# Compositional Structured Explanation Generation with Dynamic Modularized Reasoning

**Xiyan Fu**
Dept. of Computational Linguistics
Heidelberg University
fu@cl.uni-heidelberg.de

**Anette Frank**
Dept. of Computational Linguistics
Heidelberg University
frank@cl.uni-heidelberg.de

## Abstract

In this work, we propose a new task, *compositional structured explanation generation* (CSEG), to facilitate research on compositional generalization in reasoning. Despite the success of language models in solving reasoning tasks, their compositional generalization capabilities are under-researched. Our new CSEG task tests a model's ability to generalize from generating entailment trees with a limited number of inference steps – to *more steps*, focusing on the length and shapes of entailment trees. CSEG is challenging in requiring both *reasoning* and *compositional generalization* abilities, and by being framed as a *generation* task. Besides the CSEG task, we propose a new *dynamic modularized* reasoning model, MORSE, that factorizes the inference process into modules, where each module represents a functional unit. We adopt *modularized self-attention* to dynamically select and route inputs to dedicated heads, which specializes them to specific functions. Using CSEG, we compare MORSE to models from prior work. Our analyses show that the task is challenging, but that the dynamic reasoning modules of MORSE are effective, showing competitive compositional generalization abilities in a generation setting.[1]

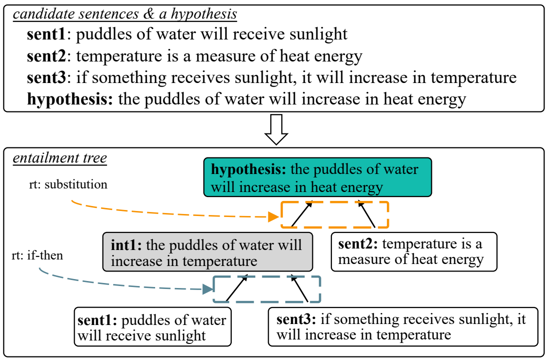


Figure 1: Structured explanation generation: generate an entailment tree including intermediate nodes (grey) for a hypothesis (green) and given candidate sentences. Each reasoning step (sent1 & sent3 $\rightarrow$ int1) is independent and belongs to one of six reasoning types (rt).

## 1 Introduction

Large-scale language models (Raffel et al., 2019; Chung et al., 2022; Touvron et al., 2023) have shown remarkable performance on reasoning tasks, such as reading comprehension (Rajpurkar et al., 2018), natural language inference (Williams et al., 2018), story generation (Mostafazadeh et al., 2016), etc. However, Russin et al. (2020); Mitchell (2021); Yuan et al. (2023) argued that these models lack human-like reasoning capabilities.

Humans excel in *compositional generalization* (Hupkes et al., 2020), a capacity to combine an inventory of known constituents to predict larger compounds, during reasoning. For example, humans who understand calculation constituents, e.g., subtraction *sub(X,Y)* and mixed addition-subtraction operations *sub(X, add(Y, Z))*, can solve larger compounds, e.g., *sub(W, sub(X, add(Y, Z)))*.

Various studies (Hudson and Manning, 2019; Goodwin et al., 2020; Yanaka et al., 2020; Liu et al., 2022) have explored compositional generalization abilities in reasoning tasks. But, these works focus on compositionality units manifesting on the word level and involving specific linguistic phenomena, and neglect inferential processes holding between sentences. But *sentence*-level composition can enhance the capacity of models to execute complex contextual reasoning.

To fill this gap, we propose a new task, *compositional structured explanation generation, CSEG*. CSEG is a new setting built on SEG (Dalvi et al., 2021), a task for models to generate a multi-step entailment tree – given a hypothesis and candidate sentences. The tree indicates how the hypothesis follows from the text. Fig. 1 shows an example. Each step (e.g., sent1 & sent3 $\rightarrow$ int1) represents a multi-premise textual inference (Lai et al.,

[1]https://github.com/xiyan524/MORSE



*Proceedings of the 13th Joint Conference on Lexical and Computational Semantics (*SEM 2024)*, pages 385 - 401
June 20-21, 2024 ©2024 Association for Computational Linguistics

2017), belonging to one of six reasoning types, such as if-then (it) and substitution (subs) (see Appendix A.1 and A.3 for examples). We consider each reasoning type as a *constituent unit*. To test compositional generalization in reasoning, our new task CSEG requires models to generalize from entailment trees with a *limited* number of reasoning steps to trees involving *more* steps. For example, a model is expected to generate a larger compound (entailment tree) with more reasoning steps, e.g., $c_3$: *subs(subs(it(*$p_1$, $p_2$*)* $\rightarrow h_1$, $p_3$*)* $\rightarrow h_2$, $p_4$*)* $\rightarrow h_3$, by combining known constituents $c_1$: *subs(it(*$p_1$, $p_2$*)* $\rightarrow h_1$, $p_3$*)* $\rightarrow h_2$ and $c_2$: *subs(*$p_1$, $p_2$*)* $\rightarrow h$, where $c_1$ replaces $p_1$ in $c_2$). Here, compositionality units, i.e., reasoning types, operate on the *sentence* level and involve reasoning components.

Our new CSEG task requires: i) *reasoning* capabilities, to infer new conclusions from existing information; and ii) *compositional generalization* capability, to generalize to unseen compounds using previously learned constituents. Recent efforts (Dalvi et al., 2021; Saha et al., 2020; Tafjord et al., 2021) aimed to improve reasoning abilities, while ignoring the compositional generalization capacity. Existing symbolic-based approaches (Martínez-Gómez et al., 2017; Gupta et al., 2019; Le et al., 2022) used multiple modules that each perform unique types of reasoning, endowing models with strong compositionality. But they rely on pre-defined reasoning rules and need training data for each pre-defined module. Inspired by this, we propose a *dynamically modularized* reasoning model MORSE. Our model simulates the symbolic process by specializing Transformer self-attention heads to what we call *dynamic modules*. We design a modularized self-attention mechanism that dynamically selects and routes inputs to dedicated modularized heads, specializing them to specific functions. The dynamics embodied in MORSE through its self-assembling modules makes it applicable to multiple datasets without pre-defined knowledge and extend to novel inference types.

Our main contributions are:

i) We propose a new compositional structured explanation *generation* task, which aims to explore *compositional generalization* capabilities in reasoning. It requires models to generalize from entailment trees with a *limited* number of inference steps to *more* steps.

ii) We design a novel dynamically modularized reasoning model that specializes transformer heads to specific functions, by *dynamically* selecting related inputs to dedicated heads.

iii) Experiments on two benchmarks targeting generalization over proof lengths and shapes demonstrate MORSE's advanced compositional generalization abilities.

## 2 Related Work

**Generalization in Reasoning** Despite the success of language models in solving reasoning tasks, their generalization abilities have attracted attention, e.g., length generalization (Clark et al., 2020; Wu et al., 2021; Anil et al., 2022), compositional generalization (Liu et al., 2022), domain generalization (Niu et al., 2023), etc. In this work, we explore compositional generalization in reasoning.

Compositional generalization has been researched for decades (Fodor and Pylyshyn, 1988; Marcus, 2003; Lake and Baroni, 2018), including two significant properties: productivity and systematicity (Hupkes et al., 2020). Among these, *productivity* is similar to length generalization, in that both evaluate generalization to deeper reasoning chains. But for evaluating productivity, primitive units needed for solving deeper samples must have been learned during training. In contrast to the related length-generalization work of Clark et al. (2020), our CSEG task aims to evaluate productivity in a *structured* compositional generalization reasoning task. We therefore guarantee that primitive units (rule types) needed for solving deeper samples have been learned in training. Importantly, we frame CSEG as a generation task, which unlike classification settings as in Clark et al. (2020), makes it harder for models to exploit shortcuts.

Recently, there has been renewed interest in exploring compositional generalization in reasoning tasks. Johnson et al. (2017); Hudson and Manning (2019); Bogin et al. (2021); Gao et al. (2022) proposed challenging compositional tasks in visual QA. Liu et al. (2022) designed compositional questions for QA and found even the strongest model struggled with these challenging questions. Other works probed the compositional abilities of models in natural language inference (Geiger et al., 2020; Goodwin et al., 2020; Yanaka et al., 2020, 2021; Fu and Frank, 2023, 2024), focusing on specific linguistic phenomena, such as quantifiers, negation, or predicate replacements. I.e., they investigate compositionality in phenomena manifesting at the word level, in contrast to inferential processes holding between sentences.

To fill this gap, we examine compositional generalization in a multi-step entailment tree generation task, where different inference rules need to be composed. Concurrent work (Saparov et al., 2023) also concentrates on sentence-level compositionality in reasoning, but is limited in using a synthetic dataset. In comparison, we employ both natural language and synthetic data, and introduce a new model, with potential for further improvement, that can serve as a strong baseline for the task.

**Neural-Symbolic and Neural Methods** Prior works show that symbolic approaches (Angeli and Manning, 2014; Mineshima et al., 2015; Martínez-Gómez et al., 2017) that adopt pre-defined inference rules to establish derivations through iterative reasoning, endow models with strong compositionality. But being dependent on pre-defined rules, the models are limited to well-defined tasks. Recently, Yi et al. (2018); Yin et al. (2018); Li et al. (2020); Jiang et al. (2021) used neural networks to map raw signals to symbolic representations and subsequently performed symbolic reasoning to make predictions. As symbolic reasoning is brittle, novel works based on Neural Modular Networks (NMN) (Andreas et al., 2016; Hu et al., 2017) combine individual neural modules endowed with *specialized* reasoning capabilities. E.g., Jiang and Bansal (2019); Gupta et al. (2019) designed various modules in an NMN to perform unique types of reasoning in end-to-end manner. Similarly, Khot et al. (2021, 2023) proposed a Text Module Network for complex reasoning tasks, where each module is an existing QA system. However, all these approaches require prior knowledge and rely on brittle symbolic transfer, to subsequently deploy pre-defined modules for each sub-task, and well-designed modules require substantial extra training data. Finally, symbolic reasoning methods are typically driven by weak supervision, given the lack of intermediate labels. This can result in error accumulation and time-consuming learning. To address these challenges, we propose a model with *dynamic modules* that make specific module functions more independent from prior knowledge, to endow models with greater flexibility when handling new tasks.

Our work may seem related to Mixture-of-Expert (MoE) models (Jacobs et al., 1991; Lepikhin et al., 2021; Li et al., 2023) that aim to decompose tasks by composing separate networks, each of which is trained to handle a subset of a complete set of training cases. By contrast, MORSE focuses on decomposition and combining primitive units *in individual samples*. In addition, it uses multiple heads of the existing Transformer cell, without inducing extra training parameters (such as FNN layers of MoE) – which has higher efficiency.

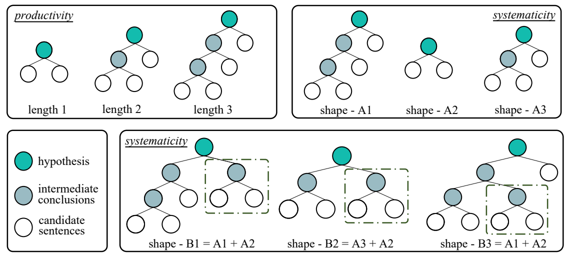


Figure 2: Entailment trees including different lengths and shapes for compositional generalization testing.

## 3 Task Setup

**Background** The Structured Explanation Generation (SEG) task (Dalvi et al., 2021) requires a system to generate a multi-step entailment tree given a hypothesis and candidate sentences. The tree serves as a structured explanation of how presented evidences leads to a conclusion.

Input to the task are i) a hypothesis $H$, a declarative statement and ii) a set $S$ of candidate sentences that express relevant knowledge needed to infer $H$. Outputs are valid entailment trees with intermediate conclusions not contained in $S$ (Fig. 1). The entailment trees are encoded as linear sequences that can be generated by a generative model. The tree depicted in Fig. 1 would be represented as:

*sent1 & sent2 $\rightarrow$ int1: the puddles of water will increase in temperature; sent2 & int1 $\rightarrow$ hypot*

Leaves $sent_i$ are sentences from the candidate set $S$, and *hypot* is the tree's root, given by the hypothesis $H$. $int_j$ are inferred intermediate conclusions that provide the basis for further reasoning steps.

**Compositional Generalization Testing** To examine compositional reasoning capabilities, we partition our benchmark datasets along two generalization properties: *productivity* and *systematicity*.

**Productivity–Length** evaluates systems on longer proof lengths than they have been trained on, where both train and test sequences are composed of identical primitives. Hence, we rearrange the data by proof length, i.e., number of intermediate nodes in each tree (including hypothesis node). We partition the data into: i) primitive entailment trees of length one or two; ii) compositional entailment trees of length three.[2] Fig. 2 shows examples.

**Systematicity–Shape** examines the capability of (re)combining known constituents to a larger compound. Hence, we rearrange the dataset by tree shapes. To select appropriate data, we proceed as follows: we i) limit the inference steps of each tree to four – given that larger steps present an unsolved challenge for existing neural models (Table A.3, Dalvi et al. (2021)); ii) extract the tree shapes from candidate data; iii) find there exist only six different shapes, depicted as *shape-** in Fig. 2 (details in Appendix A.2); iv) select, among six possible shapes, simple structures (Shape-A*) as primitives, and more complex (compositional) ones (Shape-B*) as compositions for generalization testing. We guarantee that compositional shapes are built from primitive shapes: B1=A1+A2, B2=A3+A2, B3=A1+A2. In Figure 2, we use dashed squares to single out one primitive shape for each compositional shape.

## 4 MORSE: Dynamic Modularized Reasoning Model

We introduce our Dynamic Modularized Reasoning Model MORSE that generates compositional structured explanations. MORSE contains: i) an encoder consisting of original and modularized transformer blocks to perform reasoning; ii) a decoder using original transformer blocks to generate the entailment tree structure. See the overview in Fig. 3.

### 4.1 Module-enhanced encoder

We concatenate candidate sentences $S$ and the hypothesis $H$ into an input sequence. For each sentence in $S$, we add a prefix $sent*$ following Dalvi et al. (2021). Thus the example in Fig.1 is represented as a sequence of length $n$: 'sent1: puddles of water will receive sunlight; sent2: temperature is a ...; ...; hypothesis: the puddles of water will increase in heat energy'. For each token $x_i$, we adopt an embedding layer to generate its representation $e_i$, i.e., a summation of token embedding, position embedding and segment embedding. An encoder subsequently encodes input representations.

Fig. 3.A shows that MORSE's encoder consists of *Transformer* blocks for lower layers and *Modularized Transformer* blocks for higher layers: i) Transformer blocks allow the model to focus on the representation of words themselves (Raganato and Tiedemann, 2018; Jawahar et al., 2019); ii) Modularized Transformer blocks perform modularized reasoning, where each module is encouraged to learn a different inference function.

**Transformer** All Transformer blocks consist of two sub-layers: a multi-head attention layer and a fully connected feed-forward network. Each sub-layer is followed by layer normalization (Ba et al., 2016) and a residual connection (He et al., 2016). In the multi-head attention sub-layer, sequential inputs are projected to different representation subspaces (different heads) in parallel; the layer then performs self-attention (Vaswani et al., 2017) in each head. The heads' output values are concatenated and again projected, resulting in final values.

In MORSE, we adopt $p$ Transformer blocks in lower layers, aiming to capture the representation of words in their syntactic context. Given token embeddings $e_1$, ..., $e_n$ of a sequential input of length $n$, we use $p$ Transformer blocks to encode them and generate corresponding hidden states $s_1^p$, ..., $s_n^p$.

**Modularized Transformer** We construct a Modularized Transformer block based on the Transformer. The difference is that we factorize the encoding process, by modularizing the Transformer so that each module can be tailored to a specific function. We implement this design by using Transformer 'heads'. The process of *modularization* is illustrated in Fig. 3 B.1: the modularized Transformer block contains a modularized attention layer, which consists of multiple specialized heads $h_i$. E.g., $h_0$ to $h_5$ are modularized heads that may express different inference functions. The remaining heads $h_{6,7}$ work as usual, offering space to model general knowledge not covered by the modularized heads. With such modularization, we expect that each module will specialize for specific responsibilities, further endowing MORSE with more flexibility to perform different inference functions during reasoning.

To allow a modularized head $h_i$ to specialize for specific functions, we construct dynamic masks $m_i \in [0, 1]^n$ to select sequential inputs of similar kinds to pass through $h_i$. Specifically, we define several vectors of trainable parameters for each module as a latent representation of the module's function, e.g., $rep_{h_i} \in \mathbb{R}^d$ for $h_i$. Simultaneously, we adopt a linear projection on candidate input hidden states $s_1$, ..., $s_n$ to derive their functional

[2] We only test length three here, given the significant performance challenge shown by experiments. However, our setting is a living benchmark, which can be easily extended by future research.

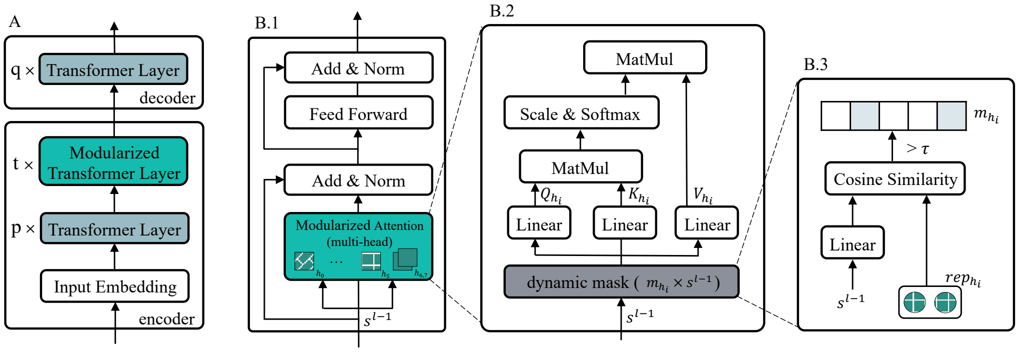


Figure 3: (A) MORSE for entailment tree generation. (B) A series of detailed illustrations of the Modularized Transformer layer. (B.1) Our novel *modularized* multi-head self-attention block. Each head may serve as a module, executing a specific function. (B.2) Computations for a single attention head with dynamic mask $m_{h_i}$. Self-attention is extended with a dynamic mask to filter out irrelevant input for a module. (B.3) Constructing dynamic mask $m_{h_i}$ using head function representation $rep_{h_i}$ and input hidden states.

representations $f_1$, ..., $f_n \in \mathbb{R}^d$. Then, we use cosine similarity $cos$ over the input's functional representations $f_j$ and the head's representation $rep_{h_i}$ to calculate a matching coefficient. If it exceeds a threshold $\tau$, MORSE is able to decide if an input word $x_j$ is allowed to join the module $h_i$. The mask calculation is shown below:

$$m_i^j = \begin{cases} e^{1-cos(rep_{h_i}, f_j)}, & cos(rep_{h_i}, f_j) > \tau \\ 0, & else \end{cases} \tag{1}$$

where the threshold $\tau$ is a fixed hyper-parameter. To avoid the vanishing gradient problem, we use $e^{1-cos(*)}$ to represent the mask for a selected word. For unselected words, we ignore their gradient. In this way, we can generate masks $m_i$ for each module $h_i$ dynamically, given sequential inputs and different module objectives.

We further adopt the generated mask $m_i$ for a module $h_i$ in Modularized Self-Attention to filter out unrelated inputs. Fig. 3 B.2 shows the process: we multiply the mask $m_i$ with input hidden states from the previous layer $s^{l-1}$, where hidden states of unrelated words are set to zero. Then, we generate the query $Q_{h_i}$, key $K_{h_i}$, and value $V_{h_i}$ matrices for self-attention by different linear projections based on filtered inputs:

$$\begin{aligned} Q_i, K_i, V_i &= \tilde{s}^{l-1}W_i^Q, \tilde{s}^{l-1}W_i^K, \tilde{s}^{l-1}W_i^V \\ \tilde{s}^{l-1} &= m_{h_i} \times s^{l-1} \end{aligned} \tag{2}$$

where $W_i^Q, W_i^K, W_i^V \in \mathbb{R}^{d \times d/k}$ are training parameters, $d$ is the hidden state dimension and $k$ is the number of heads. We then adopt scaled dot-product attention to perform self-attention:

$$a_i = softmax(\frac{Q_i K_i^T}{\sqrt{d_k}})V_i \tag{3}$$

We adopt $t$ Modularized Transformer blocks in deep layers, aiming to perform modularized reasoning. Given input hidden states $s_1^p, ..., s_n^p$ from lower Transformer blocks, the Modularized blocks generate modularized hidden states $s_1^t, ..., s_n^t$.

### 4.2 Decoder and training

We use a decoder consisting of Transformer blocks to generate the entailment tree structure and intermediate conclusions. The entailment tree is linearized from leaves to the root. For example, the tree in Fig. 1 is represented as *"sent1 & sent2 → int1: the puddles of water will increase in temperature; sent3 & int1 → hypo."* The output sequence generation process is defined as:

$$\begin{aligned} s^l &= block(s^{l-1}, enc\_state), \quad l\epsilon[1, q] \\ p(y_k|y_{<k}) &= softmax(s_k^N W^T) \end{aligned} \tag{4}$$

where $s^l$ is the $l_{th}$ layer computed through Transformer blocks, $W^T$ is the training parameter and $k$ is the decoding step number. We deploy supervised learning with ground truth by minimizing the objective in (5), where M is the maximum length of the generated entailment tree, and $H$ and $S$ are hypothesis and candidate sentences, respectively.

$$L = -\sum_{k=1}^{M} logp(y_k|y_{<k}, H, S) \tag{5}$$

## 5 Experiments Setup

### 5.1 Datasets

In this section, we prepare the compositional data from EntailmentBank (EntB) and DBpedia (DBP) for the CSEG task.

**EntailmentBank (EntB)** by Dalvi et al. (2021) contains multiple-choice questions and candidate sentences from the grad-school level science datasets ARC (Clark et al., 2018) and WorldTree (Jansen et al., 2018; Xie et al., 2020). 1,840 entailment trees each show how a hypothesis is entailed by a small number of relevant sentences. Each step in the tree represents an entailment, i.e., the conclusion expressed in each intermediate node follows from the content of its immediate children. The individual entailment steps instantiate six common reasoning types (details in A.1) [3]. EntB contains three tasks. We focus on Task1, with only correct inputs in $S$, as we focus on generalization testing.

**DBpedia** by Saeed et al. (2021) is a synthetic dataset that was re-generated from the **RuleBert** (Saeed et al., 2021) dataset[4]. We extracted six distinct logic rules mined from the DBpedia knowledge graph and instantiated examples with a varying number of variables following 'Chaining of Rule Execution' in RuleBert (cf. A.3). The reasoning chain provides a structured explanation: each intermediate node is a conclusion inferred from immediate children using a logic inference rule.

**Compositional Generalization Testing Data** To construct the dataset for systematicity and productivity testing in reasoning explanation generation, we rearrange the partitions of the above benchmarks to focus on *length* and *shape* of entailment trees following §3 (see A.4 for details). We construct i) EntB(ank)-L(ength) and DBP-L(ength) based on entailment tree length; and ii) EntB-Sh(ape) based on entailment tree shape. Since DBpedia does not contain more complex tree shapes, it is ignored in the shape test. For data statistics of the created splits for length and shape testing, see Appendix A.5.

### 5.2 Experiment Details

**Settings** Zero-shot compositional generalization is highly non-trivial due to the long generated texts of the compositional samples.[5]. We therefore consider a flexible learning scenario following Bogin et al. (2021); Yin et al. (2021). Specifically, we trained a model (both baselines and MORSE) with primitives, and further fine-tuned the model with a handful of compositional examples to familiarize itself with a complicated space. For data statistics details see Appendix A.5. To provide a comprehensive analysis for future work, we also conducted conventional zero-shot tests, where we trained a model with primitives and tested on compositions directly.

**Model** MORSE is built on T5-Small/-Large with six/ twelve layers (cf. Dalvi et al. (2021)). For each version, we use, for the lower 30% of layers (i.e., two/four layers), the original Transformer blocks, to derive hidden representations of the input words. The threshold $\tau$ for dynamic mask construction we set to 0.1. All models were evaluated on three runs. For further details see Appendix B.

### 5.3 Baselines

We choose three prior systems for structural explanation generation as baselines, and report comparative results for our new system **MORSE**.[6]

**EntailmentWriter** (Dalvi et al., 2021) is a T5-based seq-to-seq model that generates a structured explanation (tree) directly. It provides baseline results on EntailmentBank for generating entailment trees for answers to science questions.

**PROVER** (Saha et al., 2020) jointly answers binary questions over rule-bases and generates the corresponding proofs. The model learns to predict edges corresponding to proof graphs using multiple global constraints. Since PROVER focuses on edge prediction, we only evaluate the tree structure.

**ProofWriter-Iterative** (Tafjord et al., 2021) iteratively generates 1-step conclusions and proofs, adds intermediate conclusions to the context and assembles a final proof chain from 1-step fragments.

### 5.4 Automatic Evaluation Metrics

We adopt the evaluation metrics proposed by Dalvi et al. (2021) for the structured explanation generation task. Evaluation is addressed in two steps:

[3] The number of reasoning types is a flexible parameter depending on the dataset.

[4] https://github.com/MhmdSaiid/RuleBert

[5] The difficulty is primarily due to the decoder trained by maximum likelihood, which relies heavily on the distributional characteristics of the dataset and assigns low probabilities to unseen combinations in test (Holtzman et al., 2020)

[6] For reference, the results obtained by MORSE on the original structured explanation generation task SEG are reported in Appendix D.

| Models | EntailmentBank-Length (EntB-L) | | | | | | DBpedia-Length (DBP-L) | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Leaves | | Steps | | Intermediates | | Leaves | | Steps | | Intermediates | |
| | F1 | AllCorrect | F1 | AllCorrect | F1 | AllCorrect | F1 | AllCorrect | F1 | AllCorrect | F1 | AllCorrect |
| ProofWriter-It. | 91.86(0.08) | 84.55(0.78) | 35.97(2.37) | 18.81(2.76) | 42.93(1.23) | 11.88(2.14) | 90.66(0.18) | 93.09(0.72) | 76.49(0.86) | 75.44(1.04) | 85.92(1.92) | 76.73(2.24) |
| PROVER | - | - | 39.27(2.65) | 24.75(3.24) | - | - | - | - | 79.88(0.98) | 76.98(1.37) | - | - |
| EntWriter (T5-Small) | 99.78(0.12) | 98.02(1.06) | 40.59(2.97) | 29.70(2.92) | 48.24(1.12) | 22.77(2.25) | 99.92(0.15) | 99.49(0.67) | 82.01(1.21) | 79.28(1.52) | 87.05(2.23) | 78.26(2.37) |
| MORSE (T5-Small) | **99.89**(0.08) | **99.01**(0.62) | **44.22**(2.14) | **32.67**(2.32) | **50.66**(0.68) | **25.74**(1.92) | **99.96**(0.27) | **99.74**(0.84) | **82.27**(0.16) | **80.31**(0.18) | **87.72**(1.82) | **79.80**(1.87) |
| EntWriter (T5-Large) | 99.78(0.11) | 98.02(0.99) | 52.80(3.35) | 40.92(3.18) | 56.62(1.06) | 36.63(2.40) | 99.36(0.13) | 95.52(0.91) | 82.49(1.09) | 80.11(1.43) | 88.98(2.16) | 83.89(2.15) |
| MORSE (T5-Large) | **99.82**(0.06) | **98.68**(0.57) | **53.31**(2.26) | **42.57**(2.62) | **57.78**(0.81) | **37.29**(2.06) | **99.53**(0.11) | **96.68**(0.73) | **86.79**(0.12) | **83.76**(0.18) | **92.62**(1.70) | **86.70**(1.97) |
| EntWriter-0-shot (T5-L) | 97.06(0.66) | 85.73(1.61) | 18.44(1.18) | - | 24.21(2.22) | - | **90.09**(0.42) | 29.27(0.2) | 16.94(1.68) | - | 32.43(0.50) | - |
| MORSE-0-shot (T5-L) | **97.89**(0.74) | **86.83**(1.52) | **19.14**(0.89) | - | **25.42**(1.49) | - | 89.82(0.32) | **30.05**(0.90) | **18.41**(1.09) | - | **33.45**(0.22) | - |

Table 1: Results on EntailmentBank-L(ength) and DBpedia-L(ength) for compositional generalization evaluation. All modules are evaluated with 3 rounds, we show mean accuracy (std).

1) **Alignment** Exact matching between a predicted ($T_{pred}$) and a human-labeled ($T_{gold}$) entailment tree ignores the different organizations among tree nodes and leads to an inaccurate evaluation score. To admit semantic variation, all $T_{pred}$ nodes are (greedily) aligned to nodes in $T_{gold}$ using the sent* labels of leaf nodes, followed by Jaccard similarity calculation for intermediate nodes.

2) **Score** Once aligned, three metrics measure the degree of similarity of $T_{pred}$ and $T_{gold}$: (a) *Leaves* evaluates if the generated tree selects the correct leaf sentences from the candidate set $S$. (b) *Steps* assesses if the individual entailment steps in the tree are structurally correct. This is the case if for a pair of aligned intermediate nodes, both children have identical labels (sent* or int*) in $T_{pred}$ and $T_{gold}$. (c) *Intermediates* judges if all generated intermediate conclusions are correct. BLEURT (Sellam et al., 2020) with the threshold 0.28 [7] is applied for intermediate conclusion evaluation. For each metric, we compute an F1 score, and an 'AllCorrect' score for exact tree matching (F1=1).

## 6 Results

### 6.1 Overall Results

**Results on Length Composition** Table 1 displays the results of MORSE using the small vs. large T5 model as backbone, on the EntB-L and DBP-L datasets. Note that PROVER (Saha et al., 2020), EntailmentWriter (EntWriter) (Dalvi et al., 2021) and MORSE generate the complete proof chain from the input candidate set in one go, while ProofWriter-Iterative (PW-Iterative) (Tafjord et al., 2021) generates one-step implications iteratively. We find that on both datasets, and for both T5 model sizes, MORSE achieves superior results compared to all baselines, especially on 'Steps' (structural correctness) and 'Intermediates' (intermediate conclusions). 'Leaves' is not a challenge

[7]The threshold is determined following (Dalvi et al., 2021).

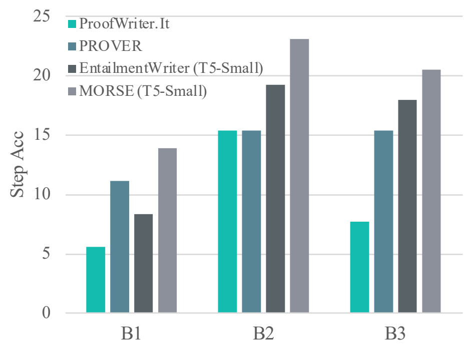


Figure 4: Results on EntB-Sh, testing for compositional generalization, i.e., systematicity.

in our Task1 setup, but even here, MORSE outperforms, being able to integrate almost all inputs. The comparison with the most competitive system EntWriter, in equivalent T5 model sizes, still shows superior performance of MORSE with both model sizes. We conclude that the advance of MORSE is not restricted to small models, but persists with models hosting richer knowledge. Compared to DBP-L, the advance of MORSE over the other baselines is stronger on EntB-L (e.g., +2.97 vs. +1.03 for 'Steps Acc'). This is explained by the synthetic (template-based) nature of the DBP-L dataset, which shows little linguistic variety.

To provide a comprehensive evaluation of the proposed new setting for future research, we further challenge MORSE by exposing it to a *zero-shot test* for length composition. Here, models trained only for trees up to depth two will directly receive inputs for proof trees of length three. We mainly compare with the most competitive system, EntWriter. In this evaluation, we ignore the 'AllCorrect' scores for 'Steps' and 'Intermediate' outputs, given the difficulty of these generation tasks in low training regimes. The last two lines in Table 1 show the results. MORSE achieves superior performance (at least +1 point improvement for zero-shot) for most evaluation categories, or else comparable results

| Models | Steps | | Intermediates | |
|---|---|---|---|---|
| | F1 | Acc | F1 | Acc |
| MORSE (T5-Small) | 44.22 | 32.67 | 50.66 | 25.74 |
| freeze rep_embed | 43.57 | 31.68 (-0.99) | 50.66 | 25.74 (-0) |
| + module | 41.58 | 29.70 (-2.97) | 49.13 | 23.76 (-1.98) |
| + masking | 38.28 | 25.74 (-6.93) | 46.62 | 20.79 (-4.95) |

Table 2: Ablation of MORSE components, freeze: **rep_embed**: the representation of module $rep_i$; **module**: parameters in specialized module; **masking**: dynamic mask in Fig. 3. d. Brackets: decrease in accuracy.

(F1 for 'Leaves'). We conclude that our model MORSE [8] outperforms other baselines in both zero-shot and fine-tuning scenarios.

**Results on Shape Composition** Fig. 4 displays the results for generalization testing on shapes.[9] MORSE clearly surpasses the step accuracy of all other baselines for all tested shape configurations. Note that shape B1 is most difficult for all systems. Entailment trees are linearized in bottom-up order. While compositions in shape B2 and B3 happen at the lowest tree level, composition in B1 happens at a higher tree level, combining trees of unequal depths. We hypothesize that combining trees of unequal lengths at higher levels makes the task more challenging compared to lower levels, given that composition at higher levels requires a more precise representation of previous reasoning steps (see Appendix C for more details).

## 6.2 Analysis of Modularization

**Ablation Study** To gain more insight into the impact of specific components of MORSE on generalization, we run an ablation study on EntB-L during *fine-tuning*. We first freeze all module representations $rep_{h_i}$ (*rep_embed*). Further, we freeze parameters in each specialized module (+*module*) (cf. Fig. 3.B.2). By freezing these parameters, we aim to preserve the function of different modules and expect a comparative performance by reusing learned functions. In the third ablation, we freeze the parameters of the dynamic mask process +*masking* (cf. Fig. 3.B.3), which affects the dynamic mask of inputs to different modules. Results in Table 2 indicate that the first two settings do not affect results much, which suggests that each module has roughly learned its specialized functions. But +*mask* incurs large drops, which indicates that masking is significant for the model to adapt to novel configurations. We hypothesize that for generalizing to longer proofs, mask generation helps to connect existing modules.

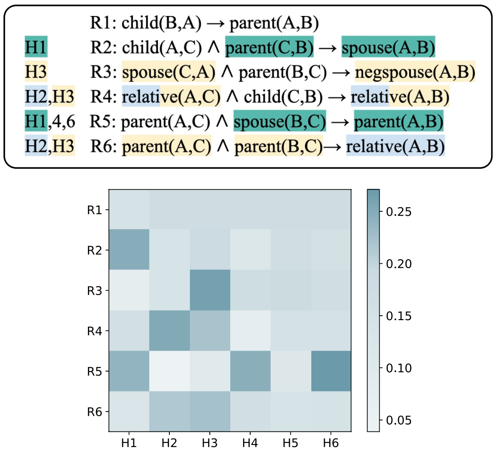


Figure 5: Correlations between reasoning rules R1-6 and module heads H1-6.

**Correlation Analysis** To further explore the effects of modularization in MORSE, we conduct an experiment on DBP-L by *masking individual heads* only in testing. We select samples that: i) contain three reasoning steps, ii) made correct predictions for the first two reasoning steps, but iii) predict the $3^{rd}$ step incorrectly in case a certain head is removed (see A.6 for details). This ensures that the reasoning rule for the $3^{rd}$ step is affected by a specific removed head. We count samples that are affected by removing head $j$ for each rule $R_i$, denoted as $n_j^{R_i}$. In case a model has $T$ heads, we normalize affected sample counts of $R_i$ across all heads, i.e., $n_j^{R_i} / \sum_{j=1}^{T} n_j^{R_i}$. This allows us to align heads and rules as shown in Fig. 5.

The heatmap shows the correlations between rules and heads, where R2-H1, R3-H3, R4-H2/H3, R5-H1/H4/H6 and R6-H2/H3 stand out. In the upper part of Fig. 5 we list all inference rules from DBP-L, aligned with the heads they are strongly correlated with, according to the heatmap. We find that heads are correlated with some rules roughly: 1) H4 and H6 are quite similar, and both prefer R5. 2) H1 prefers R2, but is distracted by R5. This is likely because R2 and R5 are similar by changing 'parent' to 'child' between A and C. 3) H2 prefers R4 and R6, which both use the predicate 'relative' and share the same relation by changing 'parent' to 'child' between B and C. 4) H3 prefers R3, but is distracted by R4 and R6. A plausible reason could be configurations of R3, R4 and R6 are similar

[8]Experiments on more powerful backbones are provided in Appendix F.

[9]Having seen linear behaviour of different model sizes in Table 1, we further on use T5-Small versions of MORSE and EntWriter, unless we explicitly say otherwise.

as they share similar predicates ('spouse' in R3, 'relative' in R4 and 'parent' in R6).

## 7 Conclusion

We present a new setup for explanation generation to facilitate compositional generalization in reasoning research. Inspired by highly compositional symbolic systems, we propose a novel modularized reasoning model MORSE that factorizes reasoning processes into a combination of *dynamically* specializing modules. Our results establish MORSE as a strong baseline for the task, using two benchmarks. A future direction is to learn how to initialize more modules on demand.

## 8 Limitations

The dynamic modularized reasoning model MORSE in its current state is limited by assuming a pre-defined number of modules, for reasoning in various scenarios. The number of modules in MORSE interacts with the ability of the model when modularizing a given number of potential logic rules in a dataset or task. A given available number of functional units can simplify the reasoning process, enabling the model to focus on module re-use similar to how a symbolic system does, instead of distracting from confirming module function granularity.

## 9 Acknowledgments

We are grateful to anonymous meta-reviewers for their valuable comments that have helped to improve this paper. We also thank Wei Liu and Wan Le for their valuable feedback on drafts of this paper. This work has been supported through a scholarship provided by the Heidelberg Institute for Theoretical Studies gGmbH.

## References

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. 2016. Neural module networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 39–48.

Gabor Angeli and Christopher D. Manning. 2014. NaturalLI: Natural logic inference for common sense reasoning. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 534–545, Doha, Qatar. Association for Computational Linguistics.

Cem Anil, Yuhuai Wu, Anders Andreassen, Aitor Lewkowycz, Vedant Misra, Vinay Ramasesh, Ambrose Slone, Guy Gur-Ari, Ethan Dyer, and Behnam Neyshabur. 2022. Exploring length generalization in large language models. *Advances in Neural Information Processing Systems*, 35:38546–38556.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Ben Bogin, Shivanshu Gupta, Matt Gardner, and Jonathan Berant. 2021. COVR: A test-bed for visually grounded compositional generalization with real images. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9824–9846, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Peter Clark, Oyvind Tafjord, and Kyle Richardson. 2020. Transformers as soft reasoners over language. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 3882–3890. International Joint Conferences on Artificial Intelligence Organization.

Bhavana Dalvi, Peter Jansen, Oyvind Tafjord, Zhengnan Xie, Hannah Smith, Leighanna Pipatanangkura, and Peter Clark. 2021. Explaining answers with entailment trees. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7358–7370, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jerry A Fodor and Zenon W Pylyshyn. 1988. Connectionism and cognitive architecture: A critical analysis. *Cognition*, 28(1-2):3–71.

Xiyan Fu and Anette Frank. 2023. SETI: Systematicity evaluation of textual inference. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 4101–4114, Toronto, Canada. Association for Computational Linguistics.

Xiyan Fu and Anette Frank. 2024. Exploring continual learning of compositional generalization in NLI. *Transations of the Association for Computational Linguistics*.

Difei Gao, Ruiping Wang, Shiguang Shan, and Xilin Chen. 2022. Cric: A vqa dataset for compositional reasoning on vision and commonsense. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Atticus Geiger, Kyle Richardson, and Christopher Potts. 2020. Neural natural language inference models partially embed theories of lexical entailment and negation. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 163–173, Online. Association for Computational Linguistics.

Emily Goodwin, Koustuv Sinha, and Timothy J. O'Donnell. 2020. Probing linguistic systematicity. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1958–1969, Online. Association for Computational Linguistics.

Nitish Gupta, Kevin Lin, Dan Roth, Sameer Singh, and Matt Gardner. 2019. Neural module networks for reasoning over text. In *International Conference on Learning Representations*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *International Conference on Learning Representations*.

Ronghang Hu, Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Kate Saenko. 2017. Learning to reason: End-to-end module networks for visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 804–813.

Drew A Hudson and Christopher D Manning. 2019. Gqa: A new dataset for real-world visual reasoning and compositional question answering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6700–6709.

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

Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. 1991. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87.

Peter Jansen, Elizabeth Wainwright, Steven Marmorstein, and Clayton Morrison. 2018. WorldTree: A corpus of explanation graphs for elementary science questions supporting multi-hop inference. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.

Hang Jiang, Sairam Gurajada, Qiuhao Lu, Sumit Neelam, Lucian Popa, Prithviraj Sen, Yunyao Li, and Alexander Gray. 2021. LNN-EL: A neuro-symbolic approach to short-text entity linking. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 775–787, Online. Association for Computational Linguistics.

Yichen Jiang and Mohit Bansal. 2019. Self-assembling modular networks for interpretable multi-hop reasoning. 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 4474–4484, Hong Kong, China. Association for Computational Linguistics.

Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. 2017. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2901–2910.

Tushar Khot, Daniel Khashabi, Kyle Richardson, Peter Clark, and Ashish Sabharwal. 2021. Text modular networks: Learning to decompose tasks in the language of existing models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1264–1279, Online. Association for Computational Linguistics.

Tushar Khot, Harsh Trivedi, Matthew Finlayson, Yao Fu, Kyle Richardson, Peter Clark, and Ashish Sabharwal. 2023. Decomposed prompting: A modular approach for solving complex tasks. In *International Conference on Learning Representations*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*.

Alice Lai, Yonatan Bisk, and Julia Hockenmaier. 2017. Natural language inference from multiple premises. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 100–109, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Brenden Lake and Marco Baroni. 2018. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *International conference on machine learning*, pages 2873–2882. PMLR.

Hung Le, Nancy Chen, and Steven Hoi. 2022. VGNMN: Video-grounded neural module networks for video-grounded dialogue systems. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3377–3393, Seattle, United States. Association for Computational Linguistics.

Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. 2021. {GS}hard: Scaling giant models with conditional computation and automatic sharding. In *International Conference on Learning Representations*.

Bo Li, Yifei Shen, Jingkang Yang, Yezhen Wang, Jiawei Ren, Tong Che, Jun Zhang, and Ziwei Liu. 2023. Sparse mixture-of-experts are domain generalizable learners. In *The Eleventh International Conference on Learning Representations*.

Qing Li, Siyuan Huang, Yining Hong, Yixin Chen, Ying Nian Wu, and Song-Chun Zhu. 2020. Closed loop neural-symbolic learning via integrating neural perception, grammar parsing, and symbolic reasoning. In *International Conference on Machine Learning*, pages 5884–5894. PMLR.

Linqing Liu, Patrick Lewis, Sebastian Riedel, and Pontus Stenetorp. 2022. Challenges in generalization in open domain question answering. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 2014–2029, Seattle, United States. Association for Computational Linguistics.

Gary F Marcus. 2003. *The algebraic mind: Integrating connectionism and cognitive science*. MIT press.

Pascual Martínez-Gómez, Koji Mineshima, Yusuke Miyao, and Daisuke Bekki. 2017. On-demand injection of lexical knowledge for recognising textual entailment. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 710–720, Valencia, Spain. Association for Computational Linguistics.

Koji Mineshima, Pascual Martínez-Gómez, Yusuke Miyao, and Daisuke Bekki. 2015. Higher-order logical inference with compositional semantics. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2055–2061, Lisbon, Portugal. Association for Computational Linguistics.

Melanie Mitchell. 2021. Abstraction and analogy-making in artificial intelligence. *Annals of the New York Academy of Sciences*, 1505(1):79–101.

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. 2016. A corpus and cloze evaluation for deeper understanding of commonsense stories. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 839–849, San Diego, California. Association for Computational Linguistics.

Yingjie Niu, Linyi Yang, Ruihai Dong, and Yue Zhang. 2023. Learning to generalize for cross-domain QA. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 1298–1313, Toronto, Canada. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Alessandro Raganato and Jörg Tiedemann. 2018. An analysis of encoder representations in transformer-based machine translation. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 287–297, Brussels, Belgium. Association for Computational Linguistics.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

Jacob Russin, Randall C O'Reilly, and Yoshua Bengio. 2020. Deep learning needs a prefrontal cortex. *Work Bridging AI Cogn Sci*, 107:603–616.

Mohammed Saeed, Naser Ahmadi, Preslav Nakov, and Paolo Papotti. 2021. RuleBERT: Teaching soft rules to pre-trained language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1460–1476, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Swarnadeep Saha, Sayan Ghosh, Shashank Srivastava, and Mohit Bansal. 2020. PRover: Proof generation for interpretable reasoning over rules. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 122–136, Online. Association for Computational Linguistics.

Abulhair Saparov, Richard Yuanzhe Pang, Vishakh Padmakumar, Nitish Joshi, Mehran Kazemi, Najoung Kim, and He He. 2023. Testing the general deductive reasoning capacity of large language models using OOD examples. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Thibault Sellam, Dipanjan Das, and Ankur Parikh. 2020. BLEURT: Learning robust metrics for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892, Online. Association for Computational Linguistics.

Oyvind Tafjord, Bhavana Dalvi, and Peter Clark. 2021. ProofWriter: Generating implications, proofs, and abductive statements over natural language. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3621–3634, Online. Association for Computational Linguistics.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Yuhuai Wu, Albert Jiang, Jimmy Ba, and Roger Baker Grosse. 2021. {INT}: An inequality benchmark for evaluating generalization in theorem proving. In *International Conference on Learning Representations*.

Zhengnan Xie, Sebastian Thiem, Jaycie Martin, Elizabeth Wainwright, Steven Marmorstein, and Peter Jansen. 2020. WorldTree v2: A corpus of science-domain structured explanations and inference patterns supporting multi-hop inference. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5456–5473, Marseille, France. European Language Resources Association.

Hitomi Yanaka, Koji Mineshima, Daisuke Bekki, and Kentaro Inui. 2020. Do neural models learn systematicity of monotonicity inference in natural language? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6105–6117, Online. Association for Computational Linguistics.

Hitomi Yanaka, Koji Mineshima, and Kentaro Inui. 2021. Exploring transitivity in neural NLI models through veridicality. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 920–934, Online. Association for Computational Linguistics.

Kexin Yi, Jiajun Wu, Chuang Gan, Antonio Torralba, Pushmeet Kohli, and Josh Tenenbaum. 2018. Neural-symbolic vqa: Disentangling reasoning from vision and language understanding. *Advances in neural information processing systems*, 31.

Pengcheng Yin, Hao Fang, Graham Neubig, Adam Pauls, Emmanouil Antonios Platanios, Yu Su, Sam Thomson, and Jacob Andreas. 2021. Compositional generalization for neural semantic parsing via span-level supervised attention. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2810–2823, Online. Association for Computational Linguistics.

Pengcheng Yin, Chunting Zhou, Junxian He, and Graham Neubig. 2018. StructVAE: Tree-structured latent variable models for semi-supervised semantic parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 754–765, Melbourne, Australia. Association for Computational Linguistics.

Zhangdie Yuan, Songbo Hu, Ivan Vulić, Anna Korhonen, and Zaiqiao Meng. 2023. Can pretrained language models (yet) reason deductively? In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 1447–1462, Dubrovnik, Croatia. Association for Computational Linguistics.

# A Data

## A.1 Reasoning Types in EntailmentBank

We list six different reasoning types in EntailmentBank dataset in Table 5.

## A.2 Data Shapes in EntailmentBank

People normally assume that trees can take various shapes, even when their depth is limited to four. However, this assumption does not hold in our CSEG task. We extract every potential shape from the dataset (Dalvi et al., 2021) and find only six different shapes (shape-* in Fig. 2) exist. This is because trees do not reflect or distinguish the different orders of siblings. That is, for a single multi-premise step of an entailment tree, the order of multiple premises (siblings) is underspecified.

## A.3 Data Construction for DBpedia

We constructed the DBpedia dataset to evaluate the compositional generalization of MORSE and other baselines. Hence, DBpedia needs to contain several rules, and instances using one of these rules to process each step in multi-step reasoning. We extracted six reasoning rules as shown in Table 3 from a rules pool. Following RuleBert (Saeed et al., 2021) (Section 4.4 Chaining of Rule Executions), we generate hypotheses given existing rules over different relations and a depth D. Subsequently, we instantiate variables in rules and hypotheses from a name pool to generate instances. Rules and hypotheses are eventually transferred to natural language by pre-defined templates.

## A.4 Data Construction for EntB-L and EntB-Sh

EntailmentBank contains 1,840 entailment trees showing how a hypothesis is entailed from a small number of relevant sentences. We constructed the EntailmentBank-Length (EntB-L) and EntailmentBank-Shape (EntB-Sh) for compositional generalization evaluation. In terms of EntB-L, we extracted data from the original dataset by the 'length_of_proof' label. As for EntB-Sh, we extracted data from the original dataset by the 'lisp_proof' label. An example of the shape of extracted trees is shown in Fig. 2.

## A.5 Data Statistics for EntailmentBank and DBPedia

Table 6 provides detailed data statistics of EntailmentBank and DBPedia. It contains the general data information for each dataset, and the data partitions we created and used in generalization evaluation. We use 20% of the training data for validation.

| Rules |
|---|
| R1: child(B,A) $\rightarrow$ parent(A,B) |
| R2: child(A,C) $\wedge$ parent(C,B) $\rightarrow$ spouse(A,B) |
| R3: spouse(A,C) $\wedge$ parent(B,C) $\rightarrow$ negspouse(A,B) |
| R4: relative(A,C) $\wedge$ child(C,B) $\rightarrow$ relative(A,B) |
| R5: parent(A,C) $\wedge$ spouse(B,C) $\rightarrow$ parent(A,B) |
| R6: parent(A,C) $\wedge$ parent(B,C) $\rightarrow$ relative(A,B) |

Table 3: Rules applied in DBpedia datasets.

## A.6 Data Statistic for Correlation Analysis

To visualize the correlations between modules and rules, we constructed a new group of samples containing three reasoning steps. We select samples: i) that contain three reasoning steps, ii) that have correct predictions for the first two reasoning steps, but iii) where the third step is incorrectly predicted in case a certain head is removed. The number of selected samples for each head is given in Table 4. We then count samples in each head over different rules and show the correlations in Fig. 5.

| | H1 | H2 | H3 | H4 | H5 | H6 |
|---|---|---|---|---|---|---|
| cases | 126 | 104 | 137 | 118 | 104 | 126 |

Table 4: Rules applied in DBpedia datasets.

## A.7 Real Examples

We provide real examples of the productivity (length) test in Fig. 6.

# B Experimental Details

## B.1 Hyperparameter

We use the T5 checkpoints from Huggingface (Wolf et al., 2020). For initialization, we treat all layers as plain transformer layers. We optimize our model using Adam Optimizer (Kingma and Ba, 2014) with learning rate 1e-4 and batch size 4. In inference, we adopt beam search decoding with beam size 3 for all models and baselines. We set the threshold $\tau$ for dynamic mask construction to 0.1 (details in Appendix B). We use 20% of training or fine-tuning datasets for validation. All models are evaluated with 3 rounds.

## B.2 Training Details

**MORSE** We conduct out-of-distribution experiments for increasing lengths and shapes of reason-

| Reasonoing Types | Example |
|---|---|
| Substitution | s1: when a light wave hits a reflective object, the light wave will be reflected<br>s2: a mirror is a kind of reflective object<br>int: when a light wave hits a mirror, the light wave will be reflected |
| Inference from Rule | s1: puddles of water are outside during the day<br>s2: if something is outside during the day then that something will receive sunlight<br>int: puddles of water will receive sunlight |
| Further Specification or Conjuction | s1: an animal requires warmth for survival as the season changes to winter<br>s2: thick fur can be used for keeping warm<br>int: thick fur can be used for keeping warm as the season changes to winter |
| Infer Class from Properties | s1: A compound is made of two or more elements chemically combined<br>s2: sodium chloride is made of two elements chemically combined<br>int: sodium chloride is a kind of compound |
| Property Inheritance | s1: an animal's shell is usually hard<br>s2: something hard can be used for protection<br>int: an animal's shell is usually hard for protection |
| Sequential Inference | s1: In molecular biology, translation follows transcription<br>s2: transcription is when genetic information flows from DNA to RNA<br>s3: translation is when genetic information flows from RNA to proteins<br>int: In molecular biology, genetic information flows from DNA to RNA to proteins |

Table 5: Six different reasoning types in EntailmentBank (Dalvi et al., 2021)

| Dataset | EntB | DBP | | EntB-L(ength) | | | DBP-L(ength) | | | | EntB-Sh(apes) | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| partitions | | | | tr | ft | te | tr | ft | te | | tr | ft | te |
| #avg.nodes | 7.6 | 4 | $L_1$ | 430 | / | / | 1800 | / | / | A1 | 390 | / | / |
| #avg.steps | 3.2 | 1.7 | $L_2$ | 450 | / | / | 1800 | / | / | A2 | 391 | / | / |
| #reas.types | 6 | 6 | $L_3$ | / | 300 | 101 | / | 160 | 391 | A3 | 219 | / | / |
| #examples | 1840 | 4560 | | | | | | | | B1 | / | 79 | 36 |
| | | | | | | | | | | B2 | / | 63 | 26 |
| | | | | | | | | | | B3 | / | 64 | 39 |
| | | | all | 880 | | | 3600 | | | all | 1000 | 206 | 101 |

Table 6: Data statistics of Ent(ailment)B(ank) and DBP(edia). We split data into different partitions, including tr(ain), f(ine-)t(une) and te(st). $L_n$ denotes different lengths, and A*, B* means various shapes.

ing trees on two benchmarks, to test MORSE's generalization abilities. Our experiments are run on Nvidia GTX 1080 Ti. As for length compositional test, MORSE (T5-Small and T5-Large) is trained for 33k steps and fine-tuning 4.5k steps on EntailmentBank-Length; trained for 8.1k steps and fine-tuning 0.6k steps on DBpedia-Length. In shape compositional test, MORSE is trained 25k steps and fine-tuning 5k steps.

**Baselines** Since ProofWriter-It and Entailment Writer are all T5-based baselines, we keep their settings as same as MORSE. In terms of Prover, we choose to use BERT-base-uncased version, given its parameters approach T5-small. We use the grid search technology for generation and select the best result. Its learning rate is 3e-5, trained for 36k steps and fine-tuning 4.5k steps on EntailmentBank-Length. In shape compositional test, Prover is trained 27k step and fine-tuning 5.5k steps.

## C Analysis for Different Shapes

In Fig. 4 we note that shape B1 is the most difficult for all systems, and provide an empirical analysis: we hypothesize that combining trees of unequal lengths at higher levels makes the task more challenging compared to lower levels. Here, we further conduct a statistical Spearman's rank correlation coefficient analysis of systematicity difficulty from the complexity of tree properties to verify our hypotheses.

For each test shape, we aim to determine how much the presence of specific tree properties influences the task accuracy of models (including baselines and our model MORSE) when performing systematicity generalization from primitive to compositional shapes. Specifically, we quantified the increase of accuracy in view of the following aspects: i) increased number of the 'Leaf' ($\Delta$#Leaf) nodes from (seen) primitive units to (predicted) compositional structures. I.e., how much the leaf length 1

**sent1**: animals need food for surviving
**sent2**: a bear is a kind of animal
**hypothesis**: a bear needs food for surviving

**entailment tree**:
sent1 & sent2 -> hypothesis [*substitution*]

length 2

**sent1**: puddles of water are outside during the day
**sent2**: temperature is a measure of heat energy
**sent3**: if something receives sunlight, it will increase in temperature
**hypothesis:** the puddles of water will increase in heat energy

**entailment tree :**
sent1 & sent3 -> int1: the puddles of water will increase in temperature; [*if-then*]
sent2 & int1 -> hypothesis [*substitution*]

length 3

**sent1**: a plate is made of metal
**sent2**: metal is a thermal conductor
**sent3**: if something is a thermal conductor, it can efficiently transmit heat
**sent4**: if something can transmit heat, it can be used for cooking
**hypothesis**: a metal plate can be used for cooking

**entailment tree** :
sent1 & sent2 -> int1: a metal plate is a thermal conductor [*substitution*]
sent3 & int1 -> int2: a metal spoon can efficiently transmit heat [*if-then*]
sent4 & int2 -> hypothesis [*if-then*]

Figure 6: Three real examples for the productivity-length test of CSEG. For each example, an entailment tree is generated based on candidate sentences and a hypothesis. Each tree is composed of several reasoning steps, and each step belongs to one specific reasoning type, here, either [*substitution*] or [*if-then*]. The length of each sample is determined by how many reasoning steps are required for the entailment tree generation. To evaluate the compositional generalization ability, we design CSEG to generalize from limited reasoning steps (e.g., length 1 or length 2) to more steps (e.g., length 3). Here, the sample of length three is compositional, and since its required reasoning types have been learned before, it is expected to be solvable.

| ComplexityDim | ProofW | PROVER | EntailW | Morse | avg |
|---|---|---|---|---|---|
| Δ#Leaf | 0.5 | 0.86 | 0.5 | 0.5 | 0.59 |
| Δ#InterNode | -0.86 | -0.5 | -0.86 | -0.86 | -0.77 |
| Δ#InterNode-L2 | 0.86 | 1.0 | 0.86 | 0.86 | 0.895 |
| Δ#InterNode-L3 | -1 | -0.86 | -1 | -1 | -0.965 |

Table 7: Spearman's rank correlation coefficient between the increase of training–test arithmetic complexity and the compositional generalization performance (accuracy) across the three shapes. *avg* is the average value.

number increased from primitive samples (e.g., A1, A2) to compositional samples (e.g., B1) and how this influences accuracy; ii) increased number of 'Intermediate Nodes' (Δ#InterNode) (again from primitive to compositional structures) and how this influences generalization accuracy.

Table 7 shows the results of our Spearman's rank correlation coefficient analysis between these two complexity dimensions of trees and the compositional generalization accuracy. Compared to the 'Leaf' dimension, 'Intermediate Nodes' shows a more notable average coefficient value.[10] That is, the more intermediate nodes in the compositional samples, the more difficult it is for the neural model to perform compositional generalization.

Based on this result, we further explore whether the location of intermediate nodes will affect compositional generalization ability. We evaluate: i) increased number of the 'Intermediate Node' at layer 2 (Δ#InterNode-L2). Layer 2 indicates the second layer of a tree from the bottom up, e.g., B1 has one intermediate node in the second layer, and B3 has two. ii) increased number of 'Intermediate Nodes' at layer 3 (Δ#InterNode-L3). Table 7 indicates that more intermediate nodes in layer three incur a notable negative value, i.e., intermediate nodes at a higher layer result in lower accuracy, meaning that compositional generalization is more difficult.

In conclusion, Table 4 indicates that the systematicity test in CSEG is challenging for existing neural models. And further exploration verifies combining trees at higher levels makes it even more difficult compared to lower levels.

[10]The permutation of a small set (here, 3 dimensions) is limited, thus limiting the range of variation of the correlation coefficient. Hence, 0.59 is an irrelevant value.

| Models | Original EntailmentBank Dataset (EntB-Orig) | | | | | |
|---|---|---|---|---|---|---|
| | Leaves | | Steps | | Intermediates | |
| | F1 | AllCorrect | F1 | AllCorrect | F1 | AllCorrect |
| Task 1 (no-distractor) - EntailmentWriter - T511b | 99.0 | 89.4 | 51.5 | 38.2 | 71.2 | 38.5 |
| Task 1 (no-distractor) - EntailmentWriter - T5Large | 98.7 | 86.2 | 50.5 | 37.7 | 67.6 | 36.2 |
| Task 1 (no-distractor) - MORSE (ours) - T5Large | 98.09(0.24) | 86.37(0.11) | 51.11(0.84) | 39.70(0.77) | 69.79(0.09) | 40.97(0.34) |
| Task 1 (no-distractor) - EntailmentWriter - T5Small | 98.40(0.41) | 86.18(0.25) | 41.72(0.96) | 34.11(0.38) | 56.95(0.21) | 40.41(0.49) |
| Task 1 (no-distractor) - MORSE (ours) - T5Small | 98.30(0.37) | 86.47(0.21) | 42.35(0.66) | 35.00(0.32) | 57.76(0.11) | 40.88(0.51) |
| Task 2 (distractor) - EntailmentWriter - T511b | 89.1 | 48.8 | 41.4 | 27.7 | 66.2 | 31.5 |
| Task 2 (distractor) - EntailmentWriter - T5Large | 84.3 | 35.6 | 35.5 | 22.9 | 61.8 | 28.5 |
| Task 2 (distractor) - MORSE (ours) - T5Large | 83.17(0.95) | 34.41(0.59) | 34.46(0.62) | 21.96(0.60) | 60.50(0.19) | 28.24(0.37) |

Table 8: Comparative results for Entailment Writer vs. MORSE on original EntailmentBank dataset for Task 1 and Task 2 with different T5 model sizes

## D Comparative results on original EntailmentBank dataset

We conduct experiments of Task 1 and Task 2 from Dalvi et al. (2021) on the *original EntailmentBank dataset and splits*. The train, dev and test sets contain 1,313, 187 and 340 instances. Task 2 includes non-fitting distractor sentences in the input. We compare differently scaled T5 models to assess differences relating from T5 model sizes: T511b, T5large. EntailmentWriter (EW) is equivalent to MORSE modulo its modulated reasoning cell. For EW we show published results from Dalvi et al. (2021); for MORSE we report averaged results over three runs w/ standard deviation in brackets, for T5large. We observe comparable or superior results of MORSE w/T5large over EW w/t5large, especially for the difficult Steps (entailment tree structure) and Intermediates (inferred intermediate node label) evaluation criteria for Task 1. For Task 2, which poses a challenge by including noisy distractors, MORSE is still competitive, with ca. 1 percentage point distance. Comparing results of EW w/T511b vs. MORSE w/T5large shows that can MORSE rival and even outperform EW using T511b, for Steps and Intermediats Accuracies in Task 1, but not for the more difficult Task 2. The experiment shows that despite using a variation of the dataset in our main experiments to focus on MORSE's generalization abilities, it is still competitive on the original dataset and data distributions.

## E Analysis of Dynamic Masking Mechanism

**Mask Sparsity** MORSE deploys masks to modularize a network dynamically. This allows each module to specialize for a specific function while selecting corresponding inputs. To gain more insight into the role of dynamic masking, we analyse masks used in length generalization testing on EntB-L. We count the number of masks with non-zero values for each module. Table 9 shows that the percentage of *non-zero values* for heads H1-6 is relatively low, indicating that dynamic masks are effective for filtering out potentially irrelevant inputs. We also note higher percentages for some modules (e.g., H4, H5). Different reasoning types require disparate inputs that may account for this.

| Head | H1 | H2 | H3 | H4 | H5 | H6 |
|---|---|---|---|---|---|---|
| non-zero (%) | 21.46 | 22.14 | 21.11 | 33.13 | 41.31 | 21.18 |

Table 9: Non-zero values in masks for each module (%).

**Mask Effects** We apply different masking strategies to test if the observed performance improvements arise from modularized masks – as opposed to naïve ones. We construct a *random_mask* model variant with 20 and 50% non-zero values, respectively. These proportions are similar to what we find in MORSE (Tab. 9). We apply random masks in length composition testing on the EntB-L dataset. Table 10 shows that compared to dynamic routing in MORSE, random masking incurs a severe performance drop. We conclude that i) unselective masking risks shielding important information from heads, and that ii) dynamic routing cannot be considered as a simple dropout mechanism.

| Models | Steps | | Intermediates | |
|---|---|---|---|---|
| | F1 | Acc | F1 | Acc |
| w modularized_mask | 44.22 | 32.67 | 50.66 | 25.74 |
| w random_mask (20%) | 30.36 | 15.84 | 42.62 | 13.86 |
| w random_mask (50%) | 36.63 | 20.79 | 45.45 | 18.81 |

Table 10: Effects of different mask strategies. (*%) indicates *% percentage of non-zero value in a mask.

| Models | EntailmentBank-Length (EntB-L) | | | | | | DBpedia-Length (DBP-L) | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Leaves | | Steps | | Intermediates | | Leaves | | Steps | | Intermediates | |
| | F1 | AllCorrect | F1 | AllCorrect | F1 | AllCorrect | F1 | AllCorrect | F1 | AllCorrect | F1 | AllCorrect |
| EntWriter (T5-Large) | 99.78 | 98.02 | 52.80 | 40.92 | 56.62 | 36.63 | 99.36 | 95.52 | 82.49 | 80.11 | 88.98 | 83.89 |
| MORSE (T5-Large) | **99.82**(+0.04) | **98.68**(+0.66) | **53.31**(+0.51) | **42.57**(+1.65) | **57.78**(+1.16) | **37.29**(+0.66) | **99.53**(+0.17) | **96.68**(+1.16) | **86.79**(+4.30) | **83.76**(+3.65) | **92.62**(+3.64) | **86.70**(+2.81) |
| EntWriter (Flan-T5-Large) | 99.78 | 98.02 | 53.18 | 41.58 | 57.93 | 39.13 | 99.53 | 95.52 | 84.98 | 83.12 | 91.27 | 84.14 |
| MORSE (Flan-T5-Large) | 100.00(+0.22) | 100.00(+1.98) | 55.51(+2.33) | 43.56(+1.98) | 58.67(+0.74) | 39.60(+0.47) | 99.53(-0) | 96.68(+1.16) | 87.21(+2.23) | 83.76(+0.64) | 93.41(+2.14) | 86.70(+2.56) |
| EntWriter-0-shot (T5-Large) | 97.06 | 85.73 | 18.44 | - | 24.21 | - | 90.09 | 29.27 | 16.94 | - | 32.43 | - |
| MORSE-0-shot (T5-Large) | **97.89**(+0.83) | **86.83**(+1.10) | **19.14**(+0.70) | - | **25.42**(+1.21) | - | **89.82**(-0.17) | **30.05**(+0.78) | **18.41**(+1.47) | - | **33.45**(+1.02) | - |
| EntWriter-0-shot (Flan-T5-Large) | 98.79 | 91.09 | 20.59 | - | 31.68 | - | 90.05 | 30.69 | 18.46 | - | 33.30 | - |
| MORSE-0-shot (Flan-T5-Large) | 99.82(+1.03) | 92.31(+1.22) | 21.22(+0.63) | - | 32.07(+0.39) | - | 91.96(+1.91) | 31.28(+0.59) | 21.99(+3.53) | - | 33.92(+0.62) | - |

Table 11: Results on EntailmentBank-L(ength) and DBpedia-L(ength) for compositional generalization evaluation based on Flan-T5. (+num) indicates the improvement of MORSE compared to the strong baseline EntWriter.

# F Morse on powerful backbones

To further investigate the effectiveness of MORSE, we conduct experiments for MORSE and the most competitive baseline EntWriter on a more powerful backbone, e.g., Flan-T5 (Chung et al., 2022). Table 11 shows results. We find that: i) compared to T5, FLAN-T5 has generally better results for both models in both settings (fine-tuning and zero-shot). With FLAN-T5, our extension with MORSE still has superior results compared to the original T5 model. That is, our conclusions remain the same with this new backbone. ii) for both EntWriter and MORSE, FLAN-T5 shows increased performance in the zero-shot setting. This indicates that FLAN-T5 may serve as a better model variant to address zero-shot setting – which is expected for an instruction-tuned model.