How Compositional is a Model?

Abstract

We present a formal definition of a general class of compositional functions, and show how various existing models fit this definition. Given this general definition, we discuss a notion of complexity of these compositional functions, and show how this complexity affects the expressivity and compositional generalization of these functions.

1 Introduction

Compositionality is often assumed to be integral to language processing [Pagin and Westerståhl, 2010a; Pagin and Westerståhl, 2010b]. *Compositional generalization* or CG is of high interest when learning with sequences since it can enable a (learned) model to generalize to a possibly infinite domain of sequences while learning from a small number of examples (assuming that the ground truth function is also compositional). With this motivation, there has been an interest in quantifying the CG of language models. This has led to various benchmarks such as SCAN [Lake and Baroni, 2018], CFQ [Keysers *et al.*, 2019], COGS [Kim and Linzen, 2020] and others [Andreas, 2018; Hupkes *et al.*, 2020], although CG can also be of interest in vision [Klinger *et al.*, 2020].

These benchmarks often demonstrate the lack of CG in standard off-the-shelf language models. Various novel methods have been proposed to improve the CG on these benchmarks [Russin *et al.*, 2019; Gordon *et al.*, 2019; Li *et al.*, 2019; Liu *et al.*, 2020; Nye *et al.*, 2020; Liu *et al.*, 2021] by utilizing specialized models with compositionality promoting inductive biases. Since the ground truth in these benchmarks are compositional, such regularized models exhibit CG.

However, the area of CG is still missing a mathematical definition and measure of compositionality. We provide one, which can be used to characterize present and future models.

2 Defining Compositionality

One definition of compositionality [Pagin and Westerståhl, 2010a] is that the meaning function $\mu(\cdot)$ of a meaningful expression $\alpha(u_1, \ldots, u_n)$ is given by $\mu(\alpha(u_1, \ldots, u_n)) = r_\alpha(\mu(u_1), \ldots, \mu(u_n))$, where α is a rule applied to the subterms u_i in an expression, and r_α is a meaning operation that depends on α. This definition inspired some inductive biases [Liu *et al.*, 2021]. A non-technical phrasing of the principle of compositionality [Partee and others, 1995] is that *"the meaning of a whole is a function the meanings of the parts, and of the way they are syntactically combined."* This is the guiding principle of some models that are specifically designed for CG [Gordon *et al.*, 2019; Li *et al.*, 2019; Liu *et al.*, 2020; Liu *et al.*, 2021]. Among the expected properties of compositional functions are *systematicity* – the ability to *consistently handle unknown combinations of known parts*, and *productivity* – the ability to *handle arbitrary length sequences*. Other properties are localism, substitutivity and over-generalization, but we will focus on systematicity and productivity here. Note that, while compositionality is a property of any function, systematicity or systematic generalization is a property of a learned model that has learned from some examples ("known parts") and is expected to generalize to unseen examples ("unknown combinations"). Productivity includes a basic prerequisite ability of taking sequences of any length as input and producing predictions, which then need to be accurate. Our goal is to ground this principle into a mathematical form that allows us to *quantify* the compositionality of models, and understand how this affects CG.

We define compositional functions $f : \mathcal{X} \to \mathcal{Y}$ with the domain $\mathcal{X}$ of input sequences $X = \{x_1, \ldots, x_L\}$ with tokens $x_i \in \mathcal{I}$ (input dictionary). The range $\mathcal{Y}$ can be $\mathbb{R}$ (regression), $\{0, 1\}$ (classification), or even $\mathcal{I}$ (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 ith token in $X \in \mathcal{X}$.*
- *A computation DAG or cDAG $D : \mathcal{X} \to \mathcal{D}$ (the space of DAGs), with $D(X)$ defining the hierarchical processing of a sequence 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 $h : \mathcal{H}^m \to \mathcal{Y}$ which maps the final set of terms in the latent space to the output space $\mathcal{Y}$.*

Given the above, we define a compositional function as

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

where $g^{\otimes D(X)}$ is the recursive operation of g over $D(X)$.

<u>*cDAG.*</u> A cDAG $D(X) \triangleq \{N(X), E(X)\}$ can depend on X, and is a leveled DAG with set of nodes $N(X)$ and edges $E(X)$. Each node $n \in N(X)$ has a level l and index i – the level of n is one more than the highest level of any of its k

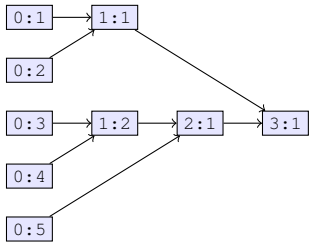

Figure 1: cDAG for the function $f(x_1, x_2, x_3, x_4, x_5)$ defined as $h\left(g\left(e_1, e_2\right), g\left(g(e_3, e_4), e_5\right)\right)$, with $k = 2$ span size, $m = 1$ sink nodes in the cDAG, and $e_i = e(x_i, i) \in \mathcal{H}$. Each node is labeled with $l{:}i$, with l as the node level, and i the node index in level l. The value of node $\texttt{0:}i$ is e_i for each $i \in \{1, \ldots, 5\}$, and the values for the non-source nodes are: $\texttt{1:1} \leftarrow g(e_1, e_2)$, $\texttt{1:2} \leftarrow g(e_3, e_4)$, $\texttt{2:1} \leftarrow g(g(e_3, e_4), e_5)$, and $\texttt{3:1} \leftarrow g(g(e_1, e_2), g(g(e_3, e_4), e_5))$.

parents, the index of n (at its level) is based on the sort order of the sorted list of the (level, index) tuples of its parents $P(n) = \{n' \in N(X) : (n' \to n) \in E(X)\}$. The recursive application of g over $D(X)$ induces a value $v_n \in \mathcal{H}$ for each non-source node $n \in N(X)$. The source nodes $n \in N(X)$ have level 0, with one for each $x_i \in X, i \in [L]$ with index i and value $v_n = e(x_i, i) \in \mathcal{H}$. There are at most m sink nodes in $N(X)$, and at most k incoming edges at any node. For a non-source node n at level l with index i, and sorted k parents $P(n)$, the value $v_n = g(v_1, \ldots, v_k) \in \mathcal{H}$ where v_j is the value of the j-th parent in $P(n)$. Figure 1 shows the cDAG for the compositional function $f(X) = h\left(g\left(e_1, e_2\right), g\left(g(e_3, e_4), e_5\right)\right)$, with $k = 2$ span size, $m = 1$ sink nodes, and $e_i = e(x_i, i) \in \mathcal{H}$.

Complexity of a compositional function. This depends on the functions g, h, e as well as the cDAG D that drives the computation. For a sequence X of length L, $D(X)$ has L source nodes, maximum in-degree of k (controls the span size for g), m sink nodes (controls the capacity of h), maximum out-degree of q (quantifies the locality of any node's effect). We also quantify the *locus of influence* or LoI of any source node:

Definition 2 (LoI)**.** *For any source node $\texttt{0:}i$ in $D(X)$, let P_i be the set of all unique paths from $\texttt{0:}i$ to any of the sink nodes in $D(X)$, and let R_i be the set of all unique edges in P_i. Then we define LoI of x_i as $\beta(x_i) \triangleq {}^{|R_i|}/_{|E(X)|}$.*

Smaller the LoI of any input token x_i, more local its effect, and thus more structure that can be transferred between examples if x_i is replaced with something else. Finally, we can now define a class of compositional functions:

Definition 3. *A function $f : \mathcal{X} \to \mathcal{Y}$ with components g, h, e, D is (k, q, m, β_L)-compositional if, for any $X \in \mathcal{X}$, $|X| = L$, the cDAG $D(X)$ has a maximum incoming degree, outgoing degree and sink nodes of k, q, m respectively, and for $\forall x_i \in X, \beta(x_i) \leq \beta_L$. Let $\mathcal{F}$ be a class of such functions.*

A smaller β_L signifies a function that possesses some level of localism across all input sequences in its domain.

Expressivity of data-dependent cDAG. Some function classes induce a data-dependent cDAG while others do not. We study the ability of a function with a data-independent cDAG to approximate one with a data-dependent one:

Proposition 1. *Consider $\mathcal{F}$ as in Definition 3, sequences $X \in \mathcal{X}$ of length L, and a data-independent cDAG $D'_L(X)$ (i.e. $D'_L(X) = D'_L \, \forall X \in \mathcal{X}, |X| = L$). Then the difference between $f \in \mathcal{F}$ (components h, g, and a data-dependent D) and $f' \in \mathcal{F}$ (components h', g', D'_L), with a common token encoder $e : \mathcal{I} \to \mathcal{H}$ is lower-bounded as*

$$\max_{h,g,D,X} \min_{h',g',D'_L} \left| h(g^{\otimes D(X)}(e(X)) - h'(g'^{\otimes D'_L}(e(X)) \right| \geq \mathcal{O}(\beta_L).$$

This results highlights the limitation of a fixed cDAG – even if we select the best possible fixed cDAG D'_L, and corresponding components g', h' (note the $\min_{h',g',D'_L}$), there are still $f \in \mathcal{F}$ with components h, g, D which cannot be approximated sufficiently well (note the $\max_{h,g,D}$). Furthermore, it highlights that, as β_L grows, this lower bound grows, implying that the larger the LoI of the cDAG in $\mathcal{F}$, the harder they are to approximate with a fixed cDAG (of same LoI).

Systematic generalization. Here we explore generalization guarantees that are of specific interest in CG – we aim to answer the following question: "If we approximate g, h, D with data (on known parts), how well can we handle unknown combinations of known parts?" Assuming that the cDAG function D and the encoder e are known, we can show that:

Proposition 2. *Consider $\mathcal{F}$ as in Definition 3, and let $f \in \mathcal{F}$ be the ground-truth and $f' \in \mathcal{F}$ be the learned function. Consider $X, X_1, X_2 \in \mathcal{I}^\star$, $X' \in \mathcal{E}(X)$, the set of exchangeable subsequences, and $X'_1, X'_2 \in \mathcal{I}^\star$ such that $W = [X_1 X X_2] \in \mathcal{X}$ and $W' = [X'_1 X' X'_2] \in \mathcal{X}$ are training examples of length L. Defining $f(Z) = h(g^{\otimes D(Z)}(e(Z))$ and $f'(Z) = h'(g'^{\otimes D(Z)}(e(Z))$ for any $Z \in \mathcal{X}$, and let*

$$|f(W) - f'(W)| \leq \epsilon, \quad |f(W') - f'(W')| \leq \epsilon,$$

for some small $\epsilon > 0$ (signifying "known parts"), we can quantify the accuracy of f' on a $V = [X_1 X' X_2]$ ("unknown combination of known parts") as

$$|f(V) - f'(V)| \leq \mathcal{O}(\epsilon/(1 - \beta_L)).$$

This shows that our definition of the compositional functions allows us to quantify the systematic generalization, highlighting that such functions provide some level of systematicity. However, note that, even if we only need to learn the g, h (with e, D given), our ability to systematically generalize depends on the LoI of any $f \in \mathcal{F}$.

Analysis of Existing Models. To validate the expressivity of the proposed definitions of compositional functions and classes, we believe it is important to position this definition with existing sequence models. We summarize the framing of the existing models into Definitions 1 and 3 in Table 1.

Model	DD	A-len	β_L	(k, q, m)
UniRNN	✗	✓	$\frac{L-1}{2L}$	$(2, 1, 1)$
BiRNN	✗	✓	$\frac{L-1}{4L}$	$(2, 2, 2)$
BinTree	✗	✓	$\frac{\log L}{L}$	$(2, 1, 1)$
Conv+Pool	†	✓	$\frac{\log L}{L}$	$(c_1 + c_2, c_1, m)$
FC+Att	✓	✗	$1 - \frac{K}{M(K+1)}$	$(K + 1, L, L)$

Table 1: Existing models framed as Def 3: DD – data-dependent cDAG. A-len – arbitrary length. 'UniRNN' – unidirectional recurrent comp. 'BiRNN' – bidirectional recurrent comp. 'BinTree' – balanced binary tree comp. 'Conv+Pool' – convolve over c_1, then pool over c_2 († Conv+Pool induces DD for max/min-pool, but not for avg/sum-pool). 'FC+Att' – M levels of fully connected with top-K attention in each.

References

[Andreas, 2018] Jacob Andreas. Measuring compositionality in representation learning. In *International Conference on Learning Representations*, 2018.

[Gordon *et al.*, 2019] Jonathan Gordon, David Lopez-Paz, Marco Baroni, and Diane Bouchacourt. Permutation equivariant models for compositional generalization in language. In *International Conference on Learning Representations*, 2019.

[Hupkes *et al.*, 2020] Dieuwke Hupkes, Verna Dankers, Mathijs Mul, and Elia Bruni. Compositionality decomposed: how do neural networks generalise? *Journal of Artificial Intelligence Research*, 67:757–795, 2020.

[Keysers *et al.*, 2019] Daniel Keysers, Nathanael Schärli, Nathan Scales, Hylke Buisman, Daniel Furrer, Sergii Kashubin, Nikola Momchev, Danila Sinopalnikov, Lukasz Stafiniak, Tibor Tihon, et al. Measuring compositional generalization: A comprehensive method on realistic data. In *International Conference on Learning Representations*, 2019.

[Kim and Linzen, 2020] Najoung Kim and Tal Linzen. Cogs: A compositional generalization challenge based on semantic interpretation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9087–9105, 2020.

[Klinger *et al.*, 2020] Tim Klinger, Dhaval Adjodah, Vincent Marois, Josh Joseph, Matthew Riemer, Alex 'Sandy' Pentland, and Murray Campbell. A study of compositional generalization in neural models, 2020.

[Lake and Baroni, 2018] Brenden Lake and Marco Baroni. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *International Conference on Machine Learning*, pages 2873–2882. PMLR, 2018.

[Li *et al.*, 2019] Yuanpeng Li, Liang Zhao, Jianyu Wang, and Joel 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)*, pages 4284–4293, 2019.

[Liu *et al.*, 2020] Qian Liu, Shengnan An, Jian-Guang Lou, Bei Chen, Zeqi Lin, Yan Gao, Bin Zhou, Nanning Zheng, and Dongmei Zhang. Compositional generalization by learning analytical expressions. *Advances in Neural Information Processing Systems*, 33, 2020.

[Liu *et al.*, 2021] Chenyao Liu, Shengnan An, Zeqi Lin, Qian Liu, Bei Chen, Jian-Guang Lou, Lijie Wen, Nanning Zheng, and Dongmei Zhang. Learning algebraic recombination for compositional generalization. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1129–1144, 2021.

[Nye *et al.*, 2020] Maxwell Nye, Armando Solar-Lezama, Josh Tenenbaum, and Brenden M Lake. Learning compositional rules via neural program synthesis. *Advances in Neural Information Processing Systems*, 33:10832–10842, 2020.

[Pagin and Westerståhl, 2010a] Peter Pagin and Dag Westerståhl. Compositionality I: Definitions and variants. *Philosophy Compass*, 5(3):250–264, 2010.

[Pagin and Westerståhl, 2010b] Peter Pagin and Dag Westerståhl. Compositionality II: Arguments and problems. *Philosophy Compass*, 5(3):265–282, 2010.

[Partee and others, 1995] Barbara Partee et al. Lexical semantics and compositionality. *An invitation to cognitive science: Language*, 1:311–360, 1995.

[Russin *et al.*, 2019] Jake Russin, Jason Jo, Randall C O'Reilly, and Yoshua Bengio. Compositional generalization in a deep seq2seq model by separating syntax and semantics. *arXiv preprint arXiv:1904.09708*, 2019.