is a labeled subgraph of $G(\mathcal{A})$, then we say that $\mathcal{A}$ contains it as a configuration.

C Proofs

C.1 Proof of Theorem 2.2

Theorem 2.2. *Let $L \subseteq \Sigma^*$ be a regular language, let $\mathbb{S}$ be its syntactic semigroup, and let $\mathcal{A}$ be the minimal DFA recognizing L. The following are equivalent:*

1. *L is definable in* **LTL** [**Y**]*;*
2. *L is definite;*
3. *every idempotent $e \in \mathbb{S}$ is a right zero, i.e., for all $s \in \mathbb{S}$, $se = e$;*

4. *every transformation induced by a non-empty string is non-permutational.*

Proof. (1$\Longrightarrow$2) Let ψ be a **LTL** [**Y**] formula, and let $r = \mathrm{od}(\psi)$. We prove, by structural induction on ψ, the following stronger claim:

$(*)$ For every string $\mathtt{w} = \mathtt{w}_1 \cdots \mathtt{w}_N$ and every position n, the truth of $\mathtt{w}, n \models \psi$ depends only on the factor

$$\mathtt{w}_{\max(1,n-r)} \cdots \mathtt{w}_{\min(n,N)}. \tag{28}$$

Base case: od$(\psi) = 0$. Then ψ is a Boolean combination of constants and atomic formulas. Each atomic formula $\pi_{\mathtt{a}}$ depends only on the token at position n (if $1 \le n \le N$), and is false otherwise. Hence the truth of $\mathtt{w}, n \models \psi$ depends only on the token at position n.

Induction step. Suppose $\psi = \mathbf{Y}\phi$ and $\mathrm{od}(\phi) = r - 1$. Then

$$\mathtt{w}, n \models \mathbf{Y}\phi \quad \Longleftrightarrow \quad \mathtt{w}, n-1 \models \phi. \tag{29}$$

By the induction hypothesis, the truth of $\mathtt{w}, n-1 \models \phi$ depends only on

$$\mathtt{w}_{\max(1,n-r)} \cdots \mathtt{w}_{\min(n-1,N)}. \tag{30}$$

Therefore $\mathtt{w}, n \models \mathbf{Y}\phi$ depends only on

$$\mathtt{w}_{\max(1,n-r)} \cdots \mathtt{w}_{\min(n,N)}. \tag{31}$$

Boolean connectives preserve the claim, so $(*)$ follows for all **LTL** [**Y**] formulas.

Now let $n = N + 1$. Since $\min(N+1, N) = N$, the truth of $\mathtt{w}, N+1 \models \psi$ depends only on

$$\mathtt{w}_{\max(1,N-r+1)} \cdots \mathtt{w}_N, \tag{32}$$

that is, only on the suffix of $\mathtt{w}$ of length at most r. Therefore $\mathbb{L}(\psi)$ is definite.

(2$\Longrightarrow$1) Suppose that L is definite. Then L is m-definite for some $m \ge 0$. By definition, L is a Boolean combination of languages of the form

$$\{\mathtt{w} \in \Sigma^* \mid S_m(\mathtt{w}) = \mathtt{u}\}, \tag{33}$$

where $\mathtt{u} \in \Sigma^m$.

We show that each such basic language is definable in **LTL** [**Y**]. For each string $\mathtt{u} \in \Sigma^*$, define a formula $\psi_{\mathtt{u}}$ inductively by

$$\psi_\varepsilon \stackrel{\text{def}}{=} \top, \tag{34}$$

and, for $\mathtt{v} \in \Sigma^*$ and $\mathtt{a} \in \Sigma$,

$$\psi_{\mathtt{va}} \stackrel{\text{def}}{=} \mathbf{Y}(\pi_{\mathtt{a}} \wedge \psi_{\mathtt{v}}). \tag{35}$$

We prove by induction on $|\mathtt{u}|$ that, for every string $\mathtt{w} = \mathtt{w}_1 \cdots \mathtt{w}_N$ and every position $n \ge |\mathtt{u}| + 1$,

$$\mathtt{w}, n \models \psi_{\mathtt{u}} \quad \Longleftrightarrow \quad \mathtt{w}_{n-|\mathtt{u}|} \cdots \mathtt{w}_{n-1} = \mathtt{u}. \tag{36}$$

Base case: $|\mathtt{u}| = 0$. Then $\mathtt{u} = \varepsilon$, and by definition $\psi_\varepsilon \stackrel{\text{def}}{=} \top$. Hence $\mathtt{w}, n \models \psi_\varepsilon$ for every $\mathtt{w}$ and every n. Since the empty factor is ε, the claim follows.

Induction step. Suppose $\mathtt{u} = \mathtt{v}\mathtt{a}$ with $\mathtt{v} \in \Sigma^*$ and $\mathtt{a} \in \Sigma$. By definition,

$$\psi_{\mathtt{v}\mathtt{a}} \stackrel{\text{def}}{=} \mathbf{Y}(\pi_{\mathtt{a}} \wedge \psi_{\mathtt{v}}). \tag{37}$$

Let $n \geq |\mathtt{u}| + 1$. Then

$$\mathtt{w}, n \models \psi_{\mathtt{v}\mathtt{a}} \iff \mathtt{w}, n-1 \models \pi_{\mathtt{a}} \wedge \psi_{\mathtt{v}} \tag{38}$$

The first conjunct says that the token at position $n-1$ is $\mathtt{a}$. Since $|\mathtt{v}| = |\mathtt{u}| - 1$, we have $n - 1 \geq |\mathtt{v}| + 1$, so by the induction hypothesis applied at position $n-1$,

$$\mathtt{w}, n-1 \models \psi_{\mathtt{v}} \quad \iff \quad \mathtt{w}_{n-|\mathtt{u}|} \cdots \mathtt{w}_{n-2} = \mathtt{v}. \tag{39}$$

Therefore,

$$\mathtt{w}, n \models \psi_{\mathtt{u}} \quad \iff \quad \mathtt{w}_{n-|\mathtt{u}|} \cdots \mathtt{w}_{n-1} = \mathtt{v}\mathtt{a} = \mathtt{u}. \tag{40}$$

This completes the induction.

Applying the claim at position $N+1$, we obtain

$$\begin{aligned} \mathtt{w} \models \psi_{\mathtt{u}} \quad &\iff \quad \mathtt{w}, N+1 \models \psi_{\mathtt{u}} && (41)\\ &\iff \quad \mathtt{w}_{N-m+1} \cdots \mathtt{w}_N = \mathtt{u} && (42)\\ &\iff \quad S_m(\mathtt{w}) = \mathtt{u}. && (43) \end{aligned}$$

Because $\mathbf{LTL}\,[\mathbf{Y}]$ is closed under Boolean connectives, every Boolean combination of such languages is also definable in $\mathbf{LTL}\,[\mathbf{Y}]$. Therefore L is definable in $\mathbf{LTL}\,[\mathbf{Y}]$.

The equivalence 2$\iff$3 can be found in Eilenberg (1974) and Brzozowski et al. (2012), and the equivalence 2$\iff$4 can be found in Perles et al. (1963) and Brzozowski et al. (2012). ■

C.2 Proof of Theorem 2.7

Theorem 2.7. *Let $L \subseteq \Sigma^*$ be a regular language, $\mathbb{S}$ its syntactic semigroup, and $\mathcal{A}$ the minimal DFA recognizing L. The following are equivalent:*

1. *L is definable in $\mathbf{LTL}\,[\mathbf{P}, \mathbf{Y}]$;*
2. *$\mathbb{S}$ is locally $\mathcal{R}$-trivial;*
3. *$\mathcal{A}$ contains no configuration of the form shown in Fig. 1.*

Proof. Cohen et al. (1993) introduce a restricted temporal logic $\mathbf{LTL}\,[\mathbf{P}^{=}, \mathbf{Y}]$ with operators $\mathbf{P}^{=}$ and $\mathbf{Y}$, where

$$\mathtt{w}, n \models \mathbf{P}^{=}\psi \quad \text{iff} \quad \exists\, m \leq n \text{ such that } \mathtt{w}, m \models \psi. \tag{44}$$

They show (Cohen et al., 1993, Proposition 4.1) that a regular language is definable in $\mathbf{LTL}\,[\mathbf{P}^{=}, \mathbf{Y}]$ if and only if its syntactic semigroup is locally $\mathcal{R}$-trivial; equivalently, its minimal DFA avoids the configuration in Fig. 1. It therefore remains to relate $\mathbf{LTL}\,[\mathbf{P}^{=}, \mathbf{Y}]$ and $\mathbf{LTL}\,[\mathbf{P}, \mathbf{Y}]$. In the presence of $\mathbf{Y}$, the operators $\mathbf{P}$ and $\mathbf{P}^{=}$ are mutually definable:

$$\mathbf{P}\psi \equiv \mathbf{Y}\,\mathbf{P}^{=}\psi, \tag{45a}$$

$$\mathbf{P}^{=}\psi \equiv \mathbf{P}\psi \vee \psi. \tag{45b}$$

Thus, $\mathbf{LTL}\,[\mathbf{P}^{=}, \mathbf{Y}]$ and $\mathbf{LTL}\,[\mathbf{P}, \mathbf{Y}]$ define the same class of languages, and the equivalences follow. ■

C.3 Proof of Theorem 2.8

Theorem 2.8. *Any locally testable language is definable in $\mathbf{LTL}\,[\mathbf{P}, \mathbf{Y}]$.*

Proof. Fix $m > 0$. It suffices to show that each of the following basic languages is definable in $\mathbf{LTL}\,[\mathbf{P}, \mathbf{Y}]$:

$$\{\mathtt{w} \in \Sigma^* \mid \mathtt{u} \in \mathbb{I}_m(\mathtt{w})\}, \qquad \mathtt{u} \in \Sigma^m, \tag{46a}$$

$$\{\mathtt{w} \in \Sigma^* \mid P_{m-1}(\mathtt{w}) = \mathtt{u}\}, \qquad \mathtt{u} \in \Sigma^{m-1}, \tag{46b}$$

$$\{\mathtt{w} \in \Sigma^* \mid S_{m-1}(\mathtt{w}) = \mathtt{u}\}, \qquad \mathtt{u} \in \Sigma^{m-1}. \tag{46c}$$

Case 1: m-factors. Let $\mathtt{u} = \mathtt{w}_1 \cdots \mathtt{w}_m \in \Sigma^m$. Then

$$\mathbf{P}\big(\pi_{\mathtt{w}_m} \wedge \mathbf{Y}(\pi_{\mathtt{w}_{m-1}} \wedge \cdots \wedge \mathbf{Y}(\pi_{\mathtt{w}_1}))\big) \tag{47}$$

defines the language $\{\mathtt{w} \in \Sigma^* \mid \mathtt{u} \in \mathbb{I}_m(\mathtt{w})\}$.

Case 2: $(m-1)$-prefixes. Let $\mathtt{u} = \mathtt{w}_1 \cdots \mathtt{w}_{m-1} \in \Sigma^{m-1}$. If $m = 1$, then this language is all of Σ^*, and is defined by $\top$. If $m > 1$, then

$$\mathbf{P}\big(\pi_{\mathtt{w}_{m-1}} \wedge \mathbf{Y}(\pi_{\mathtt{w}_{m-2}} \wedge \cdots \wedge \mathbf{Y}(\pi_{\mathtt{w}_1} \wedge \neg\mathbf{P}\top))\big) \tag{48}$$

defines the language $\{\mathtt{w} \in \Sigma^* \mid P_{m-1}(\mathtt{w}) = \mathtt{u}\}$.

Case 3: $(m-1)$-suffixes. Let $\mathtt{u} = \mathtt{w}_1 \cdots \mathtt{w}_{m-1} \in \Sigma^{m-1}$. If $m = 1$, then this language is all of Σ^*, and is defined by $\top$. If $m > 1$, then

$$\mathbf{Y}\big(\pi_{\mathtt{w}_{m-1}} \wedge \mathbf{Y}(\pi_{\mathtt{w}_{m-2}} \wedge \cdots \wedge \mathbf{Y}(\pi_{\mathtt{w}_1}))\big) \tag{49}$$

defines the language $\{\mathtt{w} \in \Sigma^* \mid S_{m-1}(\mathtt{w}) = \mathtt{u}\}$.

Because $\mathbf{LTL}\,[\mathbf{P}, \mathbf{Y}]$ is closed under Boolean connectives, every locally testable language is definable in $\mathbf{LTL}\,[\mathbf{P}, \mathbf{Y}]$. ■

C.4 Proof of Theorem 2.9

Theorem 2.9. *For any $k \geq 1$, the fragments $\mathbf{LTL}\,[\mathbf{P}]$ and $\mathbf{LTL}\,[\mathbf{Y}^{\leq k}]$ are incomparable.*

Proof. ($\Longrightarrow$) The language $\mathtt{a}\Sigma^*$ is definable in $\mathbf{LTL}\,[\mathbf{P}]$ but not in $\mathbf{LTL}\,[\mathbf{Y}^{\leq k}]$, by the same argument as in Proposition 2.3.

($\Longleftarrow$) Consider the language $L_k \stackrel{\text{def}}{=} \bigcup_{i=0}^{k-1} \Sigma^*\mathtt{a}\Sigma^i$, the set of strings that contain an $\mathtt{a}$ among the last k positions. This language is definable in $\mathbf{LTL}\,[\mathbf{Y}^{\leq k}]$ by $\mathbf{Y}^{\leq k}\pi_{\mathtt{a}}$. However, L_k is not a left-deterministic polynomial, and therefore is not definable in $\mathbf{LTL}\,[\mathbf{P}]$. See §B for the necessary definitions. ■

C.5 Proof of Corollary 2.10

Corollary 2.10. *The fragment $\mathbf{LTL}\,[\mathbf{P}, \mathbf{Y}^{\leq k}]$ is strictly more expressive than each of the fragments $\mathbf{LTL}\,[\mathbf{P}]$ and $\mathbf{LTL}\,[\mathbf{Y}^{\leq k}]$.*

Proof. The inclusions $\mathbf{LTL}\,[\mathbf{P}] \subseteq \mathbf{LTL}\,[\mathbf{P}, \mathbf{Y}^{\leq k}]$ and $\mathbf{LTL}\,[\mathbf{Y}^{\leq k}] \subseteq \mathbf{LTL}\,[\mathbf{P}, \mathbf{Y}^{\leq k}]$ hold because $\mathbf{LTL}\,[\mathbf{P}, \mathbf{Y}^{\leq k}]$ has access to both operators $\mathbf{P}$ and $\mathbf{Y}^{\leq k}$. Strictness follows from Theorem 2.9—the language L_k is definable in $\mathbf{LTL}\,[\mathbf{Y}^{\leq k}]$ but not in $\mathbf{LTL}\,[\mathbf{P}]$, so $\mathbf{LTL}\,[\mathbf{P}] \neq \mathbf{LTL}\,[\mathbf{P}, \mathbf{Y}^{\leq k}]$. Moreover, $\mathtt{a}\Sigma^*$ is definable in $\mathbf{LTL}\,[\mathbf{P}]$ but not in $\mathbf{LTL}\,[\mathbf{Y}^{\leq k}]$, so $\mathbf{LTL}\,[\mathbf{Y}^{\leq k}] \neq \mathbf{LTL}\,[\mathbf{P}, \mathbf{Y}^{\leq k}]$. ■

C.6 Proof of Proposition 2.11

As a preparation, we prove the following lemma.

Lemma C.1. *Assume Σ contains distinct tokens $\mathtt{a}, \mathtt{b}$. Then $\Sigma^*\mathtt{a}$ is not definable in $\mathbf{LTL}\,[\mathbf{Y}^{\leq k}]$ or $\mathbf{LTL}\,[\mathbf{P}, \mathbf{Y}^{\leq k}]$ for any $k > 1$.*

Proof. Since $\mathbf{LTL}\,[\mathbf{Y}^{\leq k}]$ is a subfragment of $\mathbf{LTL}\,[\mathbf{P}, \mathbf{Y}^{\leq k}]$, it suffices to show that $\Sigma^*\mathtt{a}$ is not definable in $\mathbf{LTL}\,[\mathbf{P}, \mathbf{Y}^{\leq k}]$. Let ψ be a formula in $\mathbf{LTL}\,[\mathbf{P}, \mathbf{Y}^{\leq k}]$ of operator depth r. If $r = 0$, then ψ is a Boolean combination of constants and atomic formulas. Since atomic formulas are always false at the end position, such a formula has the same truth value on every string, and therefore does not define $\Sigma^*\mathtt{a}$. So assume $r > 0$. Define

$$\mathtt{w} \stackrel{\text{def}}{=} (\mathtt{a}\mathtt{b}^{k-1})^r\mathtt{a}, \qquad \mathtt{w}' \stackrel{\text{def}}{=} (\mathtt{a}\mathtt{b}^{k-1})^r. \tag{50}$$

Then

$$\mathtt{w} \in \Sigma^*\mathtt{a}, \qquad \mathtt{w}' \notin \Sigma^*\mathtt{a}. \tag{51}$$

Let $N \stackrel{\text{def}}{=} |\mathtt{w}| = kr + 1$. Now, for each s with $0 \leq s \leq r$, and positions $n \in \{1, \ldots, N+1\}$ and $n' \in \{1, \ldots, N\}$, we say that (n, n') is **s-close** if one of the following holds:

1. $n = n'$;
2. $s \geq 1$, $n = N + 1$, and $n' = N$;
3. $n > ks$ and $n' = n - k$.

We claim that for every $0 \leq s \leq r$, every formula ϕ in $\mathbf{LTL}\left[\mathbf{P}, \mathbf{Y}^{\leq k}\right]$ of operator depth at most s, and every s-close pair (n, n'), we have

$$\mathtt{w}, n \models \phi \quad \Longleftrightarrow \quad \mathtt{w}', n' \models \phi. \tag{52}$$

We prove this claim by induction on s.

Base case: $s = 0$**.** Then ϕ is a Boolean combination of constants and atomic formulas. If $n = n'$, the claim is immediate. If $n' = n - k$, then the token at position n of $\mathtt{w}$ is the same as the token at position n' of $\mathtt{w}'$, because both strings consist of repetitions of the block $\mathtt{a}\mathtt{b}^{k-1}$, except that $\mathtt{w}$ has one extra final $\mathtt{a}$, which matches the $\mathtt{a}$ at position $N - k$ in $\mathtt{w}'$. Hence $\mathtt{w}, n$ and $\mathtt{w}', n'$ satisfy the same atomic formulas, and therefore the same Boolean combinations of atomic formulas.

Induction step. Assume the claim holds for $s - 1$, and let ϕ have operator depth at most s.

The Boolean cases are immediate, so it remains to consider the temporal operators.

Case 1: $\phi = \mathbf{P}\psi$, where ψ has operator depth at most $s - 1$.

Suppose $\mathtt{w}, n \models \mathbf{P}\psi$. Then there exists $t < n$ such that $\mathtt{w}, t \models \psi$. We choose $t' < n'$ as follows:

1. If $n = n'$, let $t' = t$.
2. If $n = N + 1$ and $n' = N$, then:
   - if $t < N$, let $t' = t$;
   - if $t = N$, let $t' = N - k$.
3. If $n > ks$ and $n' = n - k$, then:
   - if $t < n - k$, let $t' = t$;
   - if $n - k \leq t < n$, let $t' = t - k$.

In each case, $t' < n'$ and (t, t') is $(s - 1)$-close. By the induction hypothesis,

$$\mathtt{w}, t \models \psi \quad \Longleftrightarrow \quad \mathtt{w}', t' \models \psi, \tag{53}$$

so $\mathtt{w}', n' \models \mathbf{P}\psi$. The converse implication is analogous.

Case 2: $\phi = \mathbf{Y}^{\leq k}\psi$, where ψ has operator depth at most $s - 1$.

Suppose $\mathtt{w}, n \models \mathbf{Y}^{\leq k}\psi$. Then there exists t such that $n - k \leq t < n$ and $\mathtt{w}, t \models \psi$. We choose t' as follows:

1. If $n = n'$, let $t' = t$.
2. If $n = N + 1$ and $n' = N$, then:
   - if $t < N$, let $t' = t$;
   - if $t = N$, let $t' = N - k$.
3. If $n > ks$ and $n' = n - k$, let $t' = t - k$.

In each case,

$$n' - k \leq t' < n', \tag{54}$$

and (t, t') is $(s - 1)$-close. By the induction hypothesis,

$$\mathtt{w}, t \models \psi \quad \Longleftrightarrow \quad \mathtt{w}', t' \models \psi, \tag{55}$$

so $\mathtt{w}', n' \models \mathbf{Y}^{\leq k}\psi$. Again, the converse implication is analogous.

This completes the induction.

Finally, the pair $(N + 1, N)$ is r-close. Hence

$$\mathtt{w} \models \psi \quad \Longleftrightarrow \quad \mathtt{w}, N + 1 \models \psi \quad \Longleftrightarrow \quad \mathtt{w}', N \models \psi \quad \Longleftrightarrow \quad \mathtt{w}' \models \psi. \tag{56}$$

Thus no formula in $\mathbf{LTL}\left[\mathbf{P}, \mathbf{Y}^{\leq k}\right]$ can distinguish $\mathtt{w}$ from $\mathtt{w}'$. Since $\mathtt{w} \in \Sigma^*\mathtt{a}$ and $\mathtt{w}' \notin \Sigma^*\mathtt{a}$, the language $\Sigma^*\mathtt{a}$ is not definable in $\mathbf{LTL}\left[\mathbf{P}, \mathbf{Y}^{\leq k}\right]$. Therefore it is not definable in $\mathbf{LTL}\left[\mathbf{Y}^{\leq k}\right]$ either. ■

Proposition 2.11. *The fragment* $\mathbf{LTL}\left[\mathbf{Y}^{\leq k}\right]$ *is strictly less expressive than* $\mathbf{LTL}\left[\mathbf{Y}\right]$ *for* $k > 1$.

Proof. (**Inclusion.**) Any formula of the form $\mathbf{Y}^{\leq k}\psi$ can be expanded into a finite disjunction of $\mathbf{Y}$-steps by Eq. (3a). Thus, every $\mathbf{LTL}\left[\mathbf{Y}^{\leq k}\right]$ formula translates into an equivalent $\mathbf{LTL}\left[\mathbf{Y}\right]$ formula. Hence $\mathbf{LTL}\left[\mathbf{Y}^{\leq k}\right] \subseteq \mathbf{LTL}\left[\mathbf{Y}\right]$.

(**Strictness.**) Consider the language $\Sigma^*\mathtt{a}$, the set of strings whose last token is $\mathtt{a}$. This language is definable in $\mathbf{LTL}\left[\mathbf{Y}\right]$ by $\mathbf{Y}\pi_{\mathtt{a}}$, but is not definable in $\mathbf{LTL}\left[\mathbf{Y}^{\leq k}\right]$ for any $k > 1$ by Lemma C.1. Therefore $\mathbf{LTL}\left[\mathbf{Y}^{\leq k}\right] \subsetneq \mathbf{LTL}\left[\mathbf{Y}\right]$ for $k > 1$. ■

C.7 Proof of Proposition 2.12

Proposition 2.12. *The fragment* $\mathbf{LTL}\left[\mathbf{P}, \mathbf{Y}^{\leq k}\right]$ *is strictly less expressive than* $\mathbf{LTL}\left[\mathbf{P}, \mathbf{Y}\right]$ *for* $k > 1$.

Proof. The inclusion $\mathbf{LTL}\left[\mathbf{P}, \mathbf{Y}^{\leq k}\right] \subseteq \mathbf{LTL}\left[\mathbf{P}, \mathbf{Y}\right]$ follows from Eq. (3a): every occurrence of $\mathbf{Y}^{\leq k}$ can be rewritten using $\mathbf{Y}$, which is available in $\mathbf{LTL}\left[\mathbf{P}, \mathbf{Y}\right]$. For strictness, consider again the language $\Sigma^*\mathtt{a}$. This language is definable in $\mathbf{LTL}\left[\mathbf{P}, \mathbf{Y}\right]$ but not definable in $\mathbf{LTL}\left[\mathbf{P}, \mathbf{Y}^{\leq k}\right]$ for any $k > 1$ by Lemma C.1. Therefore $\mathbf{LTL}\left[\mathbf{P}, \mathbf{Y}^{\leq k}\right] \subsetneq \mathbf{LTL}\left[\mathbf{P}, \mathbf{Y}\right]$ for $k > 1$. ■

C.8 Proof of Proposition 2.13

Proposition 2.13. *For every* $k > 1$, *if modular predicates* MOD_m^r *are available for some* $m \geq k$, *then* $\mathbf{LTL}\left[\mathbf{Y}^{\leq k}\right]$ *has the same expressive power as* $\mathbf{LTL}\left[\mathbf{Y}\right]$, *and* $\mathbf{LTL}\left[\mathbf{P}, \mathbf{Y}^{\leq k}\right]$ *has the same expressive power as* $\mathbf{LTL}\left[\mathbf{P}, \mathbf{Y}\right]$.

Proof. It suffices to show that $\mathbf{Y}$ can be expressed using $\mathbf{Y}^{\leq k}$ together with modular predicates. Indeed, for any $m \geq k$, we can write

$$\mathbf{Y}\psi = \bigvee_{i=1}^{m} \Big(\mathrm{MOD}_m^i \wedge \mathbf{Y}^{\leq k}(\mathrm{MOD}_m^{i-1} \wedge \psi) \Big), \tag{57}$$

where $i-1$ is understood modulo m. To see this, fix a position n. Exactly one disjunct applies, namely the one with $n \equiv i \pmod{m}$. The formula

$$\mathbf{Y}^{\leq k}(\mathrm{MOD}_m^{i-1} \wedge \psi) \tag{58}$$

then asks whether ψ holds at some position among the previous k positions whose residue modulo m is $i-1$. Now $n-1$ is such a position. Moreover, it is the only such position among the previous k positions: if $n - j \equiv n - 1 \pmod{m}$ for some $1 \leq j \leq k$, then $j - 1 \equiv 0 \pmod{m}$. Since $0 \leq j - 1 < k \leq m$, this implies $j = 1$. Hence the unique position in the previous k positions with residue class $i-1$ is the immediate predecessor $n-1$. ■

D Additional Experimental Results

This section provides additional details of the experimental setup and supplementary results for the formal language recognition experiments in §5.

D.1 Datasets and Artifacts

For the formal language recognition experiments, we generate synthetic datasets. These data consist entirely of artificial sequences over small finite alphabets. For natural language experiments, we use the public WikiText-2 benchmark. We do not collect or annotate new natural language data. Any risks related to personally identifying information or offensive content are therefore inherited from the source dataset rather than introduced by our work. Our implementation is based in part on the Hugging Face codebase (Wolf et al., 2020), but all models in our experiments are trained from scratch and no pre-trained weights are used.

			Local			Hybrid			Global
			$k=1$	$k=2$	$k=4$	$k=1$	$k=2$	$k=4$	–
LTL[S]	$(a(ab)^*b)^*$	mean±std	75.2 ± 0.0	94.7 ± 0.1	98.4 ± 4.1	89.7 ± 8.9	87.4 ± 12.6	83.5 ± 10.7	66.7 ± 10.3
		max	75.2	94.7	99.6	99.5	98.7	92.0	99.8
	$\{a,b,d\}^*a\{c,d\}^*$	mean±std	78.4 ± 0.0	91.8 ± 0.0	99.1 ± 0.0	98.6 ± 2.8	99.9 ± 0.1	90.4 ± 4.9	76.1 ± 11.9
		max	78.4	91.8	99.2	100.0	100.0	96.3	91.2
LTL [P, Y]	$\Sigma^*ab\Sigma^*$	mean±std	71.2 ± 1.0	85.1 ± 0.0	96.2 ± 0.0	99.7 ± 0.8	100.0 ± 0.0	71.9 ± 11.9	56.9 ± 2.2
		max	72.5	85.2	96.2	**100.0**	**100.0**	91.3	61.9
	$(ab)^*$	mean±std	52.0 ± 0.0	54.1 ± 0.1	57.3 ± 2.5	99.8 ± 0.4	95.1 ± 8.9	94.6 ± 12.2	75.2 ± 13.1
		max	52.0	54.1	58.5	**100.0**	**100.0**	**100.0**	97.2
LTL [P]	$\Sigma^*a\Sigma^*b\Sigma^*$	mean±std	71.6 ± 0.1	74.3 ± 0.2	74.9 ± 0.0	100.0 ± 0.1	99.9 ± 0.2	100.0 ± 0.1	99.0 ± 2.1
		max	71.7	74.4	74.9	**100.0**	**100.0**	**100.0**	**100.0**
	$a\Sigma^*$	mean±std	50.1 ± 0.0	49.9 ± 0.0	50.0 ± 0.1	100.0 ± 0.0	100.0 ± 0.0	99.2 ± 1.7	99.4 ± 1.4
		max	50.1	50.1	50.0	**100.0**	**100.0**	**100.0**	**100.0**
LTL [Y]	Σ^*ab	mean±std	**100.0** ± 0.0	99.8 ± 0.1	100.0 ± 0.0	**100.0** ± 0.0	99.8 ± 0.1	80.7 ± 12.3	53.5 ± 0.8
		max	**100.0**	99.8	100.0	**100.0**	100.0	96.9	54.9
	Σ^*a	mean±std	**100.0** ± 0.0	99.7 ± 0.0	100.0 ± 0.0	**100.0** ± 0.0	99.7 ± 0.7	92.7 ± 8.0	58.4 ± 4.1
		max	**100.0**	99.7	100.0	**100.0**	100.0	99.1	67.1

Table 1: Accuracy (%) on the formal-language tasks with NoPE.

			Local			Hybrid			Global
			$k=1$	$k=2$	$k=4$	$k=1$	$k=2$	$k=4$	–
LTL[S]	$(a(ab)^*b)^*$	mean±std	75.2 ± 0.0	62.6 ± 16.3	**100.0** ± 0.0	82.5 ± 12.3	76.5 ± 12.5	74.9 ± 14.6	56.1 ± 4.6
		max	75.2	95.3	**100.0**	95.4	95.5	96.8	63.6
	$\{a,b,d\}^*a\{c,d\}^*$	mean±std	78.4 ± 0.0	91.8 ± 0.1	99.2 ± 0.0	91.0 ± 6.7	95.9 ± 4.7	97.3 ± 5.6	58.4 ± 11.1
		max	78.4	91.8	99.2	99.5	100.0	100.0	87.6
LTL [P, Y]	$\Sigma^*ab\Sigma^*$	mean±std	71.3 ± 0.9	85.1 ± 0.0	96.1 ± 0.1	88.3 ± 10.9	86.4 ± 6.1	89.3 ± 9.2	51.9 ± 0.5
		max	72.5	85.3	96.2	100.0	99.6	**100.0**	52.6
	$(ab)^*$	mean±std	52.0 ± 0.0	54.2 ± 0.0	57.8 ± 1.0	74.0 ± 10.1	76.5 ± 12.4	77.2 ± 13.2	53.3 ± 3.0
		max	52.0	54.2	58.6	92.4	**100.0**	99.7	61.3
LTL [P]	$\Sigma^*a\Sigma^*b\Sigma^*$	mean±std	71.7 ± 0.1	74.3 ± 0.1	74.9 ± 0.0	92.1 ± 7.8	97.6 ± 3.2	95.0 ± 7.0	88.0 ± 7.8
		max	71.7	74.4	75.0	100.0	**100.0**	**100.0**	100.0
	$a\Sigma^*$	mean±std	50.1 ± 0.0	49.9 ± 0.0	49.9 ± 0.0	70.1 ± 10.1	70.3 ± 11.9	77.7 ± 14.2	93.1 ± 8.8
		max	50.1	50.0	50.0	96.3	99.7	**100.0**	99.9
LTL [Y]	Σ^*ab	mean±std	**100.0** ± 0.0	**100.0** ± 0.0	**100.0** ± 0.0	100.0 ± 0.0	**100.0** ± 0.0	**100.0** ± 0.0	52.0 ± 1.1
		max	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	54.6
	Σ^*a	mean±std	**100.0** ± 0.0	**100.0** ± 0.0	**100.0** ± 0.0	**100.0** ± 0.0	**100.0** ± 0.0	**100.0** ± 0.0	53.0 ± 1.9
		max	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	57.0

Table 2: Accuracy (%) on the formal-language tasks with RoPE.

D.2 Experimental Setup

Our setup for formal language recognition follows Li and Cotterell (2025) and Delétang et al. (2023). We use a 5-layer transformer with model dimension 64 and 8 attention heads. Training strings are sampled at lengths up to 40, while test strings range from length 41 to 500. Models are trained for 1,000,000 steps using Adam (Kingma and Ba, 2014) with default parameters and a batch size of 128. At evaluation time, we sample 512 test strings at each length. Each configuration is run with five random seeds and three learning rates: 1×10^{-4}, 3×10^{-4}, and 5×10^{-4}. All experiments are conducted on a single GPU with 24 GB of memory, and each run takes roughly one hour. Our codebase is adapted from Delétang et al. (2023) and Li and Cotterell (2025).

D.3 Full Numerical Results

We report the full numerical results of the formal language recognition experiments in Tabs. 1 to 6.

			Local			Hybrid			Global
			$k=1$	$k=2$	$k=4$	$k=1$	$k=2$	$k=4$	$-$
LTL[**S**]	$(a(ab)^*b)^*$	mean±std	73.9 ± 1.9	74.0 ± 16.2	57.4 ± 5.9	84.5 ± 13.7	74.6 ± 16.2	59.0 ± 15.8	50.1 ± 0.4
		max	75.2	94.3	71.2	99.1	98.8	97.2	50.7
	$\{a,b,d\}^*a\{c,d\}^*$	mean±std	76.9 ± 2.7	86.3 ± 5.0	90.0 ± 5.8	92.8 ± 8.6	90.8 ± 10.8	90.9 ± 9.7	64.5 ± 12.2
		max	78.4	91.6	98.2	99.8	100.0	99.9	87.3
LTL [**P**, **Y**]	$\Sigma^*ab\Sigma^*$	mean±std	70.6 ± 2.9	71.5 ± 9.0	76.8 ± 8.7	71.4 ± 12.4	75.0 ± 16.6	58.0 ± 3.4	51.2 ± 0.2
		max	72.3	85.2	93.0	96.7	100.0	63.8	51.5
	$(ab)^*$	mean±std	51.8 ± 0.3	52.1 ± 1.4	50.9 ± 1.1	86.0 ± 13.6	62.3 ± 13.6	54.1 ± 6.6	50.3 ± 0.2
		max	52.0	54.1	54.2	99.8	**100.0**	76.1	51.1
LTL [**P**]	$\Sigma^*a\Sigma^*b\Sigma^*$	mean±std	71.0 ± 0.5	73.0 ± 1.6	73.6 ± 2.5	87.4 ± 11.2	92.5 ± 10.5	96.1 ± 5.8	92.3 ± 9.5
		max	71.7	74.5	74.9	**100.0**	**100.0**	**100.0**	**100.0**
	$a\Sigma^*$	mean±std	50.0 ± 0.0	50.0 ± 0.0	50.0 ± 0.1	98.8 ± 2.3	97.8 ± 4.4	95.8 ± 6.7	98.6 ± 4.7
		max	50.1	50.1	50.1	**100.0**	**100.0**	**100.0**	**100.0**
LTL [**Y**]	Σ^*ab	mean±std	99.5 ± 1.0	69.6 ± 11.6	60.2 ± 5.9	99.1 ± 1.4	69.9 ± 10.4	64.3 ± 10.7	50.1 ± 0.1
		max	**100.0**	91.4	68.7	**100.0**	93.9	86.1	50.3
	Σ^*a	mean±std	95.7 ± 3.5	89.6 ± 3.4	83.1 ± 3.0	98.3 ± 2.4	88.8 ± 2.3	83.3 ± 3.7	51.6 ± 0.3
		max	**100.0**	97.7	87.3	**100.0**	92.2	94.2	52.0

Table 3: Accuracy (%) on the formal-language tasks with SiPE.

			Local			Hybrid			Global
			$k=1$	$k=2$	$k=4$	$k=1$	$k=2$	$k=4$	$-$
LTL[**S**]	$(a(ab)^*b)^*$	mean±std	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	82.4 ± 16.6	81.9 ± 26.2	75.1 ± 11.0	136.5 ± 57.0
		max	0.0	0.0	0.0	126.0	174.0	104.0	320.0
	$\{a,b,d\}^*a\{c,d\}^*$	mean±std	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	73.2 ± 16.6	203.1 ± 102.1	72.3 ± 7.9	65.1 ± 12.2
		max	0.0	0.0	0.0	114.0	455.0	84.0	100.0
LTL [**P**, **Y**]	$\Sigma^*ab\Sigma^*$	mean±std	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	451.7 ± 86.4	476.4 ± 61.7	61.1 ± 7.7	61.4 ± 10.4
		max	0.0	0.0	0.0	**500.0**	**500.0**	73.0	85.0
	$(ab)^*$	mean±std	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	436.3 ± 92.2	437.3 ± 105.5	358.4 ± 157.8	80.9 ± 25.4
		max	0.0	0.0	0.0	**500.0**	**500.0**	**500.0**	132.0
LTL [**P**]	$\Sigma^*a\Sigma^*b\Sigma^*$	mean±std	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	483.9 ± 40.4	451.5 ± 84.2	452.7 ± 87.1	458.9 ± 75.2
		max	0.0	0.0	0.0	**500.0**	**500.0**	**500.0**	**500.0**
	$a\Sigma^*$	mean±std	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	494.0 ± 15.4	475.0 ± 60.1	432.1 ± 104.5	359.2 ± 127.6
		max	0.0	0.0	0.0	**500.0**	**500.0**	**500.0**	**500.0**
LTL [**Y**]	Σ^*ab	mean±std	**500.0** ± 0.0	41.9 ± 0.3	68.3 ± 18.1	**500.0** ± 0.0	152.4 ± 52.5	62.2 ± 4.9	50.8 ± 5.0
		max	**500.0**	42.0	107.0	**500.0**	236.0	75.0	64.0
	Σ^*a	mean±std	**500.0** ± 0.0	42.0 ± 0.0	77.7 ± 31.5	**500.0** ± 0.0	157.3 ± 70.5	67.0 ± 10.3	52.8 ± 7.0
		max	**500.0**	42.0	172.0	**500.0**	334.0	85.0	74.0

Table 4: Longest perfect length on the formal-language tasks with NoPE.

			Local			Hybrid			Global
			$k=1$	$k=2$	$k=4$	$k=1$	$k=2$	$k=4$	$-$
LTL[**S**]	$(a(ab)^*b)^*$	mean±std	0.0 ± 0.0	0.0 ± 0.0	**500.0** ± 0.0	50.4 ± 2.9	51.2 ± 3.8	51.1 ± 4.2	68.1 ± 18.8
		max	0.0	0.0	**500.0**	56.0	58.0	58.0	106.0
	$\{a,b,d\}^*a\{c,d\}^*$	mean±std	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	47.6 ± 4.1	53.1 ± 6.2	63.9 ± 14.2	45.9 ± 2.5
		max	0.0	0.0	0.0	58.0	67.0	103.0	51.0
LTL [**P**, **Y**]	$\Sigma^*ab\Sigma^*$	mean±std	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	189.6 ± 62.2	190.9 ± 70.1	258.7 ± 140.0	45.7 ± 1.9
		max	0.0	0.0	0.0	355.0	294.0	**500.0**	50.0
	$(ab)^*$	mean±std	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	186.4 ± 67.1	183.5 ± 96.2	145.9 ± 79.6	59.9 ± 17.0
		max	0.0	0.0	0.0	314.0	**500.0**	344.0	110.0
LTL [**P**]	$\Sigma^*a\Sigma^*b\Sigma^*$	mean±std	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	260.0 ± 71.0	356.5 ± 101.9	285.7 ± 119.9	279.3 ± 54.1
		max	0.0	0.0	0.0	410.0	**500.0**	**500.0**	406.0
	$a\Sigma^*$	mean±std	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	134.8 ± 53.9	129.4 ± 50.0	162.1 ± 100.2	156.1 ± 46.9
		max	0.0	0.0	0.0	282.0	234.0	**500.0**	260.0
LTL [**Y**]	Σ^*ab	mean±std	**500.0** ± 0.0	**500.0** ± 0.0	**500.0** ± 0.0	490.9 ± 33.9	**500.0** ± 0.0	**500.0** ± 0.0	45.1 ± 0.8
		max	**500.0**	**500.0**	**500.0**	**500.0**	**500.0**	**500.0**	46.0
	Σ^*a	mean±std	**500.0** ± 0.0	**500.0** ± 0.0	**500.0** ± 0.0	**500.0** ± 0.0	**500.0** ± 0.0	**500.0** ± 0.0	45.6 ± 2.2
		max	**500.0**	**500.0**	**500.0**	**500.0**	**500.0**	**500.0**	51.0

Table 5: Longest perfect length on the formal-language tasks with RoPE.

			Local			Hybrid			Global
			$k=1$	$k=2$	$k=4$	$k=1$	$k=2$	$k=4$	$-$
LTL[S]	$(a(ab)^*b)^*$	mean±std	0.0 ± 0.0	0.0 ± 0.0	42.0 ± 0.0	64.9 ± 23.5	63.9 ± 21.4	56.0 ± 20.1	42.0 ± 0.0
		max	0.0	0.0	42.0	122.0	102.0	90.0	42.0
	$\{a,b,d\}^*a\{c,d\}^*$	mean±std	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	56.0 ± 7.7	65.9 ± 34.3	51.5 ± 14.4	41.1 ± 0.3
		max	0.0	0.0	0.0	71.0	150.0	77.0	42.0
LTL [P, Y]	$\Sigma^*ab\Sigma^*$	mean±std	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	50.9 ± 12.4	95.8 ± 86.2	41.6 ± 0.5	41.0 ± 0.0
		max	0.0	0.0	0.0	84.0	330.0	42.0	41.0
	$(ab)^*$	mean±std	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	101.5 ± 41.8	121.5 ± 111.7	65.3 ± 38.4	42.4 ± 0.8
		max	0.0	0.0	0.0	176.0	**500.0**	176.0	44.0
LTL [P]	$\Sigma^*a\Sigma^*b\Sigma^*$	mean±std	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	154.3 ± 143.2	183.0 ± 175.8	151.4 ± 154.6	218.1 ± 191.9
		max	0.0	0.0	0.0	**500.0**	**500.0**	**500.0**	**500.0**
	$a\Sigma^*$	mean±std	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	420.9 ± 99.8	420.3 ± 96.0	336.5 ± 132.3	446.9 ± 107.3
		max	0.0	0.0	0.0	**500.0**	**500.0**	**500.0**	**500.0**
LTL [Y]	Σ^*ab	mean±std	323.2 ± 174.5	41.0 ± 0.0	41.2 ± 0.4	281.0 ± 179.9	41.0 ± 0.0	41.2 ± 0.4	41.0 ± 0.0
		max	**500.0**	41.0	42.0	**500.0**	41.0	42.0	41.0
	Σ^*a	mean±std	198.3 ± 154.9	41.3 ± 0.5	41.0 ± 0.0	218.9 ± 148.8	41.1 ± 0.3	41.0 ± 0.0	41.0 ± 0.0
		max	**500.0**	42.0	41.0	**500.0**	42.0	41.0	41.0

Table 6: Longest perfect length on the formal-language tasks with SiPE.