On the Relationship Between RNN Hidden-State Vectors and Semantic Structures

Edi Muškardin[1,2], **Martin Tappler**[3], **Ingo Pill**[1], **Bernhard K. Aichernig**[2], **Thomas Pock**[4]
[1]TU Graz - SAL DES Lab, Silicon Austria Labs
[2]Institute of Software Technology, Graz University of Technology
[3]Institute of Computer Engineering, Vienna University of Technology
[4]Institute of Computer Graphics and Vision, Graz University of Technology
edi.muskardin@silicon-austria.com, martin.tappler@tuwien.ac.at,
ingo.pill@silicon-austria.com, aichernig@ist.tugraz.at, pock@icg.tugraz.at

Abstract

We examine the assumption that hidden-state vectors of recurrent neural networks (RNNs) tend to form clusters of semantically similar vectors, which we dub the *clustering hypothesis*. While this hypothesis has been assumed in RNN analyses in recent years, its validity has not been studied thoroughly on modern RNN architectures. We first consider RNNs that were trained to recognize regular languages. This enables us to draw on perfect ground-truth automata in our evaluation, against which we can compare the RNN's accuracy and the distribution of the hidden-state vectors. Then, we consider context-free languages to examine if RNN states form clusters for more expressive languages. For our analysis, we fit (generalized) linear models to classify RNN states into automata states and we apply different unsupervised clustering techniques. With a new ambiguity score, derived from information entropy, we measure how well an abstraction function maps the hidden state vectors to abstract clusters. Our evaluation supports the validity of the clustering hypothesis for regular languages, especially if RNNs are well-trained, i.e., clustering techniques succeed in finding clusters of similar state vectors. However, the clustering accuracy decreases substantially for context-free languages. This suggests that clustering is not a reliable abstraction technique for RNNs used in tasks like natural language processing.

1 Introduction

In recent years we have seen significant advancements in the analysis and verification of artificial neural networks (ANNs) (Huang et al., 2017; Gopinath et al., 2017; Sun et al., 2018). The motivation has been to establish trust in the growing landscape of machine-learned systems. Most of the work considered feed-forward ANNs that implement functions without internal states. In this work we shift the focus to recurrent neural networks (RNNs) that model processes with internal states. Having access to an internal memory makes RNNs ideal for challenging tasks on high-dimensional sequential data, such as natural language processing (NLP) (Wang and Jiang, 2016; Tang et al., 2015) and time series forecasting (Elman, 1990).

Consequently, many researchers have turned to the internals of RNNs in hopes of understanding the RNN decision-making process (Omlin and Giles, 1996). An important, yet scarcely examined, hypothesis in this field postulates that the hidden state vectors visited by an RNN while processing data form clusters of semantically similar states. Consider sentiment analysis for movie reviews as an example (Dong et al., 2020). The sentences *"This movie is great."* and *"This movie is awesome."* are semantically similar, but syntactically different. An RNN processing these sentences is assumed to traverse states that belong to the same state clusters.

In recent years, the clustering hypothesis has been the basis of several RNN analysis methods. Dong et al., 2020 applied it to construct finite-state models over observed clusters. Through probabilistic model checking, they identified adversarial data. Weiss et al., 2018 developed an efficient technique for extracting deterministic finite automata from RNNs, which was extended to context-free languages (Yellin and Weiss, 2021). While these approaches relied on some form of the clustering hypothesis, they also encountered, and subsequently coped with inaccuracies of clustering functions. Instead of using clusters as abstract states directly, Dong et al., 2020 learned probabilistic automata over clusters. While successful applications of the clustering hypothesis *might imply* its soundness, the validity of the hypothesis still remains questioned. For example, already in early work Kolen, 1993 warned against methods that rely on state-space discretization (clustering) due to the inherent information loss. Also, Zeng et al., 1993 noticed problems with the clustering approach. Since validation and verification of RNNs

Findings of the Association for Computational Linguistics: ACL 2024, pages 5641–5658
August 11-16, 2024

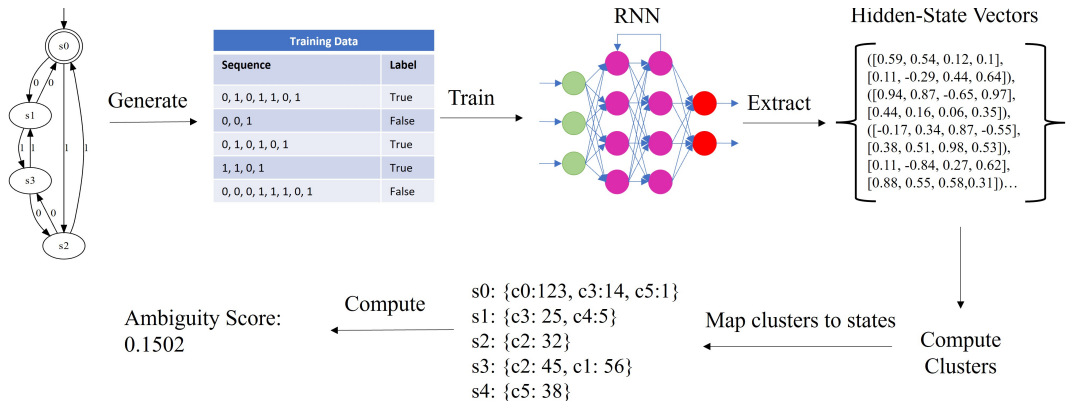

Figure 1: Overview of the process for assessing the quality of clustering functions.

should not be based on anecdotal evidence, we return to this open question and empirically examine the soundness of the clustering hypothesis.

More precisely, first, we train RNNs to recognize formal languages with known and minimal ground-truth automaton representations. Then, we process data with the RNN to investigate the following research questions: Are an RNN's hidden-state vectors linearly separable (RQ1)? Is the clustering hypothesis assumed for RNNs valid (RQ2)? Does the level of the Chomsky hierarchy affect clustering (RQ3)? We approach these questions by simulating validation data on trained RNNs to extract visited hidden state vectors. At the same time, we track the unique automaton state corresponding to each RNN state. Then, we partition the sampled RNN states using multiple clustering functions and we train linear multiclass classifiers to classify RNN states to automaton states. Linear classifiers use known ground-truth states, therefore they establish an upper bound for the precision of clustering with linear cluster boundaries. Hence, RQ1 can be seen as a prerequisite for some clustering techniques. To answer RQ2, we analyze the correspondence between derived clusters and automata states and assign an ambiguity score to each clustering configuration. Ideally, each cluster (or a set of clusters) should correspond to a unique state in the respective ground-truth automaton. An overview of this process can be seen in Fig. 1. For the first two research questions, we focus on regular languages. Regular languages provide a good basis for analysis, but many practical applications, such as natural language processing require more expressiveness. Therefore, for RQ3 we move one layer up in the Chomsky hierarchy and analyze the clustering hypothesis on a subset of context-free languages that can be modeled with deterministic pushdown automata.

Our contributions can be summarized as: (1) analyses of 1950 trained RNNs w.r.t. separability of their state space, (2) computation and analysis of clusters over the hidden-state vectors, (3) and a framework containing all presented functionalities developed with AALPY (Muškardin et al., 2022), PyTorch (Paszke et al., 2019), and scikit-learn (Pedregosa et al., 2011). To the best of our knowledge, this constitutes the first detailed empirical evaluation of the "clustering hypothesis".

Structure. In Sect. 2 we discuss related work followed by preliminaries in Sect. 3. Sect. 4 discusses our research questions and presents techniques and accuracy metrics that we use in Sect. 5 for an empirical evaluation of the clustering hypothesis. Sect. 6 concludes the paper and discusses future work.

2 Related Work.

Omlin and Giles, 1996 were among the first to mine rules from RNNs. They proposed deterministic finite automaton (DFA) extraction from a second-order RNN trained to recognize a regular language by applying a clustering algorithm over the RNN's hidden state space. They also visually analyzed the clustering of the RNN states, postulating that they are well separated. Similar works can be found in (Cleeremans et al., 1989; Giles et al., 1991; Watrous and Kuhn, 1991). Since then, RNN research made significant advancements and the clustering hypothesis was criticized (Kolen, 1993). Zeng et al., 1993 observed that the "clustering hypothesis" (Omlin and Giles, 1996) becomes unstable as longer sequences are used to extract hidden-state vectors. They propose changing the training as a solution. Schellhammer et al., 1998 trained an Elman RNN on a NLP task and constructed a state-transaction diagram representing a grammar of the data set with the help of clustering. However, they considered RNNs with only two hidden neu-

rons. Wang et al., 2018 empirically evaluated various conditions that might influence DFA extraction from second-order RNNs. Interestingly, they observed that rules extracted from RNNs were more precise than RNNs themselves in classifying longer sequences. Dong et al., 2020 combined principles from passive stochastic automata learning, with abstraction achieved by clustering the hidden-state space of an RNN. Clustering enabled adversarial data detection via probabilistic verification. Hou and Zhou, 2020 extracted automata from different types of gated RNNs with the help of clustering. They also reason about the influence of gating on clustering. However, their evaluation is restricted to two regular languages and a coarse abstraction for an NLP task, where they limit the number of clusters to just two. We improve upon their analysis by considering a significantly larger number of study subjects, RNN types, and clustering approaches, thus providing more nuanced insights into the clustering hypothesis. Michalenko et al., 2019 examined the relationship between hidden (Elman) RNN states and the states of DFAs. They learned decoding functions from hidden states to (sets of) DFA states and found that such (linear) functions exist, though some DFA states may need to be grouped. Their results suggest that "supervised clustering" may not enable perfect reconstruction of FSM rules, but rather non-deterministic approximations. We examine *if unsupervised clustering enables reconstruction of the finite-state semantics of the concept that is learned*, i.e., a view that is closer to practice.

So far we presented related work that, in some form or another, assumes or relies on the clustering hypothesis. The connections between formal languages and RNN were studied in other contexts, such as extraction of models from RNNs (Weiss et al., 2024; Mayr and Yovine, 2018; Muškardin et al., 2022; Eyraud et al., 2023), formal reasoning about the expressive power of RNNs and transformers (Merrill et al., 2020, 2022; Strobl et al., 2024), adapting an RNN's architecture and training so that the RNN implicitly learns the model of its behaviour (del Pozo Romero and Lago-Fernández, 2023; Aichernig et al., 2022), among others.

3 Preliminaries

A **deterministic finite automaton (DFA)** over alphabet Σ is a tuple $A = (Q, \Sigma, q_0, \delta, F)$, where Q is a finite set of states, $q_0 \in Q$ is the initial state, $\delta : Q \times \Sigma \to Q$ is the transition function, $F \subseteq Q$ are the final states. We extend the transition function as usual to arbitrary-length sequences $s \in \Sigma^*$, i.e., $\delta(q, \epsilon) = q$ and $\delta(q, e \cdot s') = \delta(\delta(q, e), s')$, where ϵ is the empty sequence, $e \in \Sigma$, $s' \in \Sigma^*$. A word $w \in \Sigma^*$ is accepted iff $\delta(q_0, w) \in F$. All accepted words form a regular language. **Moore machines** extend DFAs by producing an output from a discrete output alphabet O in every state defined by a function $\lambda : Q \to O$.

A **deterministic pushdown automaton (PDA)** is a 7-tuple $(Q, \Sigma, \Gamma, \delta, q_0, Z, F)$ where Q is a finite set of locations, Σ is a finite input alphabet, Γ is a finite stack alphabet, $\delta : Q \times \Sigma \times \Gamma^* \to Q \times \Gamma^*$ is the transition function, $q_0 \in Q$ is the initial location, $Z \in \Gamma$ is the initial stack symbol, and $F \subset Q$ is the set of accepting locations. The state of a PDA is a pair $(q, \gamma) \in Q \times \Gamma^*$, where q is a location and γ is a stack of symbols from Γ. As in DFAs, we extend δ to sequences w where $\delta(q_0, Z, w) = (q, \gamma)$ is the state reached after processing w.

Recurrent neural networks (RNNs). In this paper, we examine standard implementations of Elman RNNs (Elman, 1990) (with ReLU and $tanh$ activation functions), LSTMs (Hochreiter and Schmidhuber, 1997), and GRUs (Cho et al., 2014).

We view an RNN as a pair of functions (r, o), where $r : \mathbb{R}^h \times \mathbb{R}^m \to \mathbb{R}^h$ updates the hidden state based on the previous hidden state h_{t-1} and the current input i_t. We use a one-hot encoding for discrete inputs from an alphabet of size m. For simplicity, we also use this view for LSTMs by analyzing the state space spanned by the concatenation of the hidden state h_t and the cell state c_t of LSTMs. The output function $o : \mathbb{R}^h \to C$ maps the current hidden state to an output class in C. We train RNNs on sequences sampled from languages defined by DFAs, Moore machines, and PDAs. For DFAs and PDAs, we have $C = \{true, false\}$, where $true$ denotes acceptance of the sequence processed so far. For Moore machines, we have $C = O$, where O is an output alphabet and o maps to the last output produced by the corresponding Moore machine.

4 Method

This section presents the basis for the analysis of the "clustering hypothesis", including research questions and accuracy metrics.

4.1 Research Questions

Setting. In our experiments, we train RNNs on formal languages over an alphabet Σ. For each

regular language $L \subseteq \Sigma^*$, we sample a dataset $D \subset \Sigma^*$. We further split D into the training (D_t) and validation data (D_v) sets.

For a **regular language** L, let $A = \langle Q, \Sigma, q_0, \delta, F \rangle$ be a minimal DFA accepting exactly L and let $R = (r, o)$ be an RNN over the hidden state space $H = \mathbb{R}^h$ trained to recognize L. By processing every word in D_v simultaneously with the recurrent part r of R and the minimal DFA A, we determine the hidden states traversed by R as well as the corresponding automaton states reached by A. We store these data as pairs $(h, q) \in H \times Q$. For the remainder of this section let $\mathcal{HQ} \subset H \times Q$ be a concrete sample of such pairs and let $\mathcal{H}$ be the same sample without states Q.

For a **context-free language**, we follow an analogous procedure backed by a PDA $A = (Q, \Sigma, \Gamma, \delta, q_0, Z, F)$. In this case, we store triples $(h, q, \gamma) \in H \times Q \times \Gamma^*$, i.e., we track the configuration in addition to the automaton location.

RQ1: Are an RNN's hidden-state vectors linearly separable? As a first step, we investigate whether hidden-state vectors can be (piecewise linearly) separated into regions corresponding to automaton states with a focus on regular languages. We formulate this as a multiclass classification problem based on $\mathcal{HQ}$, the sampled hidden-state vectors labeled by automaton states. That is, we train a classification model on $\mathcal{HQ}$ to learn a function $cl : \mathbb{R}^h \to Q$ and we evaluate its accuracy at classifying hidden states correctly. We rely on (generalized) linear models linear discriminant analysis (LDA) and logistic regression (LR) for this task. This choice is motivated by the observation that non-linear functions were not required in a similar setting (Michalenko et al., 2019). While *piecewise linear separability* and low classification error of linear models *do not imply that the hidden states form well-defined clusters*, we gain insights into the structure of the hidden-state space. Linear separability can be seen as a prerequisite for clustering with linear boundaries, like k-means.

RQ2: Is the clustering hypothesis assumed for RNNs valid? This research question concerns the validity of the clustering hypothesis for regular languages. To examine it experimentally, we consider two aspects related to the clustering of RNN states.

(1) The definition of a cluster. Works based on the hypothesis apply different clustering techniques, hence we will analyze the effect of different popular clustering techniques.

(2) Usefulness of a cluster. A useful clustering should be a valid abstraction for model-based reasoning (Dong et al., 2020). Hence, clusters should have similar qualities as abstractions for software systems. Concrete hidden states *abstracted to the same cluster* should behave similarly. Moreover, an abstraction should be small enough to enable efficient subsequent analyses, thus we will examine the number of detected clusters. We will analyze these aspects via experiments with regular languages with known minimal ground-truth automata. Hence, **RQ2** can also be formulated as: **Do hidden state clusters correlate with states of an automaton accepting the same language?**

RQ3: Can the clustering hypothesis be extended to languages higher in the Chomsky hierarchy? While the clustering hypothesis was postulated for RNNs trained on regular languages, it has also been assumed for RNNs trained on natural language (Dong et al., 2020). As natural language cannot be modeled and analyzed in a similar manner as regular languages, we instead extend our analysis to the next level in the Chomsky hierarchy, i.e., to context-free languages. To analyze the clustering hypothesis in such a setting, we extend **RQ1** and **RQ2** to RNNs trained to recognize a subset of context-free languages that can be modeled by deterministic PDAs.

4.2 Accuracy Metrics

For **RNN accuracy**, denoted $Acc_R(AV)$ where AV is the accuracy-validation set, we use the standard approach based on the misclassification of sequences w.r.t. the ground-truth language L. That is, $Acc_R(AV)$ is a percentage of correctly classified sequences in the AV. In contrast, clustering accuracy is based on the ambiguity of interpreting clusters as states of a finite-state model, compared to the states of the ground-truth automation A.

Measuring the quality of clustering. For an intuition on clustering optimality, let us focus on regular languages first: *all data points in a cluster should correspond to a unique state in the ground truth* A. We can view a clustering as a function $c : \mathcal{H} \to K$ mapping sampled hidden states to cluster labels. This lets us compare a clustering c to the optimal mapping $hq : H \to Q$. To relate c and hq, we define c as optimal if there is an α s.t. $hq = \alpha \circ c$. Such a c allows extracting a DFA with states K from an RNN R that is equivalent to ground truth A if $Acc_R(AV) = 1$ and AV is large enough. To empirically evaluate a concrete c, we define the mismatch between c and hq based

on entropy. Recall that for c to be a useful abstraction over H, it should group semantically similar states. Therefore, we evaluate clustering impurity by computing empirical entropy values. Thus, we measure the degree of uncertainty (mismatch) in our clustering mappings via

$$amb(k) = -\sum_{q \in Q} \frac{n_{q,k}}{n_k} \log_{|Q|} \frac{n_{qk}}{n_k} \text{ where}$$

$$n_{qk} = |\{(h,q) \in \mathcal{HQ} | c(h) = k\}|, n_k = \sum_{q \in Q} n_{qk}$$

The logarithm with base $|Q|$ normalizes the ambiguity to the interval $[0, 1]$. For the ambiguity of a clustering function, we compute the average of all clusters given by $amb(c) = \frac{\sum_{k \in K} amb(k)}{|K|}$ and the weighted average $wamb(c) = \frac{\sum_{k \in K} amb(k) \cdot n_k}{|\mathcal{HQ}|}$.

We noted above that ideally $hq = \alpha \circ c$ for a renaming α. This is achieved iff $amb(c) = 0$. We say that clustering is perfect if it achieves a (weighted) ambiguity of zero. Alternatively to our notion of ambiguity, we could also use normalized mutual information (NMI), a commonly used estimate of clustering quality with an information-theoretic interpretation. However, clustering size affects NMI such that a perfect, but slightly non-minimal clustering may have a lower NMI than an ambiguous, but small clustering. Since the former (a perfect, non-minimal clustering) is more useful for RNN analyses than the latter (imperfect clustering), we examine ambiguity and clustering size separately.

For PDAs, we sample triples (h, q, γ), i.e., we need to match RNN states h to PDA states (q, γ). However, considering that clustering shall serve as an abstraction, we cannot take the complete stack γ into account. This would lead to a very large abstract state space, making linear separation and clustering of the RNN states superficially simple and the resulting abstraction not useful. As acceptance of words depends on whether a location q is accepting and the stack is empty, hidden states should form clusters corresponding to locations and stack size. In addition to stack size, the top element of the stack affects how PDAs process words. Therefore, we abstract away from the concrete stack configuration to either stack size or the top element. Altogether, we apply three abstractions in the experiments: the first stackful abstraction $abs_{tos}(q, \gamma) = (q, top(\gamma))$ maps the PDA states to the current location and top of the stack, the second stackful abstraction $abs_{sl}(q, \gamma) = (q, |\gamma|)$ maps PDA states to pairs of location and stack size, whereas the stackless abstraction $abs_l(q, \gamma) = q$ completely ignores the stack.

To enable a straightforward comparison between classification models and unsupervised clustering, we apply ambiguity also for classification models, by interpreting predicted classes as cluster labels. Note that an ambiguity of zero coincides with a misclassification rate of zero. Thus, zero ambiguity of linear models implies (piecewise linear) separability on the sample dataset $\mathcal{H}$.

5 Evaluation

Next, we present the experimental setup and empirical results related to our research questions. The code required to reproduce all experiments can be online [1]. The appendix includes numerical data from plots and additional experiments. All experiments were conducted on a laptop with an Intel® Core™i7-11800H CPU, NVIDIA 3050 Ti Moblie GPU32 GB RAM, and took ∼40h.

5.1 Experimental Setup

Case Study Subjects. We performed experiments with 59 regular languages encoded by DFAs. Five of which are evaluation subjects from the literature: three Tomita grammars (Tomita, 1982), an MQTT server model (Tappler et al., 2017), and a regular expression used by Michalenko et al., 2019. Additionally, we randomly generated 30 DFAs and 24 Moore machines with up to 12 states and 72 transitions using AALPY (Muškardin et al., 2022). For RQ3, we encoded context-free languages from (Yellin and Weiss, 2021) as PDAs. Those languages include variations of X^nY^n languages, Dyck languages (Hopcroft and Ullman, 1969), and variations of Dyck languages. Our experiments cover smaller automata often used in RNN research, as well as larger state spaces spanned by the PDAs, which use unbounded stacks.

Training. For each language, we trained an RNN to achieve perfect accuracy for three consecutive epochs on the validation data to potentially increase the likelihood of forming clusters in the hidden state space (Merrill and Tsilivis, 2022). We trained Elman RNNs with $tanh$ and $ReLU$, LSTMs, and GRUs, each of them with one layer of size t, one layer of size $1.5t$, two layers of size t, where t is the number of transitions of the ground-truth automaton. We have chosen these network sizes

[1] https://github.com/DES-Lab/Clustering_RNN_hidden_state_space

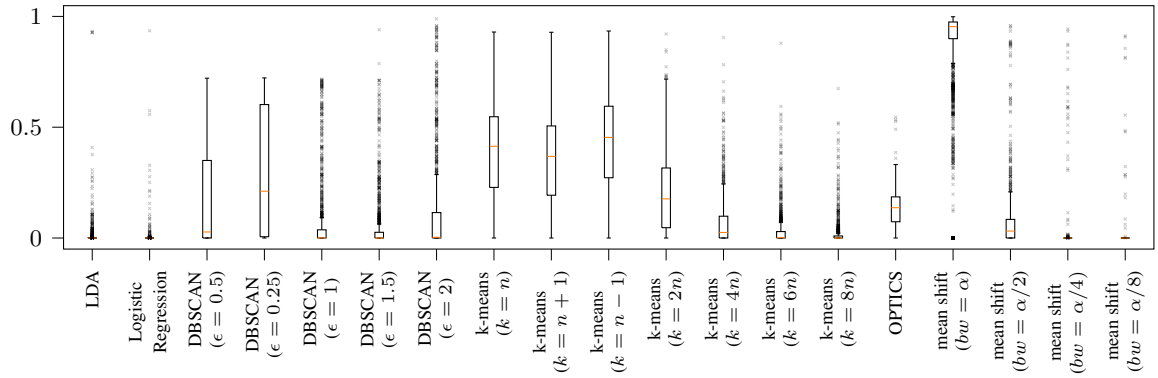

Figure 2: Boxplots of the weighted ambiguity resulting from different clustering techniques for all 1350 experiments whose RNNs achieved at least 80% accuracy.

since a one-layer Elman RNN with t hidden neurons is sufficient to encode automata with t transitions (Alquézar and Sanfeliu, 1995; Goudreau et al., 1994; Minsky, 1967). We decided to not consider very large networks, as it would complicate the clustering analysis due to dimensionality reduction becoming more important, hindering us from concentrating on clustering approaches. Moreover, the network sizes are sufficient for accurate training. We used the ADAM optimizer (Kingma and Ba, 2015) with a learning rate of 0.0005. The training data consisted of $50k$ randomly sampled words of lengths in the range $[1, 15]$, with labels derived from the ground-truth model. The validation data contained 2000 words, resulting in appr. $10k$ different hidden-state vectors for clustering. For all experiments, we trained two RNNs per configuration (network type, size).

Classification & Clustering. We used LDA and LR to learn classification models to determine if automaton states can be (piecewise linearly) separated in the hidden state space. We provide both approaches with $\mathcal{HS} \subset H \times Q$, sampled pairs of hidden states and automaton states.

We focus on popular, efficient clustering techniques available in scikit-learn (Pedregosa et al., 2011). We apply k-means (MacQueen, 1967), a partitional technique (Madhulatha, 2012) and three density-based techniques: mean shift (Comaniciu and Meer, 2002), DBSCAN (Ester et al., 1996), and OPTICS (Ankerst et al., 1999).

We examine various parameterizations of these algorithms and use Euclidean distance as a distance metric. We set the k of k-means based on the size $n = |Q|$ of the minimal ground-truth automata A with $k \in \{n-1, n, n+1, 2n, 4n, 6n, 8n\}$. With the first three values, we check whether a good clustering exists that is close to the minimal automaton representation. We also use greater values because an RNN may learn a non-minimal representation. When considering stackful mappings used in our experiments of RNN trained on CFLs, we increased the n to $n \times 3$ for stack-length abstractions and $n \times |\Sigma|$ for top-of-stack abstractions to account for alternative PDA state representations.

For DBSCAN we experiment with multiples of the neighborhood size $\epsilon = 0.5$, the default in the scikit-learn library. We leave the other parameter $minNeighbors$ at its default value of 5. For mean shift, we estimate the bandwidth bw with scikit-learn, which we denote α, to perform experiments with multiples of α. As OPTICS improves upon DBSCAN by mitigating its sensitivity on parameter values, we only apply its default parameterization. Since mean shift and OPTICS require more computation time than other techniques, we reduced the sample $\mathcal{H}$ to 25% of its size for these two.

5.2 Analysis of RNNs Trained on Regular Languages

In the first set of experiments, we consider all four RNN types, Elman RNNs with ReLU and tanh activation functions, LSTMs, and GRUs, as well as all clustering techniques. We evaluated all RNNs and only considered those whose accuracy ($Acc_R(AV)$ from Sect. 4.2) was $\geq 80\%$. We chose this accuracy cutoff as, in practice, RNNs rarely achieve perfect generalization due to the complexity of the underlying task or quality of training data. This resulted in a selection of 1350 from 1416 RNNs.

Figure 2 summarizes the results of these experiments. It shows boxplots of the weighted cluster ambiguity $wamb$ resulting from different clustering techniques, with outliers denoted by small crosses. Furthermore, the second row of Table 1 shows the number of perfect clusterings, i.e. $wamb = 0$,

Table 1: Number of perfect clusterings (zero ambiguity) achieved by selected clustering methods.

Clustering Function	LDA	LR	DBSCAN			k-means			OPTICS	mean shift		
Parameters			0.5	0.25	1.5	n	$6n$	$8n$		$\alpha/2$	$\alpha/4$	$\alpha/8$
Regular Languages (1350 experiments)	1003	1235	368	185	639	49	629	783	16	373	1296	1331
PDA stackless (600 experiments)	18	43	7	0	12	0	8	10	0	23	78	260
PDA stackfull (600 experiments)	16	30	7	0	9	4	15	16	0	14	31	81
PDA top-of-stack (600 experiments)	17	35	9	0	7	9	15	16	0	20	32	66

achieved by each method.

We observe that LDA and LR can achieve perfect classification in 74% and 91% of the cases, respectively. LDA has a mean $wamb$ of 0.0009 ± 0.05, while LR's mean $wamb$ was 0.0004 ± 0.039. Relating to **RQ1**, this high level of accuracy suggests that in most of the considered cases, *piecewise linear separability of the hidden state space is possible*. To further solidify our findings, we perform a quantile test on the $wamb$ with the alternative hypothesis being that 0.95-quantile is less than 0.05, i.e., if the $wamb$ is low in 95% of the cases. Due to the large number of outliers, we failed to reject the null hypothesis, i.e., we could not find statistically significant support for general linear separability.

The median weighted ambiguity of DBSCAN with $\epsilon \geq 1$ is almost equal to 0, which means that in at least half of the cases, we can identify semantically meaningful discrete states from clusters. Likewise, k-means achieves a low ambiguity when given at least four times as many clusters as "necessary", i.e., $k \geq 4n$. This means that RNNs seem to learn non-minimal representations of the concept that is being learned. OPTICS generally improves upon DBSCAN, but in this use case, it seems to perform slightly worse than DBSCAN. The accuracy of clusters computed by mean shift depends on the parameterization of the bandwidth bw. The bandwidth estimated by scikit-learn results in a number of clusters that is smaller than n, causing high ambiguity. But as we decrease bw, in turn increasing the number of found clusters, mean shift becomes the best clustering method, even outperforming supervised classification approaches like LDA and LR. Regarding **RQ2**, we can state that *clusters often correlate with automaton states*, with the caveat that proper parameterization is necessary. As for LDA and LR, we perform a hypothesis test on $wamb$ with the alternative hypothesis that the 0.95-quantile is less than 0.05. The only clustering method for which we rejected the null hypothesis with $p < 0.05$ is mean shift with $bw = \alpha/8$, i.e., we see significant support for the clustering hypothesis to hold in this case. Additional results related to RQ2, such as the impact of the network architecture, can be found in the appendix.

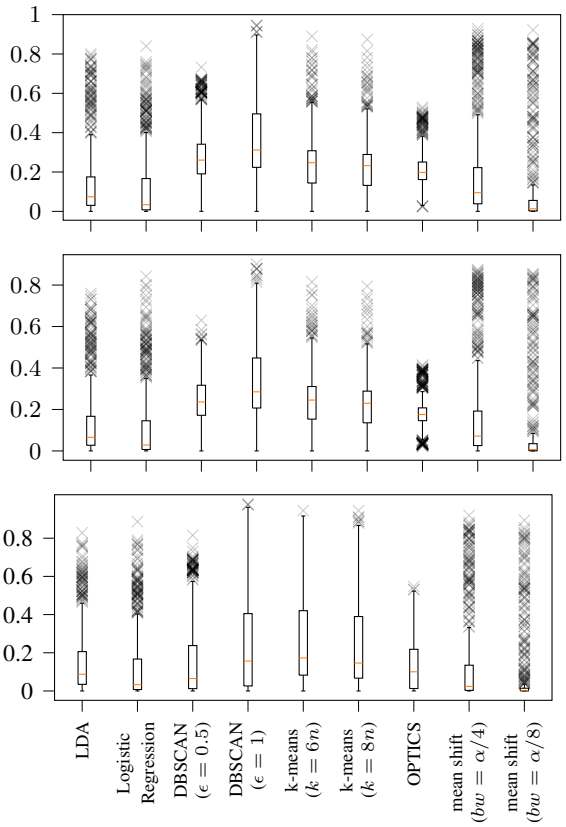

Figure 3: Weighted ambiguity for selected methods for RNNs trained on PDAs, without stack information (bottom) and with stack length (middle), and with top of the stack mapping (top).

5.3 Analysis of RNNs Trained on Context-Free Lanugages

In the second set of experiments, we trained RNN on a deterministic subset of context-free languages in the same manner as described in Sect. 5.1. We examined RNNs with $3t$ and $5t$ neurons, where t is the number of transitions in the ground-truth PDA. The training resulted in 600 networks all of which had an accuracy higher than 80%.

Figure 3 summarizes the weighted ambiguity results of selected clustering methods. We performed the analysis with three PDA state abstractions presented in Sect. 4.2. We considered two stackful

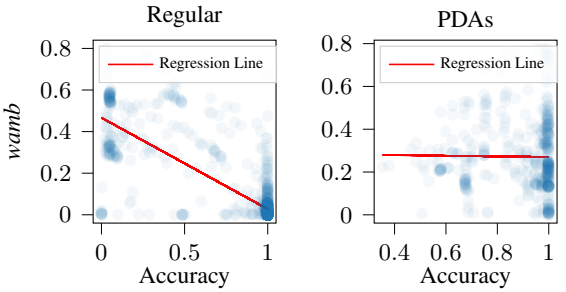

Figure 4: Relationship between RNN accuracy and weighted ambiguity for RNNs trained on regular languages (left) and PDAs (right).

mapping, one that considers the PDA location and the symbol found at the top of the stack, and the other that considers the PDA location and the current stack length. In addition, we examined a mapping that ignores the stack and only considers the current PDA location. The stackful abstractions strengthen the analysis, as they offer an alternative mapping that provides more information when compared to cluster-to-location mapping. However, we did not observe major differences between the weighted ambiguity of these abstractions, implying that additional stack information has little influence on RNN states compared to PDA locations themselves.

As seen in Tab. 1, compared to RNNs trained on regular languages, RNNs trained on context-free languages achieved perfect piecewise linear separability in a small fraction of all experiments. From this, we deduce that hidden states of RNNs trained to recognize context-free languages are *rarely perfectly linearly separable* and tend to form clusters that carry less semantic meaning than ones computed on RNNs trained on regular languages.

In general, we observe that the clustering functions of RNNs trained to recognize context-free languages are *more ambiguous than those of RNNs trained to recognize regular languages*, and as seen in Tab. 1 rarely achieve perfect clustering. More concretely, while k-means with $8n$ clusters resulted in mappings with low weighted ambiguity (0.018 ± 0.05), it failed to compute unambiguous mappings for all configurations of PDA experiments resulting in weighted ambiguities of 0.245 ± 0.238 for stackless abstraction, 0.234 ± 0.139 for length-of-stack abstraction, and 0.241 ± 0.151 for the top-of-stack abstraction.

To further compare the clustering hypothesis on RNNs trained on regular and context-free languages, we examined the relationship between weighted ambiguity and RNN accuracy. In Fig. 4, we plot the weighted ambiguity against the accuracy (with linear regression lines) of every RNN from our experiments. We observe a strong negative correlation between weighted ambiguity and accuracy for RNNs trained on regular languages, while there is no correlation between *wamb* and accuracy for RNNs trained on context-free languages. The Pearson correlation coefficient is -0.802, where a test for correlation yields a p-value smaller than 0.05, and -0.012, respectively. This finding further implies that the clustering hypothesis mostly holds for accurate RNNs trained on regular languages, while it cannot be assumed even for accurate RNNs trained on more complex languages.

Regarding **RQ3**, we showed that unlike for regular languages, clusters of hidden state vectors of RNNs trained on context-free languages are generally hardly linearly separable and do not strongly correlate with PDA locations. This can be considered a falsification of the clustering hypothesis on the higher-level languages, given that it mostly holds for regular languages, but does not hold for the next level in Chomsky hierarchy. However, in rare cases, clustering may be an effective abstraction, as it can be seen in Tab. 1.

5.4 Number of Clusters

As clustering is a tool for abstraction, we also need to look at the size of the abstraction, i.e., the number of clusters. Figure 5 shows the number of clusters derived by different techniques for experiments conducted on regular languages. We observed similar trends in cluster sizes between regular and context-free experiments.

K-means fixes the clustering size, but the number of clusters derived by other approaches depends on their parameters *and* the given data. The number of clusters in DBSCAN increases with decreasing ϵ parameter. While regardless of the ϵ the total number of clusters is higher than in k-means, it still facilitates an abstraction of RNNs. OPTICS finds a similar number of clusters as DBSCAN. The fact that mean shift with $bw = \alpha/8$ finds up to $1.5k$ clusters and that the third quartile in Fig. 5 is 500 weakens our findings regarding **RQ2**. The accurate clusterings found with mean shift hardly group states in many cases, thus making it trivial to achieve high clustering accuracy.

Discussion. Given our findings on the number of clusters, we should consider techniques and parameterizations that create reasonably-sized abstrac-

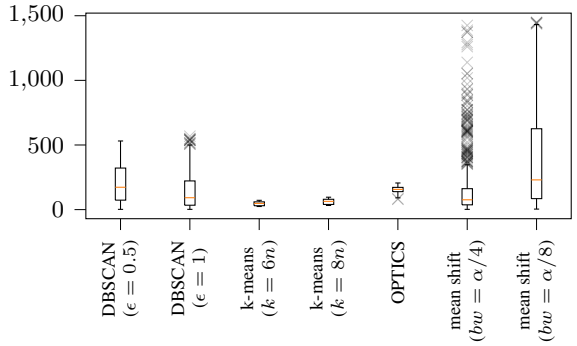

Figure 5: Number of clusters found in experiments of RNNs trained on regular languages.

tions. K-means with $k = 8n$ achieves perfect clustering in 58% of the cases on RNNs trained on regular languages (see Table 1). Additionally, we found that this k-means parameterization achieved a mean $wamb$ of 0.018 when considering regular languages, but in the most extreme case, $wamb$ was 0.67. As such reasonably-sized clustering functions do not achieve high accuracy in some cases, we cannot rely on the clustering hypothesis alone. This is exacerbated when we move to more complex languages, as we have in the experiments with deterministic context-free languages.

Generalizability. Since we consider formal languages in our experiments, our findings might not translate to NLP tasks. However, we believe that moving from context-free to free-form natural language, we may see an even larger ambiguity gap than between regular and context-free language. However, since a formal model of a natural language generally does not exist, we cannot examine the clustering hypothesis in that setting, and can only postulate based on this inductive step.

6 Conclusion and Future Work

We revisited and empirically analyzed the hypothesis that an RNN's hidden-state vectors tend to form clusters. Our analysis examines the clustering hypothesis for RNNs trained to recognize formal languages. This provided us with a ground truth enabling a comparison of the identified clusters with the original finite-state semantics that underlie the learned concept. Additionally, we examined the (linear) separability of hidden-state vectors into regions corresponding to automata states.

We observed that the hidden-state vectors of RNNs trained to recognize regular languages can often be (piecewise linearly) separated when considering regular languages. Considering unsupervised clustering, the regions identified by classifiers may contain multiple clusters and clusters may span across decision boundaries. Restricted to regular languages, we show that it is possible to compute clustering functions that correlate with automata states. For example, clusterings obtained by k-means with large k overall achieved high accuracy and perfect accuracy in 58% of the first set of experiments. This drastically changes when considering deterministic context-free languages, where the k-means with a large k achieved perfect clustering in only 2.6% of the cases. This is a strong indication that the clustering hypothesis alone should not be used as a basis for abstraction in the analysis of RNNs.

Future Work. With RNN verification being our ultimate goal, we will investigate ways to mitigate imprecisions and errors resulting from imperfect clustering. A stochastic interpretation of RNNs and stochastic automata learning over clusters seems promising (Dong et al., 2020). Further analysis of factors affecting the clustering is also important, like the effect of different optimizers or overtraining. Finally, we will investigate how to adapt the training and RNN architecture such that clusters are likely to form, enabling explainability-by-design.

Acknowledgments. This work has been supported by the "University SAL Labs" initiative of Silicon Austria Labs (SAL) and its Austrian partner universities for applied fundamental research for electronic based systems. In addition, this work has been partially supported by the WWTF project ICT22-023.

Limitations. To the best of our knowledge, our empirical analysis has two potential limitations: a) statistical significance of results, and b) dependence of our findings on the quality of clustering metric introduced in Sect. 4.2.

Regarding a), we applied exploratory data analysis to find trends in the data from a total of 1950 trained RNNs over 74 (59 regular languages and 15 context-free) formal languages. These networks include different architectures of various sizes, and for each trained network, we conduct experiments that contain multiple parameterizations of clustering algorithms. We examined selected findings through statistical tests, where we found statistically significant support for mean shift computing clusters with low ambiguity and a statistically significant correlation between ambiguity and accuracy of RNNs trained on regular languages. In cases, where trends were very clear, we did not

perform statistical tests like for the high ambiguity results involving context-free languages.

Regarding b), we believe that our entropy-based metric for the clustering quality reflects the quality of clustering in the scope of the presented research questions, that is, it encodes the correspondence of identified clusters to states. We further compared it to NMI, and the reason why NMI is not suited as a measure of clustering quality in our research context, is that might penalize perfect, but non-minimal, correspondence of clusters to states. Our experiments were performed on formal languages where the ground truth model against which we can compare RNNs performance is known and accessible. However, the existence of such a model cannot be generally assumed for NLP tasks.

Ethical Considerations. To the best of our knowledge, our work is fully in line with ACM Code of Ethics and Professional Conduct. Our experiments do not involve human subjects or any human-generated data that may intrude on an individual's privacy. We present fundamental research that contributes to the theoretical aspects of computing, for which we see no immediate issues related to ethics guidelines and/or potential misuse.

References

Bernhard K. Aichernig, Sandra König, Cristinel Mateis, Andrea Pferscher, Dominik Schmidt, and Martin Tappler. 2022. Constrained training of recurrent neural networks for automata learning. In *Software Engineering and Formal Methods - 20th International Conference, SEFM 2022, Berlin, Germany, September 26-30, 2022, Proceedings*, volume 13550 of *Lecture Notes in Computer Science*, pages 155–172. Springer.

René Alquézar and Alberto Sanfeliu. 1995. An algebraic framework to represent finite state machines in single-layer recurrent neural networks. *Neural Comput.*, 7(5):931–949.

Mihael Ankerst, Markus M. Breunig, Hans-Peter Kriegel, and Jörg Sander. 1999. OPTICS: ordering points to identify the clustering structure. In *SIGMOD 1999, Proceedings ACM SIGMOD International Conference on Management of Data, June 1-3, 1999, Philadelphia, Pennsylvania, USA*, pages 49–60. ACM Press.

KyungHyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches. *CoRR*, abs/1409.1259.

Axel Cleeremans, David Servan-Schreiber, and James L. McClelland. 1989. Finite state automata and simple recurrent networks. *Neural Comput.*, 1(3):372–381.

Dorin Comaniciu and Peter Meer. 2002. Mean shift: A robust approach toward feature space analysis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(5):603–619.

Juan Fdez. del Pozo Romero and Luis Fernando Lago-Fernández. 2023. Gradient-based learning of finite automata. In *Artificial Neural Networks and Machine Learning - ICANN 2023 - 32nd International Conference on Artificial Neural Networks, Heraklion, Crete, Greece, September 26-29, 2023, Proceedings, Part VIII*, volume 14261 of *Lecture Notes in Computer Science*, pages 294–305. Springer.

Guoliang Dong, Jingyi Wang, Jun Sun, Yang Zhang, Xinyu Wang, Ting Dai, Jin Song Dong, and Xingen Wang. 2020. Towards interpreting recurrent neural networks through probabilistic abstraction. In *35th IEEE/ACM International Conference on Automated Software Engineering, ASE 2020, Melbourne, Australia, September 21-25, 2020*, pages 499–510. IEEE.

Jeffrey L. Elman. 1990. Finding structure in time. *Cogn. Sci.*, 14(2):179–211.

Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96), Portland, Oregon, USA*, pages 226–231. AAAI Press.

Rémi Eyraud, Dakotah Lambert, Badr Tahri Joutei, Aidar Gaffarov, Mathias Cabanne, Jeffrey Heinz, and Chihiro Shibata. 2023. TAYSIR competition: Transformer+rnn: Algorithms to yield simple and interpretable representations. In *International Conference on Grammatical Inference, ICGI 2023, 10-13 July 2023, Rabat, Morocco*, volume 217 of *Proceedings of Machine Learning Research*, pages 275–290. PMLR.

C. Lee Giles, Clifford B. Miller, Dong Chen, Guo-Zheng Sun, Hsing-Hen Chen, and Yee-Chun Lee. 1991. Extracting and learning an unknown grammar with recurrent neural networks. In *Advances in Neural Information Processing Systems 4, [NIPS Conference, Denver, Colorado, USA, December 2-5, 1991]*, pages 317–324. Morgan Kaufmann.

Divya Gopinath, Guy Katz, Corina S. Pasareanu, and Clark W. Barrett. 2017. Deepsafe: A data-driven approach for checking adversarial robustness in neural networks. *CoRR*, abs/1710.00486.

Mark W. Goudreau, C. Lee Giles, Srimat T. Chakradhar, and Dong Chen. 1994. First-order versus second-order single-layer recurrent neural networks. *IEEE Trans. Neural Networks*, 5(3):511–513.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

John E. Hopcroft and Jeffrey D. Ullman. 1969. *Formal Languages and Their Relation to Automata*. Addison-Wesley Longman Publishing Co., Inc., USA.

Bo-Jian Hou and Zhi-Hua Zhou. 2020. Learning with interpretable structure from gated RNN. *IEEE Trans. Neural Networks Learn. Syst.*, 31(7):2267–2279.

Xiaowei Huang, Marta Kwiatkowska, Sen Wang, and Min Wu. 2017. Safety verification of deep neural networks. In *Computer Aided Verification - 29th International Conference, CAV 2017, Heidelberg, Germany, July 24-28, 2017, Proceedings, Part I*, volume 10426 of *Lecture Notes in Computer Science*, pages 3–29. Springer.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

John F. Kolen. 1993. Fool's gold: Extracting finite state machines from recurrent network dynamics. In *Advances in Neural Information Processing Systems 6, [7th NIPS Conference, Denver, Colorado, USA, 1993]*, pages 501–508. Morgan Kaufmann.

J. B. MacQueen. 1967. Some methods for classification and analysis of multivariate observations. In *Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297. University of California Press.

T. Soni Madhulatha. 2012. An overview on clustering methods. *CoRR*, abs/1205.1117.

Franz Mayr and Sergio Yovine. 2018. Regular inference on artificial neural networks. In *Machine Learning and Knowledge Extraction - Second IFIP TC 5, TC 8/WG 8.4, 8.9, TC 12/WG 12.9 International Cross-Domain Conference, CD-MAKE 2018, Hamburg, Germany, August 27-30, 2018, Proceedings*, volume 11015 of *Lecture Notes in Computer Science*, pages 350–369. Springer.

William Merrill, Ashish Sabharwal, and Noah A. Smith. 2022. Saturated transformers are constant-depth threshold circuits. *Trans. Assoc. Comput. Linguistics*, 10:843–856.

William Merrill and Nikolaos Tsilivis. 2022. Extracting finite automata from rnns using state merging. *CoRR*, abs/2201.12451.

William Merrill, Gail Weiss, Yoav Goldberg, Roy Schwartz, Noah A. Smith, and Eran Yahav. 2020. A formal hierarchy of RNN architectures. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 443–459. Association for Computational Linguistics.

Joshua J. Michalenko, Ameesh Shah, Abhinav Verma, Richard G. Baraniuk, Swarat Chaudhuri, and Ankit B. Patel. 2019. Representing formal languages: A comparison between finite automata and recurrent neural networks. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Marvin L. Minsky. 1967. *Computation: Finite and Infinite Machines*. Prentice-Hall, Inc., USA.

Edi Muškardin, Bernhard K. Aichernig, Ingo Pill, and Martin Tappler. 2022. Learning finite state models from recurrent neural networks. In *Integrated Formal Methods - 17th International Conference, IFM 2022, Lugano, Switzerland, June 7-10, 2022, Proceedings*, volume 13274 of *Lecture Notes in Computer Science*, pages 229–248. Springer.

Edi Muškardin, Bernhard Aichernig, Ingo Pill, Andrea Pferscher, and Martin Tappler. 2022. Aalpy: an active automata learning library. *Innovations in Systems and Software Engineering*, pages 1–10.

Christian W. Omlin and C. Lee Giles. 1996. Extraction of rules from discrete-time recurrent neural networks. *Neural Networks*, 9(1):41–52.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Ingo Schellhammer, Joachim Diederich, Michael Towsey, and Claudia Brugman. 1998. Knowledge extraction and recurrent neural networks: An analysis of an elman network trained on a natural language learning task. In *Proceedings of the Joint Conference on New Methods in Language Processing and Computational Natural Language Learning, NeMLaP/CoNLL 1998, Macquarie University, Sydney, NSW, Australia, January 11-17, 1998*, pages 73–78. ACL.

Lena Strobl, William Merrill, Gail Weiss, David Chiang, and Dana Angluin. 2024. What Formal Languages Can Transformers Express? A Survey. *Transactions of the Association for Computational Linguistics*, 12:543–561.

Youcheng Sun, Min Wu, Wenjie Ruan, Xiaowei Huang, Marta Kwiatkowska, and Daniel Kroening. 2018.

Concolic testing for deep neural networks. In *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering, ASE 2018, Montpellier, France, September 3-7, 2018*, pages 109–119. ACM.

Duyu Tang, Bing Qin, and Ting Liu. 2015. Document modeling with gated recurrent neural network for sentiment classification. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1422–1432. The Association for Computational Linguistics.

Martin Tappler, Bernhard K. Aichernig, and Roderick Bloem. 2017. Model-based testing IoT communication via active automata learning. In *2017 IEEE International Conference on Software Testing, Verification and Validation, ICST 2017, Tokyo, Japan, March 13-17, 2017*, pages 276–287. IEEE Computer Society.

M. Tomita. 1982. Dynamic construction of finite automata from examples using hill-climbing. In *Conference of the Cognitive Science Society*, pages 105–108.

Qinglong Wang, Kaixuan Zhang, Alexander G. Ororbia II, Xinyu Xing, Xue Liu, and C. Lee Giles. 2018. An empirical evaluation of rule extraction from recurrent neural networks. *Neural Comput.*, 30(9).

Shuohang Wang and Jing Jiang. 2016. Learning natural language inference with LSTM. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 1442–1451. The Association for Computational Linguistics.

Raymond L. Watrous and Gary M. Kuhn. 1991. Induction of finite-state automata using second-order recurrent networks. In *Advances in Neural Information Processing Systems 4, [NIPS Conference, Denver, Colorado, USA, December 2-5, 1991]*, pages 309–317. Morgan Kaufmann.

Gail Weiss, Yoav Goldberg, and Eran Yahav. 2018. Extracting automata from recurrent neural networks using queries and counterexamples. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 5244–5253. PMLR.

Gail Weiss, Yoav Goldberg, and Eran Yahav. 2024. Extracting automata from recurrent neural networks using queries and counterexamples (extended version). *Mach. Learn.*, 113(5):2877–2919.

Daniel M. Yellin and Gail Weiss. 2021. Synthesizing context-free grammars from recurrent neural networks. *Tools and Algorithms for the Construction and Analysis of Systems*, 12651:351 – 369.

Zheng Zeng, Rodney M. Goodman, and Padhraic Smyth. 1993. Learning finite state machines with self-clustering recurrent networks. *Neural Comput.*, 5(6):976–990.

A Appendix

A.1 Appendix A – RNNs.

A.1.1 Mathematical definitions of RNNs.

Below, we provide the mathematical definitions of Elman RNN's, LSTMs and GRUs following the PyTorch implementation (Paszke et al., 2019).

Elman RNNs:

$$h_t = g(x_t W_{ih}^T + b_{ih} + h_{t-1} W_{hh}^T + b_{hh})$$
$$\text{where } g \in \{\tanh, \text{ReLU}\}$$
$$y_t = f(h_t W_{ho}^T)$$

LSTMs:

$$i_t = \sigma(W_{ii} x_t + b_{ii} + W_{hi} h_{(t-1)} + b_{hi})$$
$$f_t = \sigma(W_{if} x_t + b_{if} + W_{hf} h_{(t-1)} + b_{hf})$$
$$g_t = \tanh(W_{ig} x_t + b_{ig} + r_t * (W_{hg} h_{(t-1)} + b_{hg}))$$
$$o_t = \sigma(W_{io} x_t + b_{io} + W_{ho} h_{(t-1)} + b_{ho})$$
$$c_t = f_t \odot c_{(t-1)} + i_t \odot g_t$$
$$\text{where } \odot \text{ is the Hadamard product and}$$
$$\sigma \text{ is the sigmoid function}$$
$$h_t = o_t \odot \tanh(c_t)$$

GRUs:

$$r_t = \sigma(W_{ir} x_t + b_{ir} + W_{hr} h_{(t-1)} + b_{hr})$$
$$z_t = \sigma(W_{iz} x_t + b_{iz} + W_{hz} h_{(t-1)} + b_{hz})$$
$$n_t = \tanh(W_{in} x_t + b_{in} + r_t \odot (W_{hn} h_{(t-1)} + b_{hn}))$$
$$h_t = (1 - z_t) \odot n_t + z_t \odot h_{(t-1)}$$

A.2 Appendix B – Additional Evaluation Details

In the following, we provide additional details on the experimental setup and additional experimental results in tables.

A.3 Impact of Network Architecture.

We examine the influence of the RNN architecture on classification and clustering. Figure 6 shows the average weighted ambiguity achieved by LR and selected clustering parameterizations for each RNN architecture separately. GRU networks appear to create state space structures that lend themselves best to clustering with any of the approaches. Interestingly, Elman ReLU RNNs lead to the highest ambiguity on average, even for LR. A potential explanation is that ReLU activations are unbounded, whereas $\tanh$ is bounded. Hence, $\tanh$ Elman RNNs may create clusters in the saturated area of $\tanh$.

Clustering of GRU Networks. In the following, we examine the clustering hypothesis for GRU networks trained on regular languages, for which we found particularly low ambiguity measurements. Figure 7 shows box plots of ambiguity values measured for different clustering techniques. On the left, we show measurements from all experiments involving GRUs and on the right, we show measurements from experiments, where LDA is able to perfectly separate hidden states corresponding to all automaton states.

We found that the hidden state space of the trained GRUs has a structure that is well-suited for linear separation of states. In 88% of the experiments, both LDA and LR can perfectly separate states.

This benefits k-means as well, which achieves lower ambiguity in all considered parameterizations when compared to other network types. In particular, k-means with $k = 8n$ performs very well, where the third quartile of the ambiguity values is equal to zero. From the other techniques, DBSCAN with $\epsilon = 1$ benefits from restricting the analysis to GRUs. OPTICS and especially mean shift are hardly affected. Focusing only on the experiments where LDA has an ambiguity of 0 (Fig. 7 (right)), we see that k-means performs especially well, with an ambiguity of at most 0.034 for $k = 8n$.

A.3.1 Further Details on Case Study Subjects.

We used regular language from the literature including three of the Tomita grammars (Tomita, 1982), where we have chosen Tomita 3, 5, and 7, as they define the most interesting languages. For example, Tomita 5 defines a parity language, which are difficult to learn for RNNs (Goudreau et al., 1994). The Tomita grammars have five, four, and four states, respectively, and an alphabet of size two. The MQTT server that we used in the evaluation has seven states, six inputs, and six outputs, and the regular expression from Michaelenko et al. (Michalenko et al., 2019) has seven states and an alphabet containing four symbols.

The randomly generated DFAs have five or ten states, and alphabet sizes of two, four, or six. For each combination of state and alphabet size, we

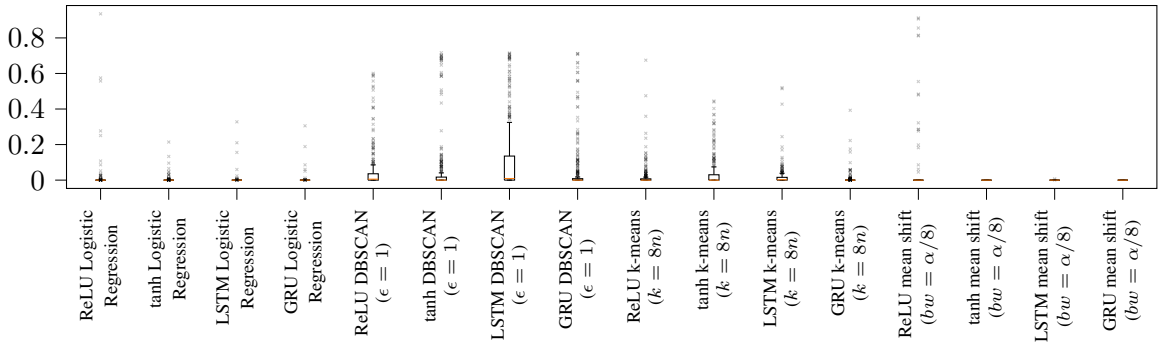

Figure 6: Weighted ambiguity for selected methods sorted by network types.

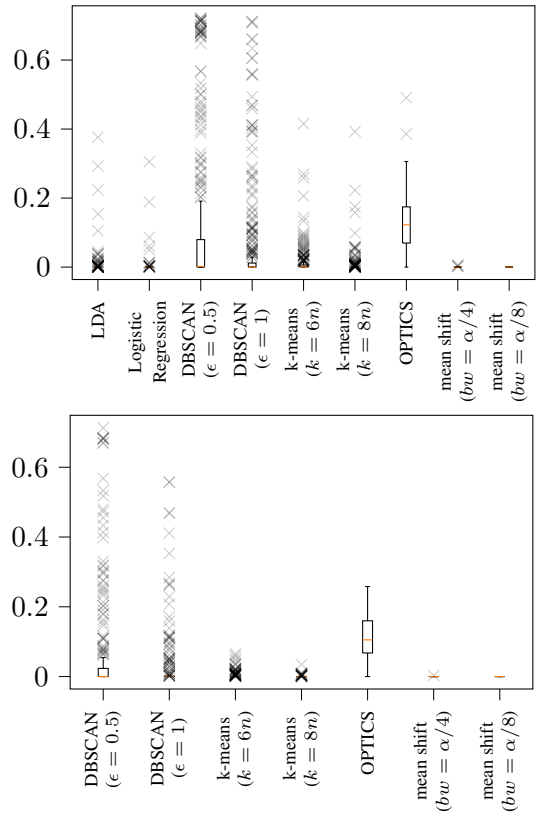

Figure 7: Boxplots of the weighted ambiguity resulting from different clustering techniques for all 346 experiments with GRUs (left) and for a subset of 278 GRU experiments, where LDA and LR acheived perfect separability.

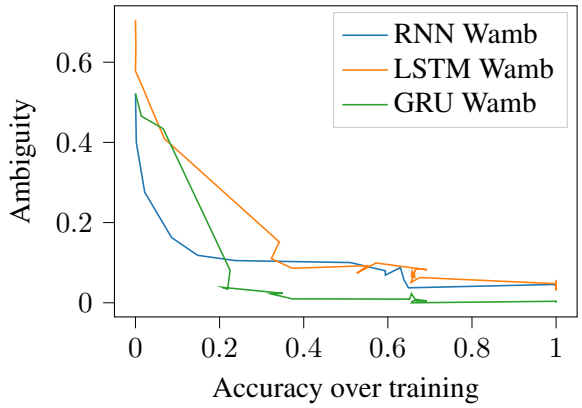

Figure 8: Relationship between accuracy and ambiguity over the training process.

generated five DFAs. We generated three Moore machines for each combination of eight and twelve states, two and four inputs, and three and five outputs. Note that Moore machines encode regular languages that contain all possible input-output sequences. From an RNN perspective, they enable us to analyze how multi-class classification tasks affect clustering.

A.3.2 Relationship between Accuracy and Ambiguity during Training

Figure 8 shows the relationship between RNN accuracy and ambiguity over the training process. We observe that the ambiguity decreases during the training process, as the accuracy increases. This shows that the clustering hypothesis is related to RNN accuracy, and even further validates the ambiguity metric introduced in Sect. 4.2. In only few cases, accuracy decreases slightly as ambiguity decreases and vice versa, which explains the kinks in the graphs. However, there is generally a strong negative correlation, i.e., the more accurate an RNN gets, the clustering gets less ambiguous: the Spearman correlation coefficients are -0.87, -0.96, and -0.9 for Elman RNNs, LSTMs, and GRUs in Fig. 8. Additionally, if the RNN retains the accuracy over long sequences (100-200 input symbols), the clustering ambiguity remains constant.

A.3.3 Extraction of cluster automata.

Figure 9 depicts a non-minimal automaton extracted from an RNN trained on the Tomita 5 gram-

mar. The extracted model is a non-minimal representation of the ground truth model, which is shown on the left-hand side of Fig. 1. We have applied the extraction method found in (Hou and Zhou, 2020) with multiple clustering functions that achieved perfect accuracy, where Fig. 9 is based on k-means clustering with $k = 4n$. Basically, we create one DFA state for each cluster $k \in K$. For the transitions, suppose that (h, q) and (h', q') are pairs of hidden states and automaton states collected consecutively while processing symbol e. In this case, we add a transition from $c(h)$ to $c(h')$ labeled by e, where c is the clustering function. If the RNN outputs $true$ for one hidden state in a cluster k, we add k to the accepting states. While the extracted model size varied, all extracted automata were non-minimal representations of the ground-truth model that was used to train the RNN. Our findings conform to those of (Hou and Zhou, 2020) and they indicate that RNNs learn non-minimal representations of regular languages.

A.3.4 Numerical Values of Results.

In Table 2 we show detailed results from our classification and clustering experiments performed on RNNs trained on regular languages, Table 4 shows detailed results of the analysis performed on RNN when mapping from clusters to states did not take stack into account, while Table 3 provides detailed results of experiments where clusters were mapped to state $\times$ stack length, whereas Table 5 presenters numerical values for top-of-stack PDA abstraction. The tables provide numerical data used to construct/be consistent with tables found in the paper.

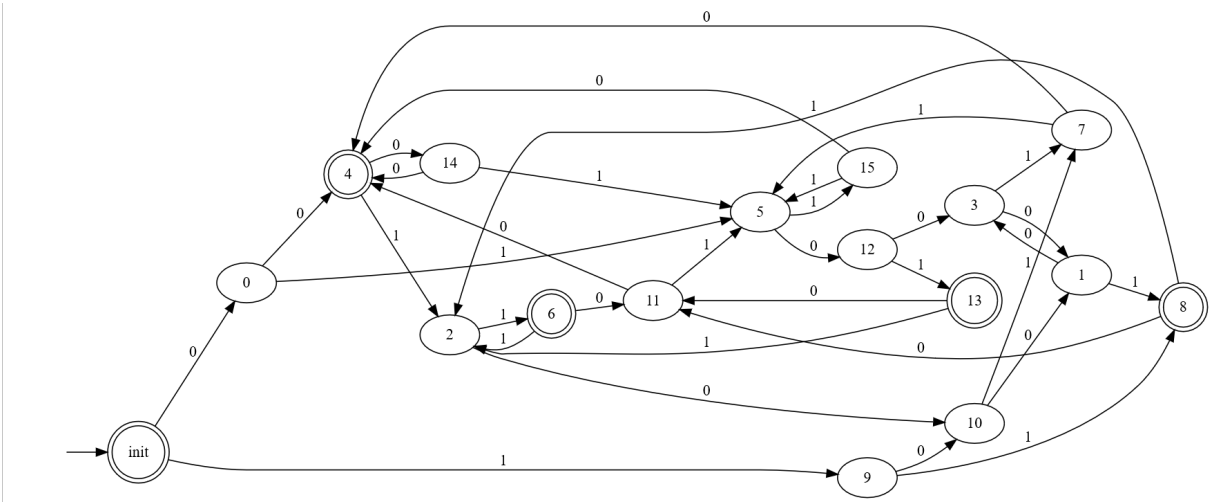

Figure 9: Extracted cluster automata from an RNN trained on the Tomita 5 grammar. K-means clustering with $k = 4n$ was used. The extracted model is a non-minimal representation of the ground truth model shown in Fig. 9.

Table 2: Clustering ambiguity results from 1350 RNNs that achieved 80% accuracy.

Clustering Function	**Ambiguity**		**Weighted Ambiguity**		**Number of Clusters**		**# Perfect Clustering**
	Mean ± Std Dev	Max	Mean ± Std Dev	Max	Mean ± Std Dev	Max	
LDA	0.01 ± 0.06	0.933	0.009 ± 0.05	0.933	8 ± 2.5	15	1003
Logistic Regression	0.003 ± 0.036	0.935	0.004 ± 0.039	0.935	8 ± 2.5	12	1235
DBSCAN ($\epsilon = 0.5$)	0.002 ± 0.002	0.025	0.179 ± 0.239	0.721	200 ± 140	531	368
DBSCAN ($\epsilon = 0.25$)	0.003 ± 0.002	0.02	0.29 ± 0.276	0.772	210 ± 118	506	185
DBSCAN ($\epsilon = 1$)	0.005 ± 0.025	0.672	0.066 ± 0.15	0.716	143 ± 135	568	591
DBSCAN ($\epsilon = 1.5$)	0.016 ± 0.063	0.939	0.05 ± 0.123	0.939	99 ± 115	596	639
DBSCAN ($\epsilon = 2$)	0057 ± 0.144	0.989	0.110 ± 0.208	0.98	69 ± 90	560	514
k-means ($k = n$)	0.359 ± 0.199	0.863	0.395 ± 0.207	0.929	8.5 ± 2.5	12	49
k-means ($k = n + 1$)	0.322 ± 0.131	0.811	0.356 ± 0.201	0.928	9.5 ± 2.5	13	68
k-means ($k = n - 1$)	0.401 ± 0.2	0.895	0.428 ± 0.21	0.934	7.5 ± 2.5	11	42
k-means ($k = 2n$)	0.175 ± 0.154	0.736	0.2 ± 0.172	0.92	16 ± 5	24	209
k-means ($k = 4n$)	0.06 ± 0.091	0.652	0.068 ± 0.105	0.904	33 ± 10	48	432
k-means ($k = 6n$)	0.0291 ± 0.065	0.585	0.032 ± 0.075	0.878	50 ± 15	72	629
k-means ($k = 8n$)	0.017 ± 0.051	0.54	0.0181 ± 0.057	0.674	67 ± 21	96	783
OPTICS	0.006 ± 0.003	0.06	0.134 ± 0.071	0.544	156 ± 20	206	16
mean shift ($bw = \alpha$)	0.847 ± 0.244	0.998	0.878 ± 0.201	0.998	1.35 ± 1.2	20	84
mean shift ($bw = \alpha/2$)	0.038 ± 0.063	0.935	0.073 ± 0.136	0.958	37 ± 51	829	373
mean shift ($bw = \alpha/4$)	0.001 ± 0.014	0.246	0.011 ± 0.086	0.943	136 ± 178	1425	1296
mean shift ($bw = \alpha/8$)	0.0004 ± 0.005	0.09	0.005 ± 0.06	0.912	391 ± 397	1448	1331

Table 3: Clustering ambiguity results from 600 RNNs trained to recognize PDAs with stackful mapping.

Clustering Function	**Ambiguity**		**Weighted Ambiguity**		**Number of Clusters**		**# Perfect Clustering**
	Mean ± Std Dev	Max	Mean ± Std Dev	Max	Mean ± Std Dev	Max	
LDA	0.169 ± 0.127	0.685	0.141 ± 0.176	0.759	5 ± 3	11	16
Logistic Regression	0.105 ± 0.119	0.847	0.121 ± 0.181	0.842	5 ± 3	11	30
DBSCAN ($\epsilon = 0.5$)	0.036 ± 0.042	0.192	0.25 ± 0.136	0.63	264 ± 129	679	7
DBSCAN ($\epsilon = 0.25$)	0.017 ± 0.031	0.178	0.264 ± 0.14	0.624	302 ± 109	684	0
DBSCAN ($\epsilon = 1$)	0.078 ± 0.093	0.879	0.342 ± 0.185	0.901	196 ± 140	656	4
DBSCAN ($\epsilon = 1.5$)	0.184 ± 0.226	0.965	0.456 ± 0.237	0.965	95 ± 119	641	9
DBSCAN ($\epsilon = 2$)	0.3700 ± 0.281	0.965	0.578 ± 0.237	0.965	51 ± 49	709	5
k-means ($k = n$)	0.341 ± 0.147	0.742	0.404 ± 0.184	0.866	16 ± 8	33	4
k-means ($k = n + 1$)	0.324 ± 0.144	0.735	0.383 ± 0.183	0.853	19 ± 8	36	5
k-means ($k = n - 1$)	0.371 ± 0.148	0.749	0.441 ± 0.179	0.875	13 ± 8	30	2
k-means ($k = 2n$)	0.291 ± 0.139	0.794	0.336 ± 0.177	0.158	33 ± 17	66	2
k-means ($k = 4n$)	0.256 ± 0.128	0.684	0.279 ± 0.16	0.822	66 ± 33	132	11
k-means ($k = 6n$)	0.242 ± 0.124	0.665	0.252 ± 0.145	0.815	98 ± 50	198	15
k-means ($k = 8n$)	0.233 ± 0.121	0.627	0.234 ± 0.139	0.794	131 ± 66	264	16
OPTICS	0.048 ± 0.035	0.179	0.183 ± 0.087	0.414	224 ± 55	376	0
mean shift ($bw = \alpha$)	0.611 ± 0.252	0.914	0.707 ± 0.201	0.914	3 ± 4	46	0
mean shift ($bw = \alpha/2$)	0.242 ± 0.147	0.839	0.395 ± 0.246	0.876	44 ± 81	783	14
mean shift ($bw = \alpha/4$)	0.066 ± 0.065	0.576	0.179 ± 0.238	0.873	513 ± 512	2589	31
mean shift ($bw = \alpha/8$)	0.02 ± 0.04	0.381	0.090 ± 0.2	0.854	1123 ± 738	2680	81

Table 4: Clustering ambiguity results from 600 RNNs trained to recognize PDAs with location $\times$ length of stack mapping.

Clustering Function	**Ambiguity**		**Weighted Ambiguity**		**Number of Clusters**		**# Perfect Clustering**
	Mean ± Std Dev	Max	Mean ± Std Dev	Max	Mean ± Std Dev	Max	
LDA	0.177 ± 0.163	0.849	0.160 ± 0.18	0.808	5 ± 3	11	18
Logistic Regression	0.131 ± 0.179	0.891	0.133 ± 0.196	0.886	5 ± 3	11	43
DBSCAN ($\epsilon = 0.5$)	0.014 ± 0.023	0.186	0.171 ± 0.217	0.815	264 ± 130	679	7
DBSCAN ($\epsilon = 0.25$)	0.007 ± 0.018	0.176	0.185 ± 0.229	0.697	302 ± 109	705	0
DBSCAN ($\epsilon = 1$)	0.046 ± 0.098	0.957	0.246 ± 0.235	0.978	169 ± 140	650	7
DBSCAN ($\epsilon = 1.5$)	0.137 ± 0.226	0.988	0.376 ± 0.286	0.988	95 ± 118	637	12
DBSCAN ($\epsilon = 2$)	0.238 ± 0.288	0.989	0.5 ± 0.29	0.99	50 ± 93	710	11
k-means ($k = n$)	0.396 ± 0.238	0.979	0.476 ± 0.25	0.979	5 ± 3	11	0
k-means ($k = n + 1$)	0.364 ± 0.225	0.966	0.447 ± 0.244	0.961	6 ± 3	12	0
k-means ($k = n - 1$)	0.469 ± 0.265	0.986	0.534 ± 0.262	0.986	4 ± 3	10	0
k-means ($k = 2n$)	0.302 ± 0.221	0.96	0.386 ± 0.249	0.953	11 ± 6	22	4
k-means ($k = 4n$)	0.235 ± 0.215	0.945	0.309 ± 0.25	0.947	22 ± 11	44	5
k-means ($k = 6n$)	0.201 ± 0.205	0.937	0.271 ± 0.245	0.944	33 ± 17	66	8
k-means ($k = 8n$)	0.181 ± 0.198	0.925	0.245 ± 0.238	0.944	44 ± 22	88	10
OPTICS	0.019 ± 0.03	0.189	0.134 ± 0.147	0.546	224 ± 55	383	0
mean shift ($bw = \alpha$)	0.567 ± 0.284	0.952	0.649 ± 0.251	0.946	3 ± 4	36	2
mean shift ($bw = \alpha/2$)	0.817 ± 0.181	0.931	0.345 ± 0.266	0.931	42 ± 75	782	23
mean shift ($bw = \alpha/4$)	0.027 ± 0.052	0.525	0.141 ± 0.234	0.919	509 ± 505	2619	78
mean shift ($bw = \alpha/8$)	0.008 ± 0.034	0.665	0.079 ± 0.192	0.893	1120 ± 735	2699	260

Table 5: Clustering ambiguity results from 600 RNNs trained to recognize PDAs with top-of-stack mapping.

Clustering Function	**Ambiguity**		**Weighted Ambiguity**		**Number of Clusters**		**# Perfect Clustering**
	Mean $\pm$ Std Dev	Max	Mean $\pm$ Std Dev	Max	Mean $\pm$ Std Dev	Max	
LDA	0.195 ± 0.181	0.777	0.154 ± 0.196	0.800	5 ± 3	11	17
Logistic Regression	0.129 ± 0.152	0.774	0.128 ± 0.188	0.839	5 ± 3	11	35
DBSCAN ($\epsilon = 0.5$)	0.054 ± 0.054	0.249	0.284 ± 0.164	0.732	264 ± 129	667	9
DBSCAN ($\epsilon = 0.25$)	0.028 ± 0.041	0.230	0.293 ± 0.169	0.674	302 ± 109	706	0
DBSCAN ($\epsilon = 1$)	0.104 ± 0.110	0.911	0.359 ± 0.208	0.694	196 ± 140	649	7
DBSCAN ($\epsilon = 1.5$)	0.207 ± 0.226	0.957	0.491 ± 0.250	0.947	95 ± 118	649	14
DBSCAN ($\epsilon = 2$)	0.327 ± 0.283	0.957	0.609 ± 0.249	0.957	50 ± 93	176	8
k-means ($k = n$)	0.355 ± 0.175	0.881	0.403 ± 0.208	0.918	33 ± 26	110	9
k-means ($k = n + 1$)	0.340 ± 0.168	0.857	0.382 ± 0.203	0.913	39 ± 28	120	11
k-means ($k = n - 1$)	0.377 ± 0.185	0.889	0.430 ± 0.217	0.924	27 ± 25	100	6
k-means ($k = 2n$)	0.313 ± 0.161	0.834	0.341 ± 0.193	0.908	66 ± 53	220	11
k-means ($k = 4n$)	0.282 ± 0.150	0.768	0.286 ± 0.173	0.895	133 ± 107	440	15
k-means ($k = 6n$)	0.267 ± 0.146	0.754	0.259 ± 0.160	0.889	200 ± 161	660	15
k-means ($k = 8n$)	0.257 ± 0.141	0.726	0.241 ± 0.151	0.873	267 ± 215	880	16
OPTICS	0.067 ± 0.04	0.198	0.215 ± 0.106	0.529	224 ± 55	373	0
mean shift ($bw = \alpha$)	0.637 ± 0.266	0.938	0.732 ± 0.215	0.938	2 ± 4	40	1
mean shift ($bw = \alpha/2$)	0.279 ± 0.177	0.932	0.435 ± 0.259	0.932	42 ± 76	813	20
mean shift ($bw = \alpha/4$)	0.078 ± 0.070	0.643	0.199 ± 0.246	0.928	512 ± 504	2577	32
mean shift ($bw = \alpha/8$)	0.024 ± 0.038	0.34	0.102 ± 0.211	0.922	1130 ± 734	2671	66