Hardness of Learning Regular Languages in the Next Symbol Prediction Setting

Satwik Bhattamishra[1*] Phil Blunsom[2] Varun Kanade[1]

[1]University of Oxford [2]Cohere

Abstract

We study the learnability of languages in the *Next Symbol Prediction* (NSP) setting, where a learner receives only positive examples from a language together with, for every prefix, (i) whether the prefix itself is in the language and (ii) which next symbols can lead to an accepting string. This setting has been used in prior works to empirically analyze neural sequence models, and additionally, we observe that efficient algorithms for the NSP setting can be used to learn the (truncated) support of language models. We formalize the setting so as to make it amenable to PAC-learning analysis. While the setting provides a much richer set of labels than the conventional classification setting, we show that learning concept classes such as DFAs and Boolean formulas remains computationally hard. The proof is via a construction that makes almost all additional labels uninformative, yielding a reduction from the conventional learning problem to learning with NSP labels. Under cryptographic assumptions, the reduction implies that the problem of learning DFAs is computationally hard in the NSP setting.

1 Introduction

We formalize and study the problem of learnability of languages in the *Next Symbol Prediction* (NSP) setting, in which a learner receives only *positive* strings from a target language, together with rich supervision for every prefix: a membership bit indicating whether the prefix itself is in the language and a vector of "continuation" bits indicating which next symbols admit some accepting continuation. A prediction is correct only if the hypothesis matches *all* membership and continuation labels at *every* prefix of the example.

This setup has a natural interpretation in the context of language models: when decoding with top-p (Holtzman et al., 2020), top-k, or min-p (Minh et al., 2025) sampling, the per-prefix continuation set is precisely the set of admissible next tokens, and termination corresponds to allowing the end-of-sequence token. In particular, positive-only NSP supervision is naturally obtained from black-box models and avoids the need for inventing artificial distributions over negative strings; at the same time, the requirement for correct predictions at every prefix makes the task challenging.

Additionally, while NSP has been widely used to *evaluate* sequence models on formal-language benchmarks (see e.g. Bhattamishra et al. (2020), Ebrahimi et al. (2020), Gers and Schmidhuber (2001), Suzgun et al. (2019) and references therein), the learnability of languages under NSP and its relationship to conventional binary classification has not been established.

Our contributions. We give a PAC-style formulation of learning using NSP labels and analyze the learnability of classes such as DFAs and Boolean formulas. We prove that NSP supervision does *not* remove the key *computational barriers* for learning regular languages. The key technical argument

*Corresponding author. Contact: satwik.bmishra@cs.ox.ac.uk

is a construction that renders all but one continuation label uninformative, allowing a reduction to well-known hardness results (Kearns and Valiant, 1994). Thus, even with the richer labels, efficient (improper) learning remains computationally intractable (under standard cryptographic assumptions). Our results suggest that while NSP labels may offer some benefit, they do not, in general, circumvent computational hardness.[1]

2 Problem Definition

Notation. A deterministic finite automaton (DFA) is a tuple $A = (Q, \Sigma, \delta, q_0, F_A)$ with finite state set Q, alphabet Σ, transition function $\delta : Q \times \Sigma \to Q$, start state q_0, and a subset of accepting states $F_A \subseteq Q$. The language of A is $L_A \subseteq \Sigma^*$; write $A(x) = L_A(x) \in \{0, 1\}$ and $|A|$ denotes the number of states $|Q|$. Fix an order $\Sigma = \{\sigma_1, \ldots, \sigma_{|\Sigma|}\}$. For $x = w_1 \cdots w_N$, let the length-n prefix be $x_{:n} := w_1 \cdots w_n$ for $0 \le n \le N$. We use q_{dead} for a dead state with $\delta(q_{\text{dead}}, \sigma) = q_{\text{dead}}$ for all $\sigma \in \Sigma$; if present, such a state is unique in a minimal DFA. We use DFA_n to denote the class of DFAs with at most n states.

2.1 Next Symbol Prediction (NSP) Setting

For a language L, for any string x and any $\sigma \in \Sigma$, we define $\varphi_L(x, \sigma)$ as follows:

$$\varphi_L(x, \sigma) := \begin{cases} 1, & \text{if } \exists s \in \Sigma^* \text{ such that } x \cdot \sigma \cdot s \in L \\ 0, & \text{otherwise.} \end{cases}$$

When the language L is regular, then it is equivalent to computing whether $x \cdot \sigma$ leads to a dead state in its canonical DFA. For any minimal DFA, let $q = \delta(q_0, x)$, then $\varphi(x, \sigma) = \varphi(q, \sigma) = 0$ if $\delta(q, \sigma) = q_{\text{dead}}$ and 1 otherwise. The continuation vector at q is defined as $\varphi(q) = [\varphi(q, \sigma_1), \ldots, \varphi(q, \sigma_{|\Sigma|})] \in \{0, 1\}^{|\Sigma|}$, and for $x \in \Sigma^*$ we use $\varphi(x) := \varphi(\delta(q_0, x))$ and $\varphi(x, \sigma) := \varphi(\delta(q_0, x), \sigma)$.

Fix a target language $L \subseteq \Sigma^*$. An example consists of a positive string $x = w_1 \cdots w_N \in L$ together with labels for each prefix $x_{:n}$ ($0 \le n \le N$): its membership label $L(x_{:n}) \in \{0, 1\}$ and the continuation labels $(\varphi(x_{:n}, \sigma))_{\sigma \in \Sigma} \in \{0, 1\}^{|\Sigma|}$, where $\varphi(x_{:n}, \sigma) = 1$ means that $x_{:n} \cdot \sigma$ has some accepting continuation in L and $\varphi(x_{:n}, \sigma) = 0$ implies that $x_{:n} \cdot \sigma \cdot s \notin L$ for every suffix s. Collecting these across all prefixes gives

$$f_L(x) \;:=\; \Big((\varphi(x_{:n}, \sigma_i))_{i=1}^{|\Sigma|},\, L(x_{:n}) \Big)_{n=0}^{N} \;\in\; \{0, 1\}^{(|\Sigma|+1)(N+1)},$$

with coordinates ordered by nondecreasing prefix length n.

Predictors and loss. Although the NSP setting is not restricted to regular languages, it is more intuitive to initially think of the setting in terms of DFAs. We instantiate predictors via automata. For a DFA A, define

$$f_A(x) \;:=\; \Big((\varphi(x_{:n}, \sigma_i))_{i=1}^{|\Sigma|},\, L_A(x_{:n}) \Big)_{n=0}^{|x|}, \qquad \forall x \in \Sigma^*, \quad f_A(x) \in \{0, 1\}^{(|\Sigma|+1)(|x|+1)}.$$

Let $A^\star$ denote an unknown target DFA. On a single example x, the error is defined as,

$$\text{err}\Big(f_A(x), f_{A^\star}(x)\Big) \;:=\; \mathbb{I}[f_A(x) \neq f_{A^\star}(x)].$$

[1] In an earlier workshop paper (Bhattamishra et al., 2023), we argued that polynomial-time learning is feasible in the NSP setting. We later found that the claim was *incorrect*; in this work, we prove that learning in the NSP setting is computationally hard.

Note that for a given input string x, the error $\mathrm{err}(f_A(x), f_{A^\star}(x)) = 0$ precisely when all $(|\Sigma|+1)(|x|+1)$ labels are correct. For a distribution $\mathcal{D}$ supported on positive examples L, the expected NSP error is

$$\mathcal{L}_{\mathrm{NSP}}(f_A; f_{A^\star}, \mathcal{D}) := \mathop{\mathbb{E}}_{x\sim\mathcal{D}}\Big[\,\mathrm{err}\Big(f_A(x), f_{A^\star}(x)\Big)\,\Big].$$

Remark. Note that, although the NSP setting only involves positive examples, some information about the negative examples can be obtained from the membership and continuation labels of the prefixes. Since a predictor needs to predict all $(|\Sigma|+1)(|x|+1)$ correctly for an example x, the setting is more stringent in the sense that a random predictor will typically have near-zero accuracy as opposed to 50% accuracy in the conventional classification setting. We will refer to labels described above of the form $f_A(x) \in \{0,1\}^{(|\Sigma|+1)(|x|+1)}$ as the NSP labels. We refer to the setting with binary classifiers $f(x) \in \{0,1\}$ as the conventional classification setting.

3 Hardness of Learning

We study efficient PAC learnability in the NSP setting. An algorithm $\mathcal{A}$ is an efficient PAC learner for a class $\mathcal{F}$ if for every $f \in \mathcal{F}$ and every distribution $\mathcal{D}$ supported on positive examples, $\mathcal{A}$ runs in polynomial time on NSP–labeled inputs and outputs $\hat{f}$ such that, with probability at least $1-\delta$, $\mathcal{L}_{\mathrm{NSP}}(\hat{f}; f, \mathcal{D}) \le \epsilon$.

Note that the continuation labels can be highly informative for certain classes. Consider *Conjunctions*: Boolean monomials $f : \{0,1\}^N \to \{0,1\}$, e.g., $f(z) = z_2 \wedge \bar{z}_4$. In the conventional classification model, learning Conjunctions is known to require $\Theta(N)$ labeled examples. In the NSP model, one positive example $x \in f^{-1}(1)$ suffices. For each prefix $x_{:k}$, the label $\varphi(x_{:k}, 0)$ equals 0 if and only if the literal z_{k+1} appears in the target conjunction; likewise, $\varphi(x_{:k}, 1) = 0$ if and only if the literal $\bar{z}_{k+1}$ appears. Thus, by reading the continuation labels across the N prefixes, the learner recovers exactly which literals are present and hence identifies the target monomial from a single positive NSP example. While the NSP labels may provide a statistical advantage for some classes, we show that in the general case, they are not enough to remove the computational barriers for learning DFAs.

We relate learnability in the NSP setting to standard PAC learning for acyclic DFAs that accept only strings of a fixed length. Throughout this section, we work over the binary alphabet $\Sigma = \{0,1\}$.

Notation for classes. For $N \in \mathbb{N}$ and a polynomial $p(\cdot)$, define

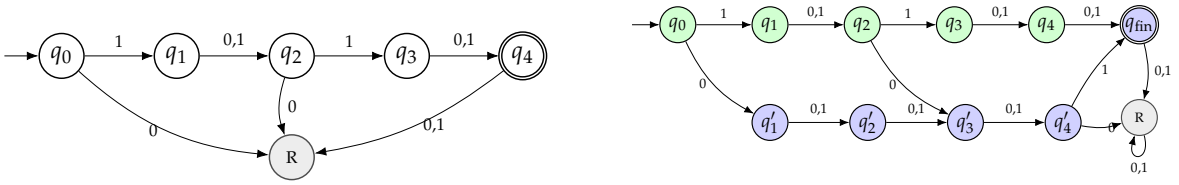
$$\mathrm{ADFA}^N_{p(N)} := \Big\{A : \ A \text{ is a DFA with at most } p(N) \text{ states and } L_A \subseteq \{0,1\}^N\Big\},$$

and write $\mathrm{ADFA}_{p(\cdot)} := \bigcup_{N\ge 1} \mathrm{ADFA}^N_{p(N)}$. As above, DFA_n denotes the class of DFAs with at most n states. Under the convention that the transition function is total, any minimal $A \in \mathrm{ADFA}^N_{p(N)}$ has a dead (sink) state with self-loops on both input symbols. If instead one allows a partial transition function δ, interpreting undefined transitions as implicit moves to a dead state, then the automata in $\mathrm{ADFA}^N_{p(N)}$ are acyclic in the usual sense.

Background. Kearns and Valiant (1994) show that, for a suitable polynomial p, weak PAC learning of $\mathrm{ADFA}_{p(\cdot)}$ in the conventional classification (binary-label) model is as hard as inverting certain cryptographic functions.

Theorem 3.1 (Kearns and Valiant (1994))**.** *There exists a polynomial $p(\cdot)$ such that the problems of inverting RSA, factoring Blum integers, etc., are probabilistic polynomial-time reducible to weakly learning $\mathrm{ADFA}_{p(\cdot)}$ in the standard PAC setting.*

We prove that an efficient PAC algorithm for $\mathrm{ADFA}^N_{p(N)}$ in the NSP setting would yield an efficient PAC algorithm for $\mathrm{ADFA}^N_{p(N)}$ in the conventional classification setting. Together with Theorem 3.1, this



(a) DFA for $z_1 \wedge z_3$, $x \in \{0,1\}^4$.

(b) Transformed DFA $(z_1 \wedge z_3) \vee z_5$, $x \in \{0,1\}^5$.

Figure 1: Transformation from Section 3.1. In (b), green states are from the original DFA; blue states $q'_1, \ldots, q'_4$ ensure every prefix of length $< N$ has continuation labels $[1,1,0]$, and q'_4 routes to accept on input 1 (to the dead state on 0).

implies cryptographic hardness for NSP learning of these Boolean Acyclic DFAs and consequently the general class of DFAs.

3.1 Boolean Acyclic DFAs

Fix $N \geq 1$ and a target DFA $A = (Q, \Sigma, \delta, q_0, F)$ with $L_A \subseteq \{0,1\}^N$. We first record a basic structural property of *minimal* DFAs for fixed-length languages.

Lemma 3.2 (Unique depth in minimal $\text{ADFA}^N_{p(N)}$)**.** *If A is minimal and $L_A \subseteq \{0,1\}^N$, then every state $q \in Q \setminus \{q_{\text{dead}}\}$ is reachable by strings of exactly one length $\ell(q) \in \{0,1,\ldots,N\}$. Consequently, every transition increases depth by one: if $\delta(q,\sigma) = q'$ with $q \neq q_{\text{dead}}$ and $q' \neq q_{\text{dead}}$, then $\ell(q') = \ell(q) + 1$. Acceptance occurs only at depth N.*

Proof. Let $q \in Q \setminus \{q_{\text{dead}}\}$ be reachable both by a string of length ℓ and by a string of length ℓ' with $\ell < \ell'$. The residual language at q is $R(q) := \{\, s \in \Sigma^* : \delta(q,s) \in F \,\}$. If q is reached after ℓ symbols, then every $s \in R(q)$ must have length exactly $N - \ell$; if q is reached after ℓ' symbols, then every $s \in R(q)$ must have length exactly $N - \ell'$. Since $N - \ell \neq N - \ell'$, these sets are disjoint; hence $R(q)$ must be empty, contradicting $q \neq q_{\text{dead}}$ in a minimal DFA. Thus, each non-dead state has a unique depth $\ell(q)$. Any transition consumes one symbol, so $\ell(q') \leq \ell(q) + 1$; equality must hold by uniqueness of depth. Finally, if $q \in F$ had $\ell(q) \neq N$, then L_A would contain strings of length other than N, contrary to the assumption. □

By Lemma 3.2, we may index non-dead states by their unique depth $\ell(q) \in \{0,\ldots,N\}$ (the dead state has no depth).

Padding by one bit. We construct from A a DFA $A^{\oplus}$ over length-$N{+}1$ inputs whose NSP labels are uninformative before depth N, while at depth N the continuation bit for symbol 0 recovers A's label.

More precisely, we will construct a DFA $A^{\oplus}$ with at most $|A| + N + 1$ states such that: (i) the NSP labels of any prefix of length $< N$ will be $(1,1,0)$, and (ii) crucially, the NSP label for a string $u \in \Sigma^N$ of length N will be $(A(u),1,0)$. The ADFA $A^{\oplus}$ accepts all strings x of length $N+1$ ending with 1 and some ending with 0 depending on $A(x_{:N})$. Thus, with respect to $A^{\oplus}$, predicting the first continuation bit (corresponding to symbol 0) is equivalent to predicting membership in L_A.

Lemma 3.3 (Padded Construction)**.** *From a Boolean Acyclic DFA A we can construct, in time polynomial in $|A| + N$, a DFA $A^{\oplus}$ with $L_{A^{\oplus}} \subseteq \{0,1\}^{N+1}$ and*

$$\text{for } u \in \{0,1\}^N \text{ and } b \in \{0,1\}: \quad u \cdot b \in L_{A^{\oplus}} \iff \Big(A(u) = 1\Big) \text{ or } \Big(b = 1\Big). \tag{1}$$

Moreover, for every prefix y with $|y| < N$,

$$\Big(\varphi(y,0),\ \varphi(y,1),\ L_{A^{\oplus}}(y)\Big) = (1,1,0),$$

and for every $u \in \{0,1\}^N$,

$$\Big(\varphi(u,0),\ \varphi(u,1),\ L_{A^\oplus}(u)\Big) = \Big(A(u),\ 1,\ 0\Big).$$

The construction adds at most $N{+}1$ *states.*

Proof. Minimize the DFA A if it is not minimal. Create new states $q'_1, \ldots, q'_N$ and a new accepting state q_{fin}. For $1 \le i < N$ set

$$\delta_{A^\oplus}(q'_i, 0) = q'_{i+1}, \qquad \delta_{A^\oplus}(q'_i, 1) = q'_{i+1},$$

and at $i = N$ set

$$\delta_{A^\oplus}(q'_N, 1) = q_{\text{fin}}, \qquad \delta_{A^\oplus}(q'_N, 0) = q_{\text{dead}}.$$

From q_{fin} send both symbols to q_{dead} (so acceptance occurs only at length $N{+}1$).

Now modify A to create $A^\oplus$ as follows. For each non-dead state q with $\ell(q) < N$ and each $\sigma \in \{0,1\}$:

- If $\delta_A(q,\sigma) = q_{\text{dead}}$ in A, set $\delta_{A^\oplus}(q,\sigma) = q'_{\ell(q)+1}$ (redirect the dead transition into the chain).
- Otherwise set $\delta_{A^\oplus}(q,\sigma) = \delta_A(q,\sigma)$.

For each state q at depth N set

$$\delta_{A^\oplus}(q,1) = q_{\text{fin}}, \qquad \delta_{A^\oplus}(q,0) = \begin{cases} q_{\text{fin}}, & q \in F_A, \\ q_{\text{dead}}, & q \notin F_A. \end{cases}$$

See Figure 1 for a simple example construction for a Boolean Acyclic DFA computing a conjunction.

All transitions out of q_{dead} point to q_{dead}, i.e., $\delta_{A^\oplus}(q_{\text{dead}}, \sigma) = q_{\text{dead}}$ for both symbols. (If A recognizes the empty language, then $Q = \{q_{\text{dead}}\}$; in this case we additionally introduce a fresh start state $\tilde{q}_0$ of depth 0 with $\delta_{A^\oplus}(\tilde{q}_0, 0) = \delta_{A^\oplus}(\tilde{q}_0, 1) = q'_1$ and take $\tilde{q}_0$ as the start state; the conclusions below still hold.)

By construction, acceptance can occur only at q_{fin} after exactly $N{+}1$ symbols, so $L_{A^\oplus} \subseteq \{0,1\}^{N+1}$. The rule at depth N gives (1). For any prefix y with $|y| < N$, either an original transition still has an accepting continuation, or a dead transition is redirected to the chain $q'_{|y|+1} \to \cdots \to q'_N \to q_{\text{fin}}$ (taking the last symbol 1). Hence $\varphi(y,0) = \varphi(y,1) = 1$ and $L_{A^\oplus}(y) = 0$ for any y with $|y| < N$. At depth N, for every $u \in \{0,1\}^N$ our construction enforces

$$\delta_{A^\oplus}\Big(\delta_{A^\oplus}(q_0,u),1\Big) = q_{\text{fin}} \quad \text{and} \quad \delta_{A^\oplus}\Big(\delta_{A^\oplus}(q_0,u),0\Big) = \begin{cases} q_{\text{fin}}, & \delta_A(q_0,u) \in F_A, \\ q_{\text{dead}}, & \delta_A(q_0,u) \notin F_A, \end{cases}$$

so $\varphi(u,1) = 1$, $\varphi(u,0) = A(u)$, and $L_{A^\oplus}(u) = 0$, which is exactly

$$\Big(\varphi(u,0),\ \varphi(u,1),\ L_{A^\oplus}(u)\Big) = \Big(A(u),\ 1,\ 0\Big).$$

□

Reduction from standard learning to NSP **learning.** Let D be any distribution on $\{0,1\}^N$. Define the *padded* distribution $D^\oplus$ on $\{0,1\}^{N+1}$ by sampling $u \sim D$ and returning $x := u \cdot 1$. By (1), $x \in L_{A^\oplus}$ for every u, so $D^\oplus$ is supported on positive examples as required by the NSP setting. Moreover, given

a labeled standard example (u, y) with $y = A(u)$, the full NSP label vector $f_{A^\oplus}(x)$ for $x = u \cdot 1$ is computable from (u, y):

$$\begin{aligned}
&\text{for } 0 \le \ell < N: \quad \Big(\varphi(x_{:\ell}, 0), \varphi(x_{:\ell}, 1), L(x_{:\ell})\Big) = (1, 1, 0),\\
&\text{for } \ell = N: \quad \Big(\varphi(x_{:N}, 0), \varphi(x_{:N}, 1), L(x_{:N})\Big) = (y, 1, 0),\\
&\text{for } \ell = N{+}1: \quad \Big(\varphi(x_{:N+1}, 0), \varphi(x_{:N+1}, 1), L(x_{:N+1})\Big) = (0, 0, 1).
\end{aligned}$$

Here $x_{:\ell}$ denotes the length-ℓ prefix of the padded string x.

Theorem 3.4. *Fix N and a polynomial $p(\cdot)$. If $\mathrm{ADFA}^N_{p(N)}$ is efficiently PAC-learnable in the* NSP *setting from positive examples, then $\mathrm{ADFA}^N_{p(N)}$ is efficiently PAC-learnable in the conventional classification (binary-label) setting.*

Proof. Let $\mathcal{A}_{\mathrm{NSP}}$ be an efficient NSP learner for $\mathrm{ADFA}^N_{p(N)}$. From i.i.d. labeled samples $(u^{(i)}, y^{(i)})$ with $y^{(i)} = A(u^{(i)})$, form positive NSP examples $\Big(x^{(i)}, f_{A^\oplus}(x^{(i)})\Big)$ where $x^{(i)} = u^{(i)} \cdot 1$ using the rule above, and feed them to $\mathcal{A}_{\mathrm{NSP}}$. Let $\hat{f}$ be the returned predictor so that, with probability at least $1 - \delta$,

$$\mathcal{L}_{\mathrm{NSP}}\Big(\hat{f}; f_{A^\oplus}, D^\oplus\Big) = \mathop{\mathbb{E}}_{u \sim D}\Big[\mathbb{I}\Big[\hat{f}(u \cdot 1) \neq f_{A^\oplus}(u \cdot 1)\Big]\Big] \le \epsilon.$$

Define a standard classifier $h : \{0,1\}^N \to \{0,1\}$ by

$$h(u) := \text{ the bit predicted by } \hat{f} \text{ for } \varphi(x_{:N}, 0) \text{ on } x = u \cdot 1.$$

Whenever $\mathbb{I}\Big[\hat{f}(x) = f_{A^\oplus}(x)\Big] = 1$, Lemma 3.3 gives $h(u) = A(u)$. Therefore

$$\Pr_{u \sim D}\Big[h(u) \neq A(u)\Big] \le \Pr_{u \sim D}\Big[\hat{f}(u \cdot 1) \neq f_{A^\oplus}(u \cdot 1)\Big] \le \epsilon,$$

yielding an efficient PAC learner in the standard setting. The state complexity of $A^\oplus$ is at most $p(N) + N + 1$, preserving polynomial time complexity. □

Corollary 3.4.1 (Cryptographic hardness for NSP learning of fixed-length acyclic DFAs)**.** *Under the assumptions of Theorem 3.1, there is no polynomial-time weak learner for $\mathrm{ADFA}_{p(\cdot)}$ in the* NSP *setting. Otherwise, Theorem 3.4 would yield a polynomial-time weak learner in the standard setting, contradicting Theorem 3.1.*

Boolean formulas. Exactly the same padding idea applies to Boolean formulas. Let F be any class of formulas $f : \{0,1\}^N \to \{0,1\}$ and define

$$f'(z_1, \ldots, z_{N+1}) := f(z_1, \ldots, z_N) \vee z_{N+1}.$$

Given a labeled standard example (u, y) with $y = f(u)$, set $x := u \cdot 1$. The NSP labels for the positive string x under f' are then computable from (u, y):

$$\text{for } \ell < N: \ (1,1,0), \qquad \text{for } \ell = N: \ (y,1,0), \qquad \text{for } \ell = N{+}1: \ (0,0,1),$$

with the same ordering (continuations first, then membership) as in §2. Thus, an efficient NSP learner for $F' := \{f' : f \in F\}$ yields, by reading the depth-N continuation bit for symbol 0, an efficient PAC learner for F in the standard setting. In particular, cryptographic hardness results for learning formulas (e.g., via NC^1) carry over to the NSP setting by this reduction.

3.2 Discussion

The result shows that richer supervision via NSP does not circumvent the computational barrier for learning regular languages: under standard assumptions, there is no polynomial-time weak learner for $\mathrm{ADFA}_{p(\cdot)}$ and hence for DFAs even in the NSP setting. This remains true when the learner receives both positive and negative examples with NSP labels, showing that such additional supervision does not mitigate these barriers. The hardness is *improper*: it rules out efficient learning even when the hypothesis need not be a DFA, and thus applies to neural models trained to match NSP labels.

References

Satwik Bhattamishra, Kabir Ahuja, and Navin Goyal. On the Ability and Limitations of Transformers to Recognize Formal Languages. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7096–7116, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.576. URL https://aclanthology.org/2020.emnlp-main.576.

Satwik Bhattamishra, Phil Blunsom, and Varun Kanade. The next symbol prediction problem: PAC-learning and its relation to language models. In *NeurIPS 2023 Workshop on Mathematics of Modern Machine Learning*, 2023. URL https://openreview.net/forum?id=3vkeYFEZxW.

Javid Ebrahimi, Dhruv Gelda, and Wei Zhang. How can self-attention networks recognize Dyck-n languages? In Trevor Cohn, Yulan He, and Yang Liu, editors, *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4301–4306, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.384. URL https://aclanthology.org/2020.findings-emnlp.384/.

Felix A Gers and E Schmidhuber. Lstm recurrent networks learn simple context-free and context-sensitive languages. *IEEE Transactions on Neural Networks*, 12(6):1333–1340, 2001.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=rygGQyrFvH.

Michael Kearns and Leslie Valiant. Cryptographic limitations on learning boolean formulae and finite automata. *Journal of the ACM (JACM)*, 41(1):67–95, 1994.

Nguyen Nhat Minh, Andrew Baker, Clement Neo, Allen G Roush, Andreas Kirsch, and Ravid Shwartz-Ziv. Turning up the heat: Min-p sampling for creative and coherent LLM outputs. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=FBkpCyujtS.

Mirac Suzgun, Yonatan Belinkov, Stuart Shieber, and Sebastian Gehrmann. LSTM networks can perform dynamic counting. In *Proceedings of the Workshop on Deep Learning and Formal Languages: Building Bridges*, pages 44–54, Florence, August 2019. Association for Computational Linguistics. doi: 10.18653/v1/W19-3905. URL https://www.aclweb.org/anthology/W19-3905.