What makes Models Compositional? A Neuro-Symbolic Theoretical View (Extended Abstract)

Parikshit Ram[1,*], Tim Klinger[1] and Alexander G. Gray[2,3]

[1]IBM Research
[2]Centaur AI Institute
[3]Purdue University

Abstract

Compositionality is thought to be a key component of language, and various compositional benchmarks have been developed to empirically probe the compositional generalization of existing sequence processing models. These benchmarks often highlight failures of existing models, but it is not clear why these models fail in this way. In this paper, we seek to theoretically understand the role the compositional structure of the models plays in these failures and how this structure relates to their expressivity and sample complexity. We propose a general neuro-symbolic definition of compositional functions and their compositional complexity. We then show how various existing general and special purpose sequence processing models (such as recurrent, convolution and attention-based ones) fit this definition and use it to analyze their compositional complexity.

1. Introduction

Compositionality is assumed to be integral to language processing [1]. Generalizing in a compositional manner or *compositional generalization* is of high interest when learning with sequences since it can enable a (learned) model to generalize well to a possibly infinite domain of sequences while learning from only a small number of examples. With this motivation, there has been interest in quantifying the compositional generalization (CG) of sequence or language models. This has led to various language modeling benchmarks [2, 3, 4, 5]. These benchmarks empirically probe the compositionality of language models, often demonstrating the lack of CG. These highlight examples on which the models fail, but there is no precise understanding of why the failures occur. Nonetheless, various novel methods with improved compositional generalization (as measured by these benchmarks) have been developed [6, 7, 8], utilizing specialized models with compositional inductive biases. However, the area of CG is still *lacks a mathematical definition and measure of compositionality.*

Our contributions. Inspired by existing discussions, and recent solutions for CG benchmarks, We make the following contributions: [1]

- We propose a general modular definition of "compositional functions" to facilitate concrete understanding of the expressiveness and generalization of such functions, and propose the notion of "compositional complexity" to quantify the complexity of such functions. Our definition teases out what we believe to be the "neural" (continuous functions) and the "symbolic" (discrete functions) parts of a model.
- We demonstrate the flexibility of this definition by highlighting how various existing sequence processing models such as recurrent models [11], convolutional models and transformer based models [12] fit this definition, and how complex their compositions are.

LNSAI 2024: First International Workshop on Logical Foundations of Neuro-Symbolic AI, August 05, 2024, Jeju, South Korea
*Corresponding author.
parikshit.ram@ibm.com (P. Ram); tklinger@us.ibm.com (T. Klinger); skirmilitor@gmail.com (A. G. Gray)
https://research.ibm.com/people/parikshit-ram (P. Ram); https://research.ibm.com/people/tim-klinger (T. Klinger); https://www.linkedin.com/in/alexander-gray-b554b64/ (A. G. Gray)
© 2024 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

[1]A preliminary version of this work [9] was previously presented at the KBCG@IJCAI23 workshop, while the extended version for this submission [10] can be found at https://www.arxiv.org/abs/2405.02350.

CEUR-WS.org/Vol-3819/short3.pdf

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073

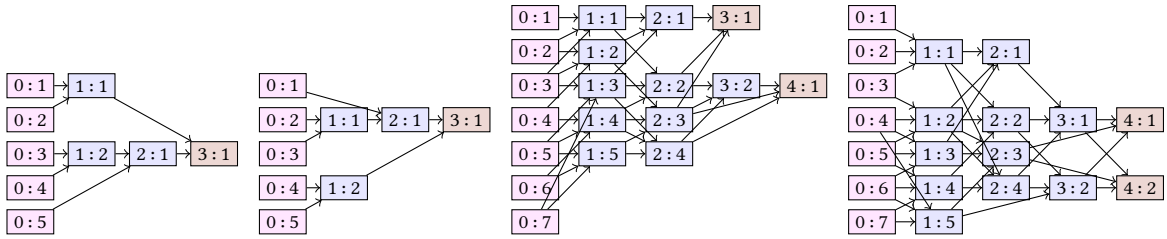


Figure 1: **cDAG**s for $f(X)$ (left) and $f(\bar{X})$ (center left) in Example 1 and $f(X)$ (center right) and $f(\bar{X})$ (right) in Example 2. Nodes are labeled $l{:}i$ (level l, index i). Sources are Fuchsia, sinks Sepia, and internal nodes Blue.

2. Defining and Quantifying Compositionality

We define compositional functions $f : \mathcal{X} \to \mathcal{Y}$ with the domain $\mathcal{X}$ of input sequences $X = \{x_1, \ldots, x_L\}$ of atoms or tokens $x_i \in \mathcal{I}$ from an input dictionary $\mathcal{I}$. The range $\mathcal{Y}$ of f can be $\mathbb{R}$ for regression, $\{0, 1\}$ for binary classification, or $\mathcal{I}$ for next token prediction.

Definition 1. *To define f, we need the following components:*

- *Token encoder $e : \mathcal{I} \times \mathbb{N} \to \mathcal{H}$ (latent space), with $e_i = e(x_i, i) \in \mathcal{H}$ encoding the i^{th} token in $X \in \mathcal{X}$.*
- *A computation directed acyclic graph (DAG) or **cDAG** $D : \mathcal{X} \to \mathcal{D}$, where $\mathcal{D}$ is the space of DAGs, and $D(X)$ defines the hierarchical processing of a sequence X. $D(X)$ can also be viewed as the trace of program used by function f to process X. We will describe this in further detail soon.*
- *Span processor $g : \mathcal{H}^k \to \mathcal{H}$ maps k terms in the latent space into a new term in the latent space.*
- *Read-out function $h : \mathcal{H}^m \to \mathcal{Y}$ maps the final set of terms in the latent space to the output space $\mathcal{Y}$.*

With $g^{\otimes D(X)}$ denoting the recursive operation of g over $D(X)$, we define a compositional function as:

$$f(X) = h\left(g^{\otimes D(X)}(e(x_1, 1), \ldots, e(x_L, L))\right). \tag{1}$$

A **computation DAG** or **cDAG** $D(X) \triangleq \{N(X), E(X)\}$ for a specific input sequence $X \in \mathcal{X}$ can depend on X or be pre-specified. This **cDAG** is a leveled DAG with set of nodes $N(X)$ and edges $E(X)$. Each node $n \triangleq (l : i) \in N(X)$ has a level l and index i. The recursive application of g over $D(X)$ induces a value $v_{l:i} \in \mathcal{H}$ for each internal node $n \in N(X)$. The sources is $N(X)$ have level 0, and there is one source for each $x_i \in X, i \in [\![L]\!] \triangleq \{1, \ldots, L\}$ with index i and value $v_{0:i} = e(x_i, i) \in \mathcal{H}$. There are m sinks in $N(X)$, and at most k incoming edges and q outgoing edges at any node. For an internal node $n \in N(X)$ with k parents $P(n)$, the value $v_{l:i} = g(v_{l_1:i_1}, \ldots, v_{l_k:i_k}) \in \mathcal{H}$ where $v_{l_j:i_j}$ is the value of the j^{th} parent in $P(n)$. One way to interpret this **cDAG** is as the trace of "forward-pass" for inference.

We consider the explicit **cDAG** because it allows us to see how the different elements $x_i, i \in [\![L]\!]$ of the input sequence X are hierarchically composed to obtain the output. This will allow us to study the complexity of any compositional function. A "simple" **cDAG**, where all source nodes just connect to a single sink node, would be "applicable" to all functions, but it does not allow us to study it in an interesting manner. When we study the compositional functions induced by general purpose models (such as recurrent, convolutional or transformer models), we will see that some models have explicit **cDAG**s with more structure, while others have less structured explicit **cDAG**s, but there are implicit structures induced in the **cDAG**; whenever possible, we will explicitly state this implicit structure and study its properties. *From a neuro-symbolic perspective [13, 14], this explicit **cDAG** can be seen as the symbolic part, while the e, g, h are the neural*; note that, in some models, this symbolic **cDAG** might be created with neural elements, while in others, the **cDAG** might be obtained with a symbolic grammar. This neuro-symbolic view offers a novel theoretical understanding of compositionality.

The **span processor** $g : \mathcal{H}^k \to \mathcal{H}$ takes as input k elements from the latent space $\mathcal{H}$ and outputs an element in $\mathcal{H}$. While the definition implies that the same g needs to be operated recursively over the **cDAG** $D(X)$, there is no restriction on the inputs and output of g regarding the information encoded in the latent space. For example, if the level l of any node $l{:}i$ is encoded into its value $v_{l:i}$, then the g

will behave differently across levels (*level-dependent*); if the index i of the node $l{:}i$ is encoded into its value, then g will be sensitive to the positional information (*order-dependent*); if the value of a node includes the type of the node (for example, a non-terminal in a grammar), then g can be *type-dependent*. Our definition states that the arity of the span processor $g : \mathcal{H}^k \to \mathcal{H}$ is k. We do so for the ease of exposition, though our definition can incorporate more flexible span processors (see Ram et al. [10, Appendix A.2]).

The **read-out function** $h : \mathcal{H}^m \to \mathcal{Y}$ finally maps m elements in the latent space to the output space $\mathcal{Y}$. This separation between g and h was necessary in our proposed definition because we require g to be operable recursively, and thus g can operate in a latent space $\mathcal{H}$ distinct from $\mathcal{Y}$. In some applications, $\mathcal{H} \supseteq \mathcal{Y}$, in which case, h can be an identity function. There are couple of aspects of this read-out function we wish to discuss explicitly – (i) We assume that h is specifically *non-compositional* and processes its input without breaking it up into any sub-problems; we explicitly define the compositional function f separating out g, D, h, where g (neural) and D (symbolic) represent the compositional part. (ii) We require h to have a fixed-arity of m since g and D are aggregating the information over the input.

In the following, we will illustrate Definition 1 with a couple of examples:

Example 1. *Figure 1 (left) shows the* **cDAG** $D(X)$ *for a compositional* f *on* $X = [x_1, \ldots, x_5]$*, with* $f(X) = h\left(g\left(g\left(e_1, e_2\right), g\left(g(e_3, e_4), e_5\right)\right)\right)$*,* $k = 2$ *in-degree,* $q = 1$ *out-degree,* $m = 1$ *sink,* $e_i = e(x_i, i) \in \mathcal{H}$*, span-processor* $g : \mathcal{H}^2 \to \mathcal{H}$*, and read-out function* $h : \mathcal{H} \to \mathcal{Y}$*. The values* $v_{0:i} = e_i$ *for sources* $0:i$*,* $i \in \{1, \ldots, 5\}$*, and the internal node values are:* $v_{1:1} \leftarrow g(e_1, e_2)$*,* $v_{1:2} \leftarrow g(e_3, e_4)$*,* $v_{2:1} \leftarrow g(v_{1:2}, e_5)$*,* $v_{3:1} \leftarrow g(v_{1:1}, v_{2:1})$*.* h *operates on* $v_{3:1}$ *at sink* $3:1$*. Figure 1 (center left) shows the* **cDAG** $D(\bar{X})$ *of the same* f *on* $\bar{X} \neq X$ *with the same* $k = 2, q = 1, m = 1$ *and* $f(X') = h\left(g(g(e_1, g(e_2, e_3)), g(e_4, e_5))\right)$*. Note that* $D(X)$ *is not the same as* $D(\bar{X})$*.*

Example 2. *Figure 1 (center right) shows the* **cDAG** $\mathsf{D}(X)$ *for a compositional* f *on* $X = [x_1, \ldots, x_7]$*, with* $\mathsf{f}(X) = \mathsf{h}\left(v_{4:1}, v_{3:1}\right)$*,* $k = 3$ *maximum in-degree,* $q = 3$ *maximum out-degree,* $m = 2$ *sinks,* $e_i = e(x_i, i) \in \mathcal{H}$*, span processor* $\mathsf{g} : \mathcal{H}^3 \to \mathcal{H}$*, and read-out function* $\mathsf{h} : \mathcal{H}^2 \to \mathcal{Y}$*. The source values* $v_{0:i} = e_i$ *for each* $i \in \{1, \ldots, 7\}$*, and the internal node values are:* $v_{1:1} \leftarrow \mathsf{g}(e_1, e_2, e_3)$*,* $v_{1:2} \leftarrow \mathsf{g}(e_2, e_3, e_4)$*,* $v_{1:3} \leftarrow \mathsf{g}(e_3, e_5, e_7)$*,* $v_{1:4} \leftarrow \mathsf{g}(e_4, e_5, e_6)$*,* $v_{1:5} \leftarrow \mathsf{g}(e_5, e_6, e_7)$*,* $v_{2:1} \leftarrow \mathsf{g}(v_{1:1}, v_{1:2}, v_{1:3})$*,* $v_{2:2} \leftarrow \mathsf{g}(v_{1:1}, v_{1:3}, v_{1:4})$*,* $v_{2:3} \leftarrow \mathsf{g}(v_{1:2}, v_{1:4}, v_{1:5})$*,* $v_{2:4} \leftarrow \mathsf{g}(v_{1:3}, v_{1:4}, v_{1:5})$*,* $v_{3:1} \leftarrow \mathsf{g}(v_{2:1}, v_{2:2}, v_{2:3})$*,* $v_{3:2} \leftarrow \mathsf{g}(v_{2:2}, v_{2:3}, v_{2:4})$*,* $v_{4:1} \leftarrow \mathsf{g}(v_{3:2}, v_{2:3}, v_{2:4})$*.* h *operates on* $v_{3:1}$ *and* $v_{4:1}$ *at sinks* $3:1$ *and* $4:1$*. Figure 1 (right) shows the* **cDAG** $\mathsf{D}(\bar{X})$ *of the same* f *on* $\bar{X} \neq X$ *with the same* $k = 3, q = 3, m = 2$*.*

While Example 1 is a simple compositional function on a sequence, Example 2 is a more sophisticated one. This is to highlight that our proposed Definition 1 can handle functions which require more complex interactions between the tokens in a sequence. Example 1 has a **cDAG** with a maximum out-degree $q = 1$, implying a single path from any source to a sink. Example 2 has a **cDAG** with a maximum out-degree $q = 3$ across all levels in the DAG, implying that there can be a large number of paths to any sink from a source. This allows the definition to include functions where certain tokens in the sequence are of much higher importance to the output than others. These examples also highlight that edges in the **cDAG** are allowed to skip levels, and the sinks can be from different levels, further highlighting the compositional flexibility.

We like to remark on a couple of points here: (i) Through these examples, we show that our definition explicitly considers how the problem of sequence processing is broken up into sub-problems – the **cDAG** embodies how disjoint or intertwined these "sub-problems" are by explicitly considering the computation hierarchy. (ii) For input sequences $X, \bar{X}$ from the same problem domain, and the same compositional function f, we allow the **cDAG** to be different – **cDAG** $D(X)$ can be input-dependent – thereby allowing different input sequences to have different sub-problem hierarchies. At a non-technical level, we also believe that our proposed Definition 1 connects intuitively to existing definitions:

$$\underbrace{\text{The meaning of the whole}}_{f:\mathcal{X}\to\mathcal{Y}} \text{ is a } \underbrace{\text{function}}_{h:\mathcal{H}^m\to\mathcal{Y}} \text{ of } \underbrace{\text{the meanings of the parts}}_{g:\mathcal{H}^k\to\mathcal{H}}$$
$$\text{and of } \underbrace{\text{the way they are syntactically combined.}}_{D:\mathcal{X}\to\mathcal{D}}$$

Both Examples 1 and 2 can be seen as compositional functions, but Example 2 is clearly a more complex composition. In addition to its intuitive nature, our proposed definition allows us to understand *how complex the compositionality is* beyond just stating if a function is compositional. The **compositional complexity** of f depends on the functions g, h, e as well as the **cDAG** function D that drives the computation. For a sequence X of length L, $D(X)$ has L source nodes, maximum in-degree of k (controlling the span size for g), m sink nodes (controlling the capacity of h), maximum out-degree of q (quantifying the "localism" of the effect of a node). However, these do not explicitly incorporate the fact that changes to nodes at lower levels of the **cDAG** *can* have a larger effect on the output than changes to nodes at higher levels of the **cDAG**. We propose a new quantification – the *locus of influence* (LoI):

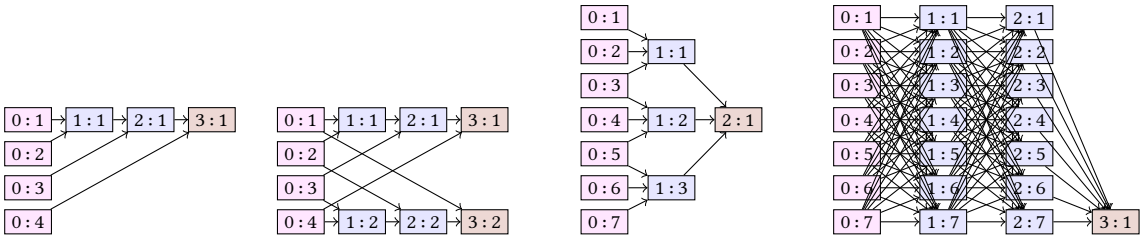
Definition 2 (LoI of a source node)**.** *Consider a function f with components e, D, g, h (Definition 1). Let $(v_{n_1}, \ldots, v_{n_j}, \ldots, v_{n_k}) \in \mathcal{H}^k$ be any input to the span processor g, with $v_n = g(v_{n_1}, \ldots, v_{n_j}, \ldots, v_{n_k})$ its output. Let $\varepsilon \in \mathcal{H}$ be a "perturbation" to the j^{th} argument to g, $j \in [\![k]\!]$, resulting in the perturbed output $v_n^j(\varepsilon) = g(v_{n_1}, \ldots, v_{n_j} + \varepsilon, \ldots, v_{n_k})$. Let $c > 0$ be a constant such that $\forall j \in [\![k]\!]$, $\forall \varepsilon \in \mathcal{H}$, $\left\| v_n - v_n^j(\varepsilon) \right\| \leq c\|\varepsilon\|$. For a sequence $X \in \mathcal{X}$ of length L, and a source node $0:i$ in $D(X)$, let $P(x_i)$ be the set of all unique paths from $0:i$ to any of the sink nodes in $D(X)$. The* **absolute LoI** *of index i is $\delta_i = \sum_{P \in P(x_i)} c^{|P|}$, with $|P|$ as the length of a path $P \in P(x_i)$, and the* **relative LoI** *is $\beta_i = \delta_i / \sum_{j \in [\![L]\!]} \delta_j$.*

This definition of the complexity of composition incorporates both the complexity of the **cDAG** $D(X)$ and the complexity of the span processor $g : \mathcal{H}^k \to \mathcal{H}$ in terms of its smoothness, with higher values of c indicating more complex (less smooth) g. The absolute LoI δ_i incorporates the effect of longer paths, with the effect growing with path length, and corresponds to the sensitivity of the compositional function output to any one input token in the sequence.

The smaller the absolute LoI δ_i of any input index i, more local its effect, and thus more structure that can be transferred between examples if x_i is replaced with something else. A relative LoI β_i greater than $1/L$ denotes that the input index i (and thus input token x_i) has an out-sized effect on $D(X)$ (and thus the computation) compared to the other indices (tokens). In Example 1 (left), $\delta_1 = c^2, \beta_1 = {}^1\!/_{2c+3} < {}^1\!/_5$ while $\delta_3 = c^3, \beta_3 = {}^c\!/_{2c+3} > {}^1\!/_5$, implying that x_3 has more influence (absolute and relative) function than x_1 (assuming $c > 1$). In Example 2 (left), $\delta_1 = c^4 + 2c^3, \beta_1 = {}^{c+2}\!/_{27c+39} \approx {}^1\!/_{22} < {}^1\!/_7$, while $\delta_5 = 7c^4 + 9c^3, \beta_5 = {}^{7c+9}\!/_{27c+39} \approx {}^1\!/_4 > {}^1\!/_7$, hence x_5 has a significantly larger influence than x_1. We utilize the LoI to define the complexity of a compositional function, and a class of such compositional functions:

Definition 3. *A function $f : \mathcal{X} \to \mathcal{Y}$ with components g, h, e, D is $(k, q, m, \boldsymbol{\delta}, \boldsymbol{\beta})$-compositional if, for any $X \in \mathcal{X}$ of length L, the* **cDAG** *$D(X)$ has a in-degree of k, maximum outgoing degree of q, and m sink nodes, and for $\forall i \in [\![L]\!], \delta_i \leq \boldsymbol{\delta}$, and $\beta_i \leq \boldsymbol{\beta} \in [1/L, 1)$. We denote with $\mathcal{F}$ a class of such $(k, q, m, \boldsymbol{\delta}, \boldsymbol{\beta})$-compositional functions.*

A small $\boldsymbol{\delta}$ and a $\boldsymbol{\beta}$ close to $1/L$ signifies a function that possesses a high level of localism across all input sequences and tokens in its domain. While this function has the most structure, it might not be suitable for practical purposes. A high $\boldsymbol{\delta}$ and a $\boldsymbol{\beta}$ close to $1/L$ signifies a very complex function where there is a lot of interaction between all the input tokens in all input sequences, making it hard to exploit any compositional structure in the function. A high $\boldsymbol{\delta}$ and a $\boldsymbol{\beta}$ significantly higher than $1/L$ indicates an interesting class of functions where, some input tokens *can have a high influence over the function computation*, but, for most tokens, there is a compositional structure in the function that can be exploited. This intuitively seems to be an interesting and more practical class of compositional functions since assuming all tokens have an equal level of relative influence seems quite restrictive.



(a) Unidirectional rec. (b) Bidirectional rec (c) Convolve + Pool (d) Transformer

Figure 2: The **cDAG** for various existing sequence processing models such as the unidirectional and bidirectional recurrence models, convolutional models and attention based transformer models.

Model	**Input-dependent cDAG**	**Arbitrary Len**	(k, q, m)	$\boldsymbol{\delta}$	$\boldsymbol{\beta}$
Unidirectional RNN (2a)	✗	✓	$(2, 1, 1)$	c^{L-1}	$1/2$
Bidirectional RNN (2b)	✗	✓	$(2, 2, 2)$	c^{L-1}	$1/4$
Convolve (width w) + pool (width p) (2c)	†	‡	$(w{+}p, w, m)$	$c^{\log L}$	$\frac{2/L}{1+1/p}$
Transformer (2d)	✗	✓	$(L, L, 1)$	$(Lc)^M$	$1/L$
Transformer w/ K-sparse attention	✓	✓	$(K, L, 1)$	$L(Kc)^M$	$1/K$

Table 1
Compositional complexities of existing models with input sequences of length L. LoI is specified approximately for the ease of exposition. †: Convolve+Pool induces input-dependent **cDAG**s for max/min-pool, not for avg/sum-pool. ‡: The number of sinks m needs to be specified for Convolve+pool, and the model can handle arbitrary length if it can recursively apply Convolve+pool until the number of nodes at a level is reduced to m.

In Fig. 2, we re-express existing sequence processing models as per our definition, teasing out the symbolic **cDAG** (and the neural g, h), and we present their (simplified) compositional complexity in Table 1 assuming that all models classes utilize span processors g with the same smoothness parameter c for the ease of exposition. This highlights the flexibility and utility of our proposed quantification of compositionality (see Ram et al. [10, Section 4] for more details and models). Beyond the properties of the **cDAG** (the in-degree k, out-degree q and number of sink nodes m) and the upper bounds on the absolute LoI $\boldsymbol{\delta}$ and relative LoI $\boldsymbol{\beta}$, we also highlight two properties: (i) Whether the model utilizes (implicitly or explicitly) an input-dependent **cDAG** (that is, $D(X)$ is not the same DAG for all X of length L), and (ii) Whether the same model is able to operate on arbitrary length input sequences. The use of input-dependent **cDAG** has implications in terms of the expressivity of the model – it can be shown that functions from a model class (with compositional complexites $\boldsymbol{\delta}, \boldsymbol{\beta}$) with a fixed input-agnostic **cDAG** cannot approximate functions from a model class of matching compositional complexity (that is, same compositional complexity $\boldsymbol{\delta}, \boldsymbol{\beta}$) that utilize input-dependent **cDAG**s. Ram et al. [10, Theorem 1] show that the approximation is lower and upper bounded by $\mathcal{O}(\boldsymbol{\delta})$ and $\mathcal{O}(\boldsymbol{\delta}/\boldsymbol{\beta})$ respectively. This implies that a higher value of absolute compositional complexity $\boldsymbol{\delta}$, and a smaller relative compositional complexity $\boldsymbol{\beta}$ adversely affect the approximation guarantees. The absolute compositional complexity $\boldsymbol{\delta}$ has been shown to be directly tied to the generalization gap for a learned compositional function, with higher $\boldsymbol{\delta}$ implying worse systematic generalization guarantee [10, Theorem 2]. The ability to operate on arbitrary length sequences is a prerequisite to the ability of a model to possess *length generalization* or *productivity* – the ability to generalize to sequences of larger lengths than those seen during training. We will be pursuing length generalization in our future work.

3. Conclusion

In this paper, we briefly present our novel definition of compositional functions that explicitly separates out the neural and symbolic aspects of a model for the ease of analysis. We also present a notion of

compositional complexity that quantifies the complexity with which the tokens in an input sequence are put together to get to the output. We briefly highlight the generality and utility of this definition by demonstrating how existing sequence processing models fit into this definition.

References

[1] P. Pagin, D. Westerståhl, Compositionality I: Definitions and variants, Philosophy Compass 5 (2010) 250–264. URL: https://compass.onlinelibrary.wiley.com/doi/abs/10.1111/j.1747-9991.2009.00228.x.

[2] B. Lake, M. Baroni, Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks, in: International Conference on Machine Learning, PMLR, 2018, pp. 2873–2882. URL: https://proceedings.mlr.press/v80/lake18a.html.

[3] N. Kim, T. Linzen, COGS: A compositional generalization challenge based on semantic interpretation, in: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2020, pp. 9087–9105. URL: https://aclanthology.org/2020.emnlp-main.731.pdf.

[4] D. Hupkes, V. Dankers, M. Mul, E. Bruni, Compositionality decomposed: how do neural networks generalise?, Journal of Artificial Intelligence Research 67 (2020) 757–795. URL: https://jair.org/index.php/jair/article/view/11674.

[5] T. Klinger, D. Adjodah, V. Marois, J. Joseph, M. Riemer, A. S. Pentland, M. Campbell, A study of compositional generalization in neural models, arXiv preprint arXiv:2006.09437 (2020). URL: https://arxiv.org/abs/2006.09437.

[6] Y. Li, L. Zhao, J. Wang, J. Hestness, Compositional generalization for primitive substitutions, in: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), 2019, pp. 4284–4293. URL: https://aclanthology.org/D19-1438/.

[7] Q. Liu, S. An, J.-G. Lou, B. Chen, Z. Lin, Y. Gao, B. Zhou, N. Zheng, D. Zhang, Compositional generalization by learning analytical expressions, Advances in Neural Information Processing Systems 33 (2020). URL: https://proceedings.neurips.cc/paper_files/paper/2020/file/83adc9225e4deb67d7ce42d58fe5157c-Paper.pdf.

[8] M. Nye, A. Solar-Lezama, J. Tenenbaum, B. M. Lake, Learning compositional rules via neural program synthesis, Advances in Neural Information Processing Systems 33 (2020) 10832–10842. URL: https://proceedings.neurips.cc/paper_files/paper/2020/file/7a685d9edd95508471a9d3d6fcace432-Paper.pdf.

[9] P. Ram, T. Klinger, A. G. Gray, How compositional is a model?, in: International Joint Conference on Artificial Intelligence 2023 Workshop on Knowledge-Based Compositional Generalization, 2023. URL: https://openreview.net/forum?id=OImyRhNLv3.

[10] P. Ram, T. Klinger, A. G. Gray, What makes Models Compositional? A Theoretical View: With Supplement, arXiv preprint arXiv:2405.02350 (2024). URL: https://arxiv.org/abs/2405.02350.

[11] S. Hochreiter, J. Schmidhuber, Lstm can solve hard long time lag problems, Advances in neural information processing systems 9 (1996). URL: https://proceedings.neurips.cc/paper/1996/file/a4d2f0d23dcc84ce983ff9157f8b7f88-Paper.pdf.

[12] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Advances in neural information processing systems 30 (2017). URL: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[13] M. K. Sarker, L. Zhou, A. Eberhart, P. Hitzler, Neuro-symbolic artificial intelligence, AI Communications 34 (2021) 197–209. URL: https://arxiv.org/pdf/2105.05330.pdf.

[14] A. d. Garcez, S. Bader, H. Bowman, L. C. Lamb, L. de Penning, B. Illuminoo, H. Poon, C. G. Zaverucha, Neural-symbolic learning and reasoning: A survey and interpretation, Neuro-Symbolic Artificial Intelligence: The State of the Art 342 (2022) 327. URL: https://arxiv.org/pdf/1711.03902.pdf.