Formal Languages, Deep Learning, Topology and Algebraic Word Problems

Joshua Ackerman
Department of Computer Science
Dartmouth College
Hanover, NH
joshua.m.ackerman.gr@dartmouth.edu

George Cybenko
Thayer School of Engineering
Dartmouth College
Hanover, NH
gvc@dartmouth.edu

***Abstract*—This paper describes relationships between various state-of-the-art neural network architectures and formal languages as, for example, structured by the Chomsky Language Hierarchy. Of particular interest are the abilities of a neural architecture to represent, recognize and generate words from a specific language by learning from positive and negative samples of words in the language. Of specific interest are some relationships between languages, networks and topology that we outline analytically and explore through several illustrative experiments. By specifically comparing analytic results relating formal languages to topology via algebraic word problems with empirical results based on neural networks and persistent homology calculations, we see evidence that certain observed topological properties match analytically predicted properties. Such results are encouraging for understanding the role that modern machine learning can play in formal language processing problems.**

I. Introduction

The underlying motivation for this work is to explore novel machine learning techniques that can be used in the DARPA Safedocs program [1], and language security more generally [30], in several ways, including:

1) To generate new sample documents from a language;
2) To "learn" a grammar from sample documents from the grammar's language;
3) To classify documents as being within a given grammar's language or not;
4) To classify documents as having one or more of several malformations;
5) To have an empirical comparisons of machine learning vs formal methods, and;
6) To supplement formal methods when there are uncertainties in a language's definition but there is a corpus of known documents deemed to be compliant with the language's definition or grammar.

Figure I depicts the logical relationships between several technical areas that we describe in this paper and are using to understand and exploit towards the above goals. As indicated, here we focus on the relationships between formal languages and topology through algebraic word problems and experiments that get at understanding of how deep networks translate formal languages into sets that can be studied empirically using persistent homology [38].

It is important to note that, like many important computing problems, some formulations of "learning" a formal language are provably extremely difficult. As a result, one has to be careful about how the "learning" problem is formulated.

For example, given a finite training set of positive and negative samples of words from a regular language, the problem of computing a nondeterministic finite automaton (NDFA) that correctly accepts and rejects those training set words is a simple and efficient construction (linear in the total number of symbols in the sample words in fact). However, finding the *minimal state* deterministic finite automaton (DFA) that correctly accepts and rejects those words is known to be NP-Hard [28].

Generally speaking, there are two aspects of machine learning problems:

1) Does the underlying model architecture (for example, a decision tree, deep neural network, etc.) have the expressive power to represent the expected class of solutions?
2) Can a specific, near optimal instance of that architecture be efficiently learned from the training data?

We refer the reader to a recent survey paper [2] for additional details about formal language and machine learning relationships while [27], [15] contain recent results on the effects that deep learning have on transforming topological properties of datasets.

II. Background

Our work ties together a variety of topics such as formal languages, deep neural networks, algebraic word problems and algebraic topology. These topics span computer science and mathematics, typically at the graduate study level so that a comprehensive introduction to and understanding of all the material we are drawing on is beyond the scope of this paper.

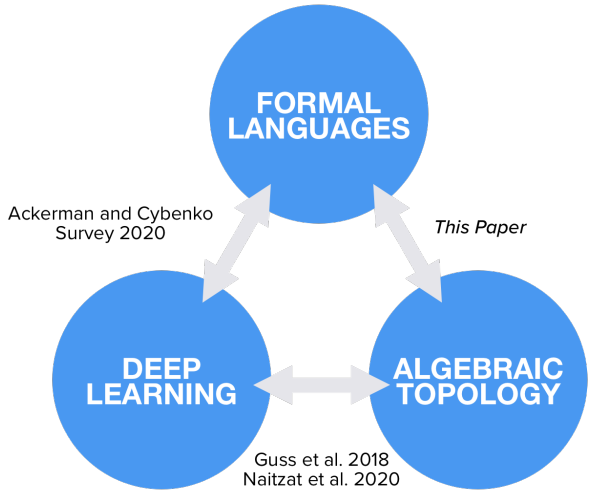

Figure 1. This figure depicts various relationships between the three topics we address in this paper. We are studying the relationships between formal languages and algebraic topology in two ways – analytically and experimentally. Analytically, we use basic ideas from algebraic word problems and homotopy groups to relate languages to topology. Experimentally, we have learned a variety of languages and empirically evaluated their homology properties using a variety of neural networks and persistent homology techniques. This two pronged approach is used to compare analytic and experimental results.

Grammar	Definition
Tomita 1	1^*
Tomita 2	$(10)^*$
Tomita 3	All strings without $1^{2\ell+1}0^{2k+1}$ as a substring
Tomita 4	All strings without 000 as a substring

Table I
DEFINITIONS OF THE FIRST FOUR TOMITA GRAMMARS – A COMMON SET OF BENCHMARK LANGUAGES USED THROUGHOUT LITERATURE AT THE INTERSECTION OF NEURAL NETWORKS AND FORMAL LANGUAGES. ONE PARTICULARLY NICE PROPERTY OF THESE GRAMMARS IS THAT THEIR MINIMAL AUTOMATA ARE KNOWN.

As a result, we introduce concepts as needed and at a high level to give readers at least an intuitive feel for the ingredients.

First off, machine learning using deep neural networks is a very active research topic these days so we assume most readers are at least intuitively familiar with basic concepts of deep learning such as fully connected, convolutional neural networks, recurrent networks and the concept of a hidden layer. There are many excellent references for this subject, including the seminal work [23] which is already slightly dated due to the intense research work being performed in the area.

In this section we introduce relevant definitions from formal language theory and algebraic topology that we use subsequently.

Definition 1 (Tomita Grammars [35]). *See table I.*

Definition 2 (Dyck Languages). *Given a bipartite set of characters $(P, \bar{P})$, the Dyck language, $\mathcal{D}_P$, is defined by the set,*

$$\mathcal{D}_P = \{x \in (P \cup \bar{P})^* \mid x \text{ is a well balanced set of parenthesis } \}.$$

In contexts where we are only concerned with the number of parenthesis, we will write $\mathcal{D}_n$ as short hand for $\mathcal{D}_{[2n]}$. More details and background can be found in [16] for example.

On an interesting side note, it is clear that nesting of parentheses can be handled by a stack wherein we push and pop as symbols are read. If the stack is empty when the word is read, it is a proper Dyck word. When implementing a stack using stateful, real valued recurrent neural nodes, the depth of the stack, that is the depth of the nesting, challenges the precision of the real arithmetic being used. In very early work (1971!) on the depth of nesting of FOR loops in early Fortran programs, Knuth found the following distribution of loop depths [21]:

Depth	1	2	3	4	5	>5
Number	4,211	1,853	1,194	437	118	120
Per cent	53.5	23.0	15.0	5.5	1.5	1.5

Table II
THIS EMPIRICAL STUDY OF NESTING DEPTH OF FOR LOOPS IN A LARGE SAMPLE OF FORTRAN PROGRAMS FROM 1971 SHOWS THAT FOR THAT CLASS OF PROGRAMS THE NESTING DEPTH IS RELATIVELY SMALL IN MOST PROGRAMS. THIS SUGGESTS THAT REAL WORLD REALIZATIONS OF DYCK TYPE LANGUAGES MAY HAVE SMALL DEPTH AS WELL ALTHOUGH WE ARE NOT AWARE OF ANY SUCH STUDIES FOR MODERN PROGRAMMING LANGUAGES OR DOCUMENT TYPES.

Definition 3 (Homology Groups [15], [12]). *If $\mathcal{X}$ is a topological space, then $H_n(\mathcal{X}) := \mathbb{Z}^{\beta_n}$ is called the nth homology group of $\mathcal{X}$ if the power β_n is the number of 'holes' of dimension n in $\mathcal{X}$. The number $\beta_n(\mathcal{X})$ is called the nth Betti number of $\mathcal{X}$. The lowest Betti number β_0 represents the number of connected components. We write the homology of $\mathcal{X}$, $H(\mathcal{X}) = (H_n(\mathcal{X}))_{n\geq 0}$. Moreover, $H_n(\mathcal{X})$, the first homology group is the Abelianization of the first homotopy group, $\pi_1(\mathcal{X})$, of the space $\mathcal{X}$.*

With these definitions in hand we can define the topological complexity of $\mathcal{X}$ as,

$$\omega(\mathcal{X}) = \sum_i \beta_i(\mathcal{X}).$$

See sources such as [4], [12], [15] for more details.

III. DEEP NETWORKS AND FORMAL LANGUAGES

Understanding how well different neural network architectures can decide membership in classes of formal

languages is a fundamental problem spanning machine learning and language processing. In principle, it is known that even the simplest variants of Recurrent Neural Networks are capable of emulating a Turing Machine in real-time, and consequently are Turing Complete [32]. However, this result and similar ones, rely on unrealistic assumptions such as unbounded computation time and infinite precision representation of real-valued states.

For these reasons, our understanding of the relationship between different realistic network architectures and the Chomsky hierarchy [19] is not complete, and as a consequence there is growing interest in understanding how different networks operate under these more realistic constraints. In particular, to be operationally useful, computation time should be linear in the input size and bounded precision arithmetic should be a constraint. Adopting the terminology of [37], such networks are called input-bounded neural networks with finite-precision states (IBFP-NNs).

Research in this general area dates back to the early 1990s, with works such as [33] [36] [17] [29] empirically studying the ability of different networks to learn variations of context-free counter languages [10]. Notably, it was around this time when researchers started exploring the idea of augmenting networks with memory constructs such as in [7], which introduced the Recurrent Neural Network Pushdown Automaton (NNPDA) – an RNN augmented with an external stack. Very recently, the body of research on such Memory Augmented Neural Networks has grown considerably, with the introduction of fully differentiable memory models such as Neural Stacks [14], Neural Queues [14], and even Neural Turing Machines [13]. Many of these papers present empirical results, so naturally, it is not guaranteed their findings hold in general.

However, in the last few years, a number of papers have appeared that make very direct comparisons of modern network architectures with respect to formal language processing. Papers such as [24] and [37] explore the theoretical power of IBFP variants of many such state-of-the-art networks, while papers like [34] explore how networks can be augmented to better perform formal language focused tasks.

Initially, we are primarily focused on two key tasks from the list above: (i) generating novel samples from unknown grammars given a small sample set of positive examples; (ii) empirically deciding whether new samples are in a language thus learned. Although the aforementioned papers primarily discuss results in the spirit of task (ii), it is worth noting that generative modeling (i), is a comparably harder task than that of discriminative modeling (ii), so many of these results are still applicable (albeit less precisely) with respect to quantifying the limits of networks towards generative tasks.

Readers seeking a more details and references about these works should consult [2].

IV. Algebraic Topology and Deep Networks

Understanding how deep neural networks transform datasets' topological properties is a very active area of research today. Details of such results can be found in [27], [15] as well as in more recent works that explore the subject. A detailed description of the relevant concepts and results is beyond the scope of this present paper but we can offer some intuition about what the basic results are.

Considering two-class classification problems only for the sake of exposition, it is well known that linearly separable classes are easy to classify using a single logistic or sigmoidal nonlinear regression function, that is a simple one node neural network. Now if the dataset of interest has a complex distribution with many isolated interspersed islands, holes and shapes, a neural network that seeks to solve the classification problem must transform the complex geometry ultimately to a linearly separable point cloud of positive and negative samples at the final layer. That is, each layer of a deep network somehow simplifies the topological properties. The complexity of a topology can be measured by its homology which is what we measure empirically using persistent homology techniques [38] in our experiments.

As previously mentioned, the inspiration for our ongoing work comes in part from efforts such as described in [27], [15] which perform comprehensive experiments to support and quantify empirically such insights and intuitions.

V. Languages, Group Word Problems and Topology

In this section, we relate formal languages to certain types of group word problems. A group word problem starts with a finitely generated algebraic free group and a finite set of relations between products of group elements, collectively called a group presentation.

The meaning of the relations is that they define equivalences between various products of elements and their inverses. The group word problem is the problem of deciding which general products are equal to the group's identity through reductions based on the relations of the presentation.

More specifically, let G be the finitely generated free group with generators, $g_1, g_2, ..., g_n$, meaning that G consists of products of the form

$$g_{i_1}^{k_1} g_{i_2}^{k_2} ... g_{i_m}^{k_m}$$

where $1 \leq i_j \leq n$ and $k_j \in \mathbb{Z} = \{..., -1, 0, 1, 2, ...\}$. In formal language terms, let $\Sigma =$

$\{1, g_1, g_2, ..., g_n, g_1^{-1}, g_2^{-1}, ..., g_n^{-1}\}$ be an alphabet and Σ^* be its Kleene closure, with 1 being the multiplicative group identity and the property that $g_i^0 = 1$.

A *group presentation* goes further by augmenting the generator set, Σ, with a finite set of words from Σ^*, $R = \{ r_i \mid r_i \in \Sigma^* \}$ that by definition are equal to the identity element. Then a finite group presentation for a group, H, is written as

$$H = \langle \Sigma \mid R \rangle.$$

In this formalism, the *Algebraic Word Problem* is the problem of recognizing the words in Σ^* that are equal to the identity, 1, given the relations in R. If that subset of words in Σ^* is denoted by L, we are basically asking what kind of language is L ? More precisely,

$$L = \{w \in \Sigma^* \mid w \in H,\ w = 1\}$$

and we ask what kind of language, in the sense of Formal Language Theory, is L?

An early result in algebraic word problem area is that H is a finite group if and only L is a regular language [3]. Extensions to context-free languages and more general results can be found in [26], [25], [8], [5]. Of particular interest are groups whose word problems are context-free languages. Such groups are characterized in the following result:

> **Theorem:** Let G be a finitely generated group. Then G has a context-free word problem if and only if G is virtually free [26], [25], [8].

Here, a group, G, is by definition virtually free if G has a free subgroup, G', of finite index, namely that the set of cosets of G given by $\{gG' \mid g \in G \}$ is finite.

This result does not state nor imply that every context-sensitive language corresponds to the word problem for some algebraic group which is in fact not the case [31], [22]. On the other hand, it has been shown that other large classes of groups, such as "automatic groups," also have context-sensitive word problems [18], [11]. Furthermore, so-called "braid groups" which describe the family of configurations of braids are automatic groups and have multiple relationships to topological spaces and their properties [9].

Recent work in this general area is summarized in [20].

In this framework, Dyck-n words are a structured subset of the language of the free group with n generators, noting that each bracket type $(_j,)_j$ pair can appear as either pairs of the form $(_j ...)_j$ or $)_j ...(_j$ corresponding to group products of the form $g_j g_j^{-1} = g_j^{-1} g_j = 1$.

We now review a relationship between certain groups and topology through the concept of fundamental groups and homotopy. Briefly, the homotopy group of a topological space is formed by equivalence classes of continuous embeddings of a circle into that space. Two embeddings are in the same equivalence class if one can be continuously deformed into the other within the space. Such equivalence classes can be shown to have a group structure through concatenation of embeddings (group products) and reversing the orientation of an embedding (inverse of a group element). More details can be found in basic references on algebraic topology and homotopy such as [4].

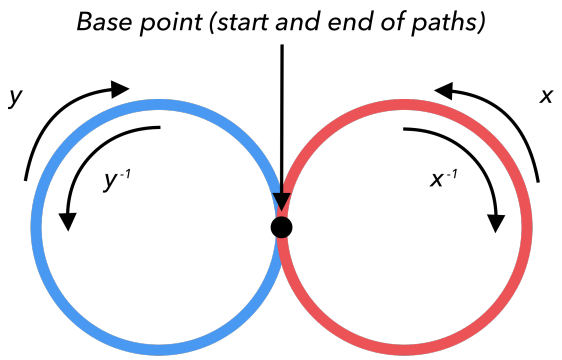

Figure 2. This depicts the relationship between a Dyck-2 language and the fundamental group of a two-petal space, that is a figure eight. Clearly, words such as xx^{-1} and $xxyx^{-1}xy^{-1}x^{-1}x^{-1}$ can be continuously deformed to the identity (the base point) within the figure 8 but words such as xy and $xyx^{-1}y^{-1}$ cannot be. The former two words are in the extended notion of the Dyck-2 language but the later two are not.

Of particular interest here is the free group with two generators because it is precisely the homotopy group of a figure eight topological structure. This affords us a geometric interpretation of a class of context-free languages in terms of topological properties. In particular, the equivalence class of embeddings of a circle into a figure eight space that contains the identity element is precisely the Dyck-2 context-sensitive language described above, namely the language of the word problem for the fundamental group of the figure eight. This relationship is simply illustrated in Figure V.

We note also that the first homology group is the Abelianization of the first homotopy group, namely in this case the free group, which is $\mathbb{Z}^k$ for a Dyck-k language so the first Betti number is precisely k.

These properties and relationships extend to Dyck-k languages for $k > 2$ as well, where a figure eight can be thought of as a two petaled flower and for other k, we have a k petaled flower.

To summarize, in this section we have outlined a relationship between context-sensitive languages and topological spaces via algebraic groups, wherein a group

word problem defines a language and the group is the fundamental group of a topological space. As described in Section IV, deep neural networks empirically transform topological properties of an input data space into other simpler topological structures. Moreover, Section III summarizes some recent analytic relationships between deep networks' abilities to recognize certain formal language classes.

VI. Experiments

A. Setup

We perform a preliminary series of preliminary experiments, with the hopes of finding insights at the intersection of the perspectives offered by [24] [15], and [27]. The specific area of interest is whether or not there is a dataset-specific characterization of syntatic complexity like [15] or [27], that captures the hardness of learning increasingly complex syntactic relationships.

While this question is broad, here, we focus on addressing related foundational questions such as:

1) To what degree does syntactic complexity influence topological complexity empirically?
2) Is the aformentioned topological perspective of deep learning useful in the analysis of syntactic data?

In this exploratory paper we address these in a very direct way. Specifically, for item (1), we look at random sets of syntactically disparate languages and see if they induce richer dataset homology, that would in principle, make them harder to learn by neural architectures. For item (2), we track the topological complexity throughout networks and analyze simple trends to see if topology is even a sensible method to reason about random syntactic datasets.

To motivate why this question is intriguing beyond the prior works of [15] and [27], we remark that syntatic data is particularly 'stringent' in contrast to other types of natural data. Given some well-formed string $\mathbf{x}$ of a language, it is usual for malformed strings to have closer L_1 distance to $\mathbf{x}$ than other well-formed strings in the language. As such, one might imagine that such collections of positive and negative strings have empirically smaller-scale topological structure, that might negate the usefulness of using tools like persistent homology to predict dataset hardness.

The experimentation we propose to study this question first involves generating a dataset, $\mathcal{X} = \mathcal{X}^+ \cup \mathcal{X}^-$, comprised of correct and incorrect strings, $\mathcal{X}^+$ and $\mathcal{X}^-$ from a formal language. Once complete, we estimate lower Betti numbers of the data before and while it flows through a well-trained neural network.

The languages we study include $\mathcal{D}_1$ and $\mathcal{D}_2$, in addition to the first four Tomita grammars [35]. These specific Tomita grammars are of particular interest due to the variation represented in the state complexity of their known minimal automata. In particular, the first and second Tomita grammars have minimal automata of size two and three respectively, while the third and fourth Tomita grammars have minimal automata of size five and four respectively.

For each of these languages, we use a random subset of 5,000 samples from the respective datasets released by bhattamishra et al. in [6]. We then generate an equally large dataset of poorly formed samples i.e., $\mathcal{X}^-$ by randomly perturbing one to seven characters of the well-formed samples.

The feedforward network we train is a fully connected network with six hidden layers with the first having the same size as the input dimension. The subsequent layers have size 200, 100, 100, 50, 50 respectively. Finally the last layer of size two, has a softmax activation.

To estimate topological metrics on the data and network outputs outputs, we follow [15] and use a locally linear embedding (LLE) with L_1-norm neighbors to perform dimension reduction of the data down to two dimensions before running persistent homology. We then apply min-max normalization to the embedding so that across the network features occupy the same space. As [15] notes, an LLE with sufficiently low reconstruction error will approximately preserve the homology of the original set. We remark that this step is necessary due to the run-time of persistent homology in high dimensions. To estimate the Betti numbers, we filter through the (b_i, d_i) pairs returned from persistence homology, keeping only pairs where $|b_i - d_i| > \epsilon$ for $\epsilon := 0.025, 0.015, 0.05$. Note, as a consequence of this convention, it is possible for the topological complexity of a set to appear to be zero. This simply means that ϵ was set too high, and the estimate can be discarded.

B. Results

Let h_i generically refer to the ith hidden layer of a neural network. Table III shows the estimated homology of the six different language datasets at different resolutions as they flow through a well-trained feedforward neural network. In detail, the average accuracy of all networks across all languages was approximately 92.0%, and the average reconstruction error for the dataset embedding was 4.7×10^{-4}.

Overall, we observed sporadic differences in the topological complexity of the data across hidden layers. Contrary to our expectations, we frequently saw initial increases in topological complexity followed by a final decrease. We suspect this could be explained by using a wide layer earlier in the network. At $\epsilon = 0.015$, we can roughly separate the Tomita languages, and the Dyck languages into two different classes of hardness using

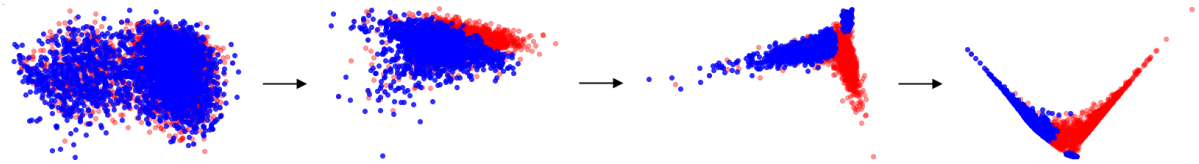

Figure 3. This figure illustrates the simplification of topological structure of a Dyck-2 language at four intermediate layers of our trained neural network. The Dyck-2 words are of length 1-50 so this is a projection of high dimensional euclidean spaces to two dimensions for visualization purposes, using LLE for the projections. Note that at the final layer displayed, the dataset is nearly linearly separable as a result of the nonlinear network layers' transformations.

	Scale	$\omega(\mathcal{X})$	$\omega(h_1(\mathcal{X}))$	$\omega(h_2(\mathcal{X}))$	$\omega(h_3(\mathcal{X}))$	$\omega(h_4(\mathcal{X}))$	$\omega(h_5(\mathcal{X}))$	$\omega(h_6(\mathcal{X}))$
$\mathcal{D}_1$ Dataset	0.015	374	539	775	427	284	68	32
	0.025	100	47	115	75	55	19	12
	0.050	17	3	9	10	8	4	3
$\mathcal{D}_2$ Dataset	0.015	558	537	495	298	79	47	32
	0.025	104	51	56	54	25	15	12
	0.050	18	9	11	9	5	4	5
Tomita 1	0.015	48	110	182	287	238	81	70
	0.025	18	36	105	176	134	34	47
	0.050	2	2	34	35	19		10
Tomita 2	0.015	60	143	168	242	12	99	38
	0.025	24	47	90	144	5	37	27
	0.050	9	9	37	30	3	10	14
Tomita 3	0.015	44	109	828	825	654	583	197
	0.025	7	11	224	145	55	56	24
	0.050	0	3	8	7	4	5	3
Tomita 4	0.015	331	591	595	825	233	60	45
	0.025	102	118	56	131	22	10	12
	0.050	25	12	6	9	0	2	3

Table III
TOPOLOGICAL COMPLEXITY OF EACH DATASET AS IT FLOWS THROUGH A WELL-TRAINED FEEDFORWARD NETWORK. AS IN [15] AND [27], WE ULTIMATELY OBSERVE A FINAL REDUCTION IN THE DATASET TOPOLOGY, HOWEVER, FOR OUR DATA THE REDUCTION IS NOT MONOTONIC ACROSS THE HIDDEN LAYERS.

their topological complexity. However, at $\epsilon = 0.025$ or $\epsilon = 0.05$ this trend does not hold. Further, we also do not see any correspondence between the state-complexity of the Tomita grammars and the topological complexity of their dataset. Unfortunately, due to the approximate nature of persistent homology it is difficult to decisively resolve these observed differences. However, in future work, we plan to explore the impact of using more principled and complex methods to analyze these persistent persistent diagrams.

Ultimately, persistent homology only provides an estimate of the data homology, and minor differences in ϵ can impact the results. Further, our experiments only test random datasets of samples of the aforementioned languages. It certainly could be that all-encompassing datasets exhibit different behavior, and more principled topological measurements would yield different results. Possibly, in light of the large combinatorial size of some of the baseline languages, our selected datasets are uncharacteristic of larger behavior. Alternatively, given the pragmatic conditions intrinsic to any empirical exploration of syntactic complexity, these observations could be explained by the fact that any finite collection of strings is regular, and other factors beyond the ground-truth grammatical complexity of the dataset impact the results.

VII. SUMMARY AND FUTURE WORK

This paper has related a variety of technologies relating formal languages, algebraic groups, algebraic topology and deep neural networks, working towards the six challenges identified in the Introduction. Primarily, our experiments suggest that either a finer-grained measure of complexity beyond homology is necessary to understand syntax in the context of machine learning, or a more principled approach to applying topological tools to such syntactic datasets is necessary. We leave the exploration of these ideas to future work.

While there is considerable ongoing work today devoted to empirical experiments showing how well both natural and formal languages can be learned from training data, there are few analytic results about how specific language types can be effectively learned.

In particular, we believe there are interesting questions to investigate related to:

1) Refining or expanding the Chomsky language hierarchy to better accommodate the representability and learnability of formal languages;
2) Efficiently inferring a language type from empirical data;
3) Efficiently inferring a language grammar from empirical data;
4) Using appropriate generative models to simultaneously improve classification and generation;
5) Using modern machine learning constructs to better understand the specific language constructs that are most susceptible and likely to be exploited by attackers;
6) Universal embeddings of language samples into suitably high-dimensional Euclidean space in such a manner that features of the language can be estimated and inferred from the geometric and topological structure of the embedded data.

As a result, we view the findings and relationships presented in this paper as a starting point as opposed to the conclusion of a rich research area.

Acknowledgements

The authors sincerely thank Sergey Bratus, Michael Robinson, Mikael Vejdemo-Johansson, Walt Woods, Shankar Natarajan and other members of the Safedocs community for discussions and feedback on this work.

References

[1] Darpa Safedocs Program, https://www.darpa.mil/program/safe-documents.

[2] Ackerman, J., and Cybenko, G. A survey of neural networks and formal languages. *arXiv preprint arXiv:2006.01338* (2020).

[3] Anisimov, A. V. Group languages. *Cybernetics 7*, 4 (1971), 594–601.

[4] Arkowitz, M. *Introduction to homotopy theory*. Springer Science & Business Media, 2011.

[5] Berstel, J., and Boasson, L. Context-free languages. In *Formal Models and Semantics*. Elsevier, 1990, pp. 59–102.

[6] Bhattamishra, S., Ahuja, K., and Goyal, N. 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 2020, Online, November 16-20, 2020* (2020), B. Webber, T. Cohn, Y. He, and Y. Liu, Eds., Association for Computational Linguistics, pp. 7096–7116.

[7] Das, S., Giles, C. L., and Sun, G.-Z. Learning context-free grammars: Capabilities and limitations of a recurrent neural network with an external stack memory. In *Proceedings of The Fourteenth Annual Conference of Cognitive Science Society* (1992).

[8] Dunwoody, M. J. The accessibility of finitely presented groups. *Inventiones mathematicae 81*, 3 (1985), 449–457.

[9] Epstein, D. B. *Word processing in groups*. CRC Press, 1992.

[10] Fischer, P. C., Meyer, A. R., and Rosenberg, A. L. Counter machines and counter languages. *Mathematical systems theory 2*, 3 (1968), 265–283.

[11] Gilman, R. H. Formal languages and infinite groups. *DIMACS Ser. Discrete Math. Theoret. Comput. Sci 25* (1996), 27–51.

[12] Goldman, W. Topology and geometry. by glen e. bredon. *The American Mathematical Monthly 105*, 2 (1998), 192–194.

[13] Graves, A., Wayne, G., and Danihelka, I. Neural turing machines. *ArXiv abs/1410.5401* (2014).

[14] Grefenstette, E., Hermann, K. M., Suleyman, M., and Blunsom, P. Learning to transduce with unbounded memory. *CoRR abs/1506.02516* (2015).

[15] Guss, W. H., and Salakhutdinov, R. On characterizing the capacity of neural networks using algebraic topology. *arXiv preprint arXiv:1802.04443* (2018).

[16] Haussler, D. Insertion languages. *Information Sciences 31*, 1 (1983), 77–89.

[17] Hölldobler, S., Kalinke, Y., and Lehmann, H. Designing a counter: Another case study of dynamics and activation landscapes in recurrent networks. In *KI* (1997).

[18] Holt, D. F., Rees, S., and Shapiro, M. Groups that do and do not have growing context-sensitive word problem. *International Journal of Algebra and Computation 18*, 07 (2008), 1179–1191.

[19] Jäger, G., and Rogers, J. Formal language theory: refining the chomsky hierarchy. *Philosophical Transactions of the Royal Society B: Biological Sciences 367*, 1598 (2012), 1956–1970.

[20] Jones, S. A., and Thomas, R. M. Word problems of groups: Formal languages, characterizations and decidability. *Theoretical Computer Science 750* (2018), 2–23.

[21] Knuth, D. E. An empirical study of fortran programs. *Software: Practice and experience 1*, 2 (1971), 105–133.

[22] Lakin, S. R., and Thomas, R. M. Context-sensitive decision problems in groups. In *International Conference on Developments in Language Theory* (2004), Springer, pp. 296–307.

[23] LeCun, Y., Bengio, Y., and Hinton, G. Deep learning. *nature 521*, 7553 (2015), 436–444.

[24] Merrill, W. Sequential neural networks as automata. *ArXiv abs/1906.01615* (2019).

[25] MILLER, C. F. Decision problems for groups—survey and reflections. In *Algorithms and classification in combinatorial group theory*. Springer, 1992, pp. 1–59.

[26] MULLER, D. E., AND SCHUPP, P. E. The theory of ends, pushdown automata, and second-order logic. *Theoretical Computer Science 37* (1985), 51–75.

[27] NAITZAT, G., ZHITNIKOV, A., AND LIM, L.-H. Topology of deep neural networks. *Journal of Machine Learning Research 21*, 184 (2020), 1–40.

[28] PAREKH, R., AND HONAVAR, V. Learning dfa from simple examples. *Machine Learning 44*, 1 (2001), 9–35.

[29] RODRIGUEZ, P., AND WILES, J. Recurrent neural networks can learn to implement symbol-sensitive counting. In *Advances in Neural Information Processing Systems 10*, M. I. Jordan, M. J. Kearns, and S. A. Solla, Eds. MIT Press, 1998, pp. 87–93.

[30] SASSAMAN, L., PATTERSON, M. L., BRATUS, S., AND LOCASTO, M. E. Security applications of formal language theory. *IEEE Systems Journal 7*, 3 (2013), 489–500.

[31] SHAPIRO, M. A note on context sensitive languages and word problems. *arXiv preprint math/9306204* (1993).

[32] SIEGELMANN, H. T., AND SONTAG, E. D. Analog computation via neural networks. *Theoretical Computer Science 131*, 2 (1994), 331 – 360.

[33] STEIJVERS, M., AND GRÜNWALD, P. A recurrent network that performs a context-sensitive prediction task. In *Proceedings of the 18th Annual Conference of the Cognitive Science Society* (1996), pp. 335–339.

[34] SUZGUN, M., GEHRMANN, S., BELINKOV, Y., AND SHIEBER, S. M. Memory-augmented recurrent neural networks can learn generalized dyck languages. *ArXiv abs/1911.03329* (2019).

[35] TOMITA, M. Dynamic construction of finite automata from examples using hill-climbing. In *Proceedings of the Fourth Annual Conference of the Cognitive Science Society* (Ann Arbor, Michigan, 1982), pp. 105–108.

[36] TONKES, B., AND WILES, J. Learning a context-free task with a recurrent neural network: An analysis of stability. In *In Proceedings of the Fourth Biennial Conference of the Australasian Cognitive Science Society* (1997), Citeseer.

[37] WEISS, G., GOLDBERG, Y., AND YAHAV, E. On the practical computational power of finite precision rnns for language recognition. *ArXiv abs/1805.04908* (2018).

[38] ZOMORODIAN, A., AND CARLSSON, G. Computing persistent homology. *Discrete & Computational Geometry 33*, 2 (2005), 249–274.