# Learning of Tree Automata applied to Neural Language Acceptors

**J. da Silva** JD229475@ORT.EDU.UY
**S. Yovine** YOVINE@ORT.EDU.UY
*Facultad de Ingeniería, Universidad ORT Uruguay, Montevideo, Uruguay*

## Abstract

We extract visibly pushdown grammars (VPGs) from neural language models in a fully black-box setting. Our work builds on the VPL* framework of Barbot et al. (2021), which learns VPGs from recurrent networks by exploiting access to the target's internal state. However, unlike VPL*, our algorithm learns a tree automaton describing discovered behaviors of the target even outside the most permissive VPG, which results in a larger learnable space. Our approach applies unchanged to transformer language models and, more generally, to any acceptor exposing only binary decisions. The preliminary experimental evaluation on several cases are promising.

**Keywords:** Active Learning, Tree Automata, Visible Pushdown Languages, Transformers

## 1. Introduction

The deployment of neural language models in domains where failures carry real consequences demands more than predictive accuracy, it requires the ability to inspect their behavior. A model that cannot be examined cannot be trusted in safety-critical settings, and post-hoc explanations of individual predictions do little to characterize a model as a whole. Grammatical inference explores another route: extract a symbolic surrogate model that captures the language the target model accepts, and reason about the surrogate. For models trained on inputs with no recursive structure, regular surrogates suffice, and a substantial body of work extracts deterministic finite automata from recurrent networks (Angluin, 1987; Mayr and Yovine, 2018; Weiss et al., 2018). Many languages of practical interest; like arithmetic expressions, balanced parentheses or the syntax of programming languages; are not regular but exhibit nesting, and the visibly pushdown languages (Alur and Madhusudan, 2004) are the natural class for them, strictly above regular.

The state of the art for active learning of visibly pushdown grammars (VPGs) from neural acceptors is the VPL* framework of Barbot et al. (2021), which operates over recurrent networks. Its equivalence oracle runs an A* search over the network's internal state, presupposing both white-box access to the model and a recurrent architecture. Membership queries in their framework go through the construction of the most permissive VPG (called BParse) with the target's ranked alphabet, called *BParse*, which by definition restricts the learner's view to well-matched inputs encoded as BParse-shaped trees. Thus, the extracted VPG describes the target only on this domain, being silent about whatever the model does on the rest of the input space.

A complementary line of work learns VPGs directly from sample strings rather than from neural acceptors. Jia and Tan (2024) propose V-Star, which adapts Angluin's L* to actively learn visibly pushdown grammars, inferring nesting structure from program inputs such as JSON, XML and S-expressions. Like VPL*, the model V-Star learns is always a

© J. da Silva & S. Yovine.

visibly pushdown grammar, hence a VPL describing only well-matched inputs. Our learner instead extracts a tree automaton over the structured alphabet, a strictly more expressive object that can capture target behavior outside any VPL and produces witnesses when the target accepts ill-matched sequences that no VPG could.

This work makes three contributions over Barbot et al. (2021). First, we replace their white-box equivalence oracle with PAC sampling over trees, yielding a fully black-box variant of VPL$^*$ that applies to any acceptor exposing only binary decisions. This allows applying the resulting learning algorithm to transformer language models, a family for which the A$^*$ oracle originally proposed in VPL$^*$ is not available. Besides, we drop the BParse gate on membership queries and sample trees directly from the structured alphabet's language, and measure the errors incurred by VPL$^*$ when avoiding querying the target on non-well formed sequences. Finally, building on the previous step, we develop a learner that recovers a tree automaton that describes the target on *all* trees over the structured alphabet. When the target is not a VPG, the BParse-consistent observation table is projected to a VPG that agrees with the one produced by VPL$^*$. We stress the fact that our implementation of VPL$^*$ is the first publicly available one.

## 2. Background

Fix a structured alphabet $\Sigma = \Sigma_{push} \cup \Sigma_{pop} \cup \Sigma_{int}$ partitioning $\Sigma$ into push, pop, and internal letters. The well-formed words $W_\Sigma \subseteq \Sigma^*$ are those in which every push has a matching pop in the standard bracket sense. Visibly pushdown languages (Alur and Madhusudan, 2004) are the subsets of $W_\Sigma$ recognized by visibly pushdown grammars (VPGs), and we treat them throughout via their tree representation.

To each well-formed word, the BParse construction of Barbot et al. (2021) assigns a tree over the ranked alphabet $\Gamma$ with $\Gamma_0 = \{\square\}$, $\Gamma_1 = \Sigma_{pop} \cup \Sigma_{int}$, and $\Gamma_2 = \Sigma_{push}$. The encoding $\varphi : W_\Sigma \to \mathsf{Trees}(\Gamma)$ is defined inductively by $\varphi(\varepsilon) = \square$, $\varphi(c\,w) = c(\varphi(w))$ for $c \in \Sigma_{int}$, and $\varphi(a\,w_1\,b\,w_2) = a(\varphi(w_1), b(\varphi(w_2)))$ for $a \in \Sigma_{push}$ and $b \in \Sigma_{pop}$. The map is a bijection between $W_\Sigma$ and its image $\mathcal{T}_\Gamma = \varphi(W_\Sigma) \subseteq \mathsf{Trees}(\Gamma)$, the *BParse-shaped trees*. The reverse direction extends to a map $\mathrm{seq} : \mathsf{Trees}(\Gamma) \to \Sigma^*$ defined as the pre-order traversal of the tree's internal-node labels, with $\square$ leaves treated as $\varepsilon$. On $\mathcal{T}_\Gamma$ the map agrees with $\varphi^{-1}$; on the rest of $\mathsf{Trees}(\Gamma)$ it is many-to-one.

Given a binary acceptor $T : \Sigma^* \to \{0, 1\}$ that realizes some VPL $L \subseteq W_\Sigma$, the VPL$^*$ framework of Barbot et al. (2021) learns a VPG for $L$ by lifting $T$ to a tree-language target through $\varphi$ and running the tree-language extension of L$^*$ due to Sakakibara (1992) and Drewes and Högberg (2007) (TL$^*$) on top. TL$^*$ maintains an observation table over trees and contexts, issues membership queries on individual trees, conjectures a deterministic tree automaton (DTA), and refines the conjecture on counterexamples returned by an equivalence query until convergence. Membership queries against $T$ are answered by

$$\mathrm{MQ}^{\mathrm{gated}}(t) = \begin{cases} T(\mathrm{seq}(t)) & \text{if } t \in \mathcal{T}_\Gamma, \\ 0 & \text{otherwise,} \end{cases}$$

that is, trees outside the BParse image are gated to *false* before the target is consulted; the DTA produced by TL$^*$ is then post-processed into a VPG by the construction of Barbot et al. (2021). We refer to this as *gated*-VPL$^*$. Figure 1 shows the decision the gated membership

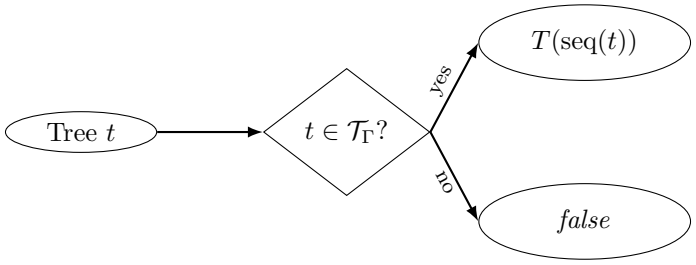



Figure 1: The gated membership policy of VPL$^*$.

oracle makes on each incoming tree. The equivalence query in their framework relies on A$^*$ search over the internal state of the RNN that realizes $T$, which ties the algorithm to recurrent neural networks in a white-box setting.

## 3. Leakage analysis on the gated learner

MQ$^{\text{gated}}$ returns *false* on every tree outside $\mathcal{T}_\Gamma$, irrespective of what the target would say on the corresponding sequence. When the target is in fact a VPL this is the right choice; when it is not, the gate hides information. Our second contribution is to instrument the gated learner with a side oracle that quantifies what the gate hides, at the cost of one extra target query per class-C tree and no change to the membership semantics seen by TL$^*$.

On class A the gated MQ already issues $T(\text{seq}(t))$ and any information the target carries on $\text{seq}(t)$ enters the observation table. On class B the sequence $\text{seq}(t)$ is in $W_\Sigma$ but the tree $t$ is a class-A tree, so the gate's *false* on class B suppresses a duplicate of an existing class-A row rather than hiding information. Class C is where the gate can lie. The sequence is outside $W_\Sigma$, no class-A tree yields it, and the gate's *false* can disagree with $T(\text{seq}(t))$ when the target accepts non-well-formed sequences. Concretely, a target that is not a VPL accepts some $w \in \Sigma^* \setminus W_\Sigma$ and the gate masks that fact. We call a class-C tree on which the target accepts the sequence a *leakage witness*: a sequence that the target accepts but no VPL with the same alphabet could.

The detector is a side observer added to the gated MQ. When the gated MQ is called on a tree $t$, the detector classifies $t$ and acts only on class C: it issues a single extra query $T(\text{seq}(t))$ and records the pair $(\text{seq}(t), T(\text{seq}(t)))$. The membership value returned to TL$^*$ is unchanged — still *false* on class C — so the observation table, the DTA, and the extracted VPG are exactly what the gated algorithm of Barbot et al. (2021) would have produced. Figure 2 shows the augmented decision: the upper branch is the original MQ$^{\text{gated}}$, and the lower branch routes the class-A complement through an additional check that filters out class B before issuing the target query that drives the leakage signal. A *probe* is an instance of the lower branch's target query. A *witness* is a probe the target accepted.

## 4. Conclusions

We extended the VPL$^*$ framework of Barbot et al. (2021) in order to learn a VPG while detecting whether part of the target's language is outside VPL. First, We added a leakage detector that issues one extra target query per class-C tree and records witnesses. Second,

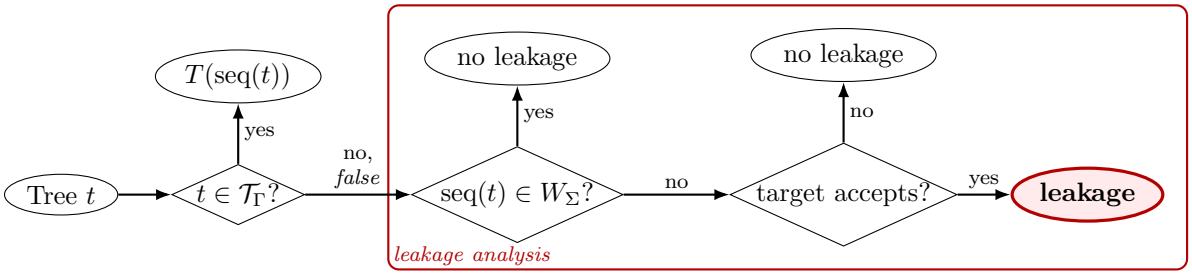



Figure 2: Gated VPL* with the leakage detector.

we defined a hybrid membership policy that lifts the gate on class C and keeps it on class B, folding the leakage signal into the observation table. The full version of this work is found in da Silva (2026). Future work includes performing a thorough experimental evaluation and reducing the size of the extracted grammar Alur et al. (2005).

## Acknowledgments

The research that led to the results presented in this publication received funding from Agencia Nacional de Investigación e Innovación (ANII) under grants POS_NAC_2023_1_178663 and FMV_1_2023_1_175864, and PEDECIBA, Uruguay.

## References

Rajeev Alur and P. Madhusudan. Visibly pushdown languages. In *Symposium on the Theory of Computing*, 2004.

Rajeev Alur, Viraj Kumar, P. Madhusudan, and Mahesh Viswanathan. Congruences for visibly pushdown languages. In *Automata, Languages and Programming, 32nd International Colloquium (ICALP)*, volume 3580 of *Lecture Notes in Computer Science*, pages 1102–1114, 2005.

Dana Angluin. Learning regular sets from queries and counterexamples. *Information and Computation*, 75(2):87–106, 1987.

Benoît Barbot, Benedikt Bollig, Alain Finkel, Serge Haddad, Igor Khmelnitsky, Martin Leucker, Daniel Neider, Rajarshi Roy, and Lina Ye. Extracting context-free grammars from recurrent neural networks using tree-automata learning and A* search. In *Proceedings of the 15th International Conference on Grammatical Inference*, volume 153 of *Proceedings of Machine Learning Research*, pages 113–129, 2021.

Juan da Silva. Active learning of context-free grammars from neural networks via tree-automata. Master's thesis, Universidad ORT Uruguay, 2026. URL https://hdl.handle.net/20.500.11968/8026.

Frank Drewes and Johanna Högberg. Query learning of regular tree languages: How to avoid dead states. *Theor. Comp. Sys.*, 40(2):163–185, February 2007. ISSN 1432-4350. doi: 10.1007/s00224-005-1233-3.

Xiaodong Jia and Gang Tan. V-Star: Learning visibly pushdown grammars from program inputs, 2024. URL https://arxiv.org/abs/2404.04201.

Franz Mayr and Sergio Yovine. Regular inference on artificial neural networks. In *International Cross-Domain Conference for Machine Learning and Knowledge Extraction (CD-MAKE)*, Lecture Notes in Computer Science, pages 350–369, 2018.

Yasubumi Sakakibara. Efficient learning of context-free grammars from positive structural examples. *Information and Computation*, 97(1):23–60, 1992.

Gail Weiss, Yoav Goldberg, and Eran Yahav. Extracting automata from recurrent neural networks using queries and counterexamples. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 5247–5256, 2018.