Real-World Compositional Generalization with Disentangled Sequence-to-Sequence Learning

Hao Zheng and **Mirella Lapata**
Institute for Language, Cognition and Computation
School of Informatics, University of Edinburgh
10 Crichton Street, Edinburgh EH8 9AB
`Hao.Zheng@ed.ac.uk` `mlap@inf.ed.ac.uk`

Abstract

Compositional generalization is a basic mechanism in human language learning, which current neural networks struggle with. A recently proposed **D**isent**angle**d sequence-to-sequence model (Dangle) shows promising generalization capability by learning specialized encodings for each decoding step. We introduce two key modifications to this model which encourage more disentangled representations and improve its compute and memory efficiency, allowing us to tackle compositional generalization in a more realistic setting. Specifically, instead of adaptively re-encoding source keys and values at each time step, we disentangle their representations and only re-encode keys periodically, at some interval. Our new architecture leads to better generalization performance across *existing* tasks and datasets, and a *new* machine translation benchmark which we create by detecting *naturally occurring* compositional patterns in relation to a training set. We show this methodology better emulates real-world requirements than artificial challenges.[1]

1 Introduction

The Transformer architecture (Vaswani et al., 2017) and variants thereof have become ubiquitous in natural language processing. Despite widespread adoption, there is mounting evidence that Transformers as sequence transduction models struggle with *compositional generalization* (Kim and Linzen, 2020; Keysers et al., 2020; Li et al., 2021). It is basically the ability to produce and understand a potentially infinite number of novel linguistic expressions by systematically combining known atomic components (Chomsky, 2014; Montague, 1970). Attempts to overcome this limitation have explored various ways to explicitly inject compositional bias through data augmentation (Jia and Liang, 2016; Akyürek et al., 2021; Andreas, 2020; Wang et al., 2021) or new training objectives (Conklin et al., 2021; Oren et al., 2020; Yin et al., 2021). The majority of existing approaches have been designed with semantic parsing in mind, and as a result adopt domain- and task-specific grammars or rules which do not extend to other tasks (e.g., machine translation).

In this work we aim to improve generalization via general architectural modifications which are applicable to a wide range of tasks. Our starting point are Zheng and Lapata (2022) who unveil that one of the reasons hindering compositional generalization in Transformers relates to their representations being entangled. They introduce Dangle, a sequence-to-sequence model, which learns more **D**isent**angle**d representations by adaptively re-encoding (at each time step) the source input. For each decoding step, Dangle learns specialized source encodings by conditioning on the newly decoded target which leads to better compositional generalization compared to vanilla Transformers where source encodings are shared throughout decoding. Although promising, their results are based on synthetic datasets, leaving open the question of whether Dangle is effective in real-world settings involving both complex natural language and compositional generalization.

We present two key modifications to Dangle which encourage learning *more disentangled* representations *more efficiently*. The need to perform re-encoding at each time step substantially affects Dangle's training time and memory footprint. It becomes prohibitively expensive on datasets with long target sequences, e.g., programs with 400+ tokens in datasets like SMCalFlow (Andreas et al., 2020). To alleviate this problem, instead of adaptively re-encoding at each time step, we only re-encode periodically, at some interval. Our decoder is no different from a vanilla Transformer decoder except that it just re-encodes once in a while in order to update its history information. Our second

[1] Our code and dataset will be available at `https://github.com/mswellhao/Dangle`.

Findings of the Association for Computational Linguistics: ACL 2023, pages 1711–1725
July 9-14, 2023 ©2023 Association for Computational Linguistics

modification concerns disentangling the representations of source keys and values, based on which the encoder in Dangle (and also in Transformers) passes source information to the decoder. Instead of computing keys and values using shared source encodings, we disassociate their representations: we encode source *values once* and re-encode *keys periodically*.

We evaluate the proposed model on existing benchmarks (Andreas et al., 2020; Li et al., 2021) and a new dataset which we create to better emulate a real-world setting. We develop a new methodology for *detecting* examples representative of compositional generalization in naturally occurring text. Given a training and test set: (a) we discard examples from the test set that contain out-of-vocabulary (OOV) or rare words (in relation to training) to exclude novel atoms which are out of scope for compositional generalization; (b) we then measure how compositional a certain test example is with respect to the training corpus; we introduce a metric which allows us to identify a candidate pool of highly compositional examples; (c) using uncertainty estimation, we further select examples from the pool that are both compositional in terms of surface form and challenging in terms of generalization difficulty. Following these three steps, we create a *machine translation* benchmark using the IWSLT 2014 German-English dataset as our training corpus and the WMT 2014 German-English shared task as our test corpus.

Experimental results demonstrate that our new architecture achieves better generalization performance across tasks and datasets and is adept at handling real-world challenges. Machine translation experiments on a diverse corpus of 1.3M WMT examples show it is particularly effective for long-tail compositional patterns.

2 Background: The Dangle Model

We first describe Dangle, the **D**isent**angle**d Transformer model introduced in Zheng and Lapata (2022) focusing on their encoder-decoder architecture which they show delivers better performance on complex tasks like machine translation.

Let $X = [x_1, x_2, ..., x_n]$ denote a source sequence; let $f_{\texttt{Encoder}}$ and $f_{\texttt{Decoder}}$ denote a Transformer encoder and decoder, respectively. X is first encoded into a sequence of N contextualized representations:

$$N = f_{\texttt{Encoder}}(X) \tag{1}$$

which are then used to decode target tokens $[y_1, y_2, ..., y_m]$ one by one. At the t-th decoding step, the Transformer takes y_t as input, reusing the source encodings N and target memory M_{t-1} which contains the history hidden states of all decoder layers corresponding to past tokens $[y_1, y_2, ..., y_{t-1}]$:

$$y_{t+1}, M_t = f_{\texttt{Decoder}}(y_t, M_{t-1}, N) \tag{2}$$

This step not only generates a new token y_{t+1}, but also updates the internal target memory M_t by concatenating M_{t-1} with the newly calculated hidden states corresponding to y_t.

Dangle differs from vanilla Transformers in that it concatenates the source input with the previously decoded target to construct target-dependent input for *adaptive* decoding:

$$C_t = [x_1, x_2, ..., x_n, y_1, ..., y_t] \tag{3}$$

$$H_t = f_{\texttt{Adaptive\_Encoder}}(C_t) \tag{4}$$

The adaptive encoder consists of two components. C_t is first fed to k_1 Transformer encoder layers to fuse the target information:

$$\bar{H}_t = f_{\texttt{Adaptive\_Encoder}_1}(C_t) \tag{5}$$

where $\bar{H}_t$ is a sequence of contextualized representations $[\bar{h}_{t,1}, \bar{h}_{t,2}, ..., \bar{h}_{t,n}, \bar{h}_{t,n+1}, ..., \bar{h}_{t,n+t}]$. Then, the first n vectors corresponding to source tokens are extracted and fed to another k_2 Transformer encoder layers for further processing:

$$H_t = f_{\texttt{Adaptive\_Encoder}_2}(\bar{H}_t[:n]) \tag{6}$$

Finally, the adaptive source encodings H_t together with the target context $[y_1, y_2, ..., y_t]$ are fed to a Transformer decoder to predict y_{t+1}:

$$y_{t+1}, M_t = f_{\texttt{Decoder}}(y_{<t+1}, \{\}, H_t) \tag{7}$$

In a departure from vanilla Transformers, Dangle does not reuse the target memory from previous steps, but instead re-computes *all* target-side hidden states based on new source encodings H_t.

Similarly to Transformers, Dangle accesses source information at each decoding step via encoder-decoder attention layers where the same encodings H_t are used to compute both keys K_t and values V_t:

$$K_t = H_t W^K \tag{8}$$

$$V_t = H_t W^V \tag{9}$$

$$O_t = \text{Attention}(Q_t, K_t, V_t) \tag{10}$$

where key and value projections W^K and W^V are parameter matrices; and Q_t, K_t, V_t, and O_t are respectively query, key, value, and output matrices, at time step t.

3 The R-Dangle Model

In this section, we describe the proposed model, which we call R-Dangle as a shorthand for **R**eal-world **D**isent**angle**d Transformer.

3.1 Re-encoding at Intervals

The need to perform re-encoding (and also re-decoding) at each time step substantially increases Dangle's training cost and memory footprint, so that it becomes computationally infeasible for real-world language tasks with very long target sequences (e.g., in the region of hundreds of tokens). Adaptively re-encoding at every time step essentially means separating out relevant source concepts for each prediction. However, the Transformer is largely capable of encoding source phrases and decoding corresponding target phrases (or logical form fragments in semantic parsing), as evidenced by its remarkable success in many machine translation and semantic parsing benchmarks (Vaswani et al., 2017; Keysers et al., 2020; Zheng and Lapata, 2021). This entails that the entanglement problem (i.e., not being able to dis-associate the representations of different concepts for a sequence of predictions) does not occur very frequently. We therefore relax the strict constraint of re-encoding at every step in favor of the more flexible strategy of re-encoding at intervals.

Given source sequence $X = [x_1, x_2, ..., x_n]$, we specify $P = [t_1, t_2, ..., t_l](t_{i+1} - t_i = o)$ in advance, i.e., a sequence of re-encoding points with interval o. Then, during decoding, when reaching a re-encoding point $t(t = t_i)$, we update source encodings H_t and target memory M_t:

$$H_t = f_{\texttt{Adaptive\_Encoder}}(C_t) \quad (11)$$

$$y_{t+1}, M_t = f_{\texttt{Decoder}}(y_{<t+1}, \{\}, H_t) \quad (12)$$

where $f_{\texttt{Adaptive\_Encoder}}$ denotes the adaptive encoder described in Section 2. For the next time step $t(t_i < t < t_{i+1})$, we fall back to the vanilla Transformer decoder using the source encodings H_{t_i} computed at time step t_i:

$$y_{t+1}, M_t = f_{\texttt{Decoder}}(y_t, M_{t-1}, H_{t_i}) \quad (13)$$

Note that we always set t_1 to 1 to perform adaptive encoding at the first time step.

3.2 Disentangling Keys and Values

During decoding, Dangle accesses source information via cross-attention (also known as encoder-decoder attention) layers where the same source encodings are used to compute both keys *and* values. The core design principle underlying Dangle is that learning specialized representations for different purposes will encourage the model to zero in on relevant concepts, thereby disentangling their representations. Based on the same philosophy, we assume that source keys and values encapsulate different aspects of source information, and that learning more specialized representations for them would further improve disentanglement, through the separation of the concepts involved.

A straightforward way to implement this idea is using two separately parameterized encoders to calculate two groups of source encodings (i.e., corresponding to keys and values, respectively) during re-encoding. However, in our preliminary experiments, we observed this leads to serious overfitting and performance degradation. Instead, we propose to encode values once and update keys only during adaptive encoding. We compute source *values* via the standard Transformer encoder:

$$H^v = f_{\texttt{Encoder}}(X) \quad (14)$$

and adaptively re-encode source *keys* at an interval:

$$H_t^k = f_{\texttt{Adaptive\_Encoder}}(C_t) \quad (15)$$

$$y_{t+1}, M_t = f_{\texttt{KV\_Decoder}}(y_{<t+1}, \{\}, H^v, H_t^k) \quad (16)$$

where $f_{\texttt{KV\_Decoder}}$ denotes a slightly modified Transformer decoder where source keys and values in each cross-attention layer are calculated based on different source encodings:

$$K_t = H_t^k W^K \quad (17)$$

$$V = H^v W^V \quad (18)$$

$$O_t = \text{Attention}(Q_t, K_t, V) \quad (19)$$

At time step t (where $t_i < t < t_{i+1}$), we perform vanilla Transformer decoding:

$$y_{t+1}, M_t = f_{\texttt{KV\_Decoder}}(y_t, M_{t-1}, H^v, H_{t_i}^k) \quad (20)$$

Note that fully sharing values could potentially cause some entanglement, however, we we did not observe this in practice. We also experimented with a variant where keys are shared and values are repeatedly re-computed but empirically observed

Selected	Examples	Compositional Degree	Uncertainty
✗	but what can we do about this ?	2 / 8 = 0.25	—
✗	please report all changes here .	5 / 6 = 0.83	0.054
✔	you have disabled your javascript !	5 / 6 = 0.83	0.274

Table 1: Candidate examples from the WMT corpus. Different n-grams previously seen in the IWSLT training corpus are highlighted in color. The first example is composed of two n-grams (*but* and *what can we do about this?*) with a compositional degree 0.25, and is discarded in the second stage. The second example has a high compositional degree but receives a low uncertainty score, and is thus filtered in the third stage. The third example scores high in terms of both compositional degree and uncertainty, and is included in the compositional test set.

it obtains significantly worse generalization performance than the value-sharing architecture described above. This indicates that entanglement is more likely to occur when sharing keys.

4 A Real-world Compositional Generalization Challenge

Models of compositional generalization are as good as the benchmarks they are evaluated on. A few existing benchmarks are made of artificially synthesized examples using a grammar or rules to systematically control for different types of generalization (Lake and Baroni, 2018a; Kim and Linzen, 2020; Keysers et al., 2020; Li et al., 2021). Unfortunately, synthetic datasets lack the complexity of real natural language and may lead to simplistic modeling solutions that do not generalize to real world settings (Dankers et al., 2022). Other benchmarks (Finegan-Dollak et al., 2018; Shaw et al., 2021) focus on naturally occurring examples but create train-test splits based on the properties of their formal meaning representations (e.g., logical forms). However, formal annotations of meaning are not readily available for tasks other than semantic parsing. Since compositional generalization is a general problem, it is desirable to define it on the basis of natural language alone rather than by means of semantic parsing and the availability of formal annotations.

It is fair to assume that a SOTA model deployed in the wild, e.g., a Transformer-based machine translation system, will be constantly presented with new test examples. Many of them could be similar to seen training instances or compositionally different but in a way that does not pose serious generalization challenges. An ideal benchmark for evaluating compositional generalization should therefore consist of phenomena that are of practical interest while challenging for SOTA models. To this end, we create ReaCT, a new **REA**l-world dataset for **C**ompositional generalization in machine **T**ranslation. Our key idea is to obtain a generalization test set by *detecting* compositional patterns in relation to an existing training set from a large and diverse pool of candidates. Specifically, we use the IWSLT 2014 German → English dataset as our training corpus and the WMT 2014 German → English shared task as our test corpus (see Section 5 for details) and detect from the pool of WMT instances those that exemplify compositional generalization with respect to IWSLT. This procedure identifies naturally occurring compositional patterns which we hope better represent practical generalization requirements than artificially constructed challenges.

In the following, we describe how we identify examples that demand compositional generalization. While we create our new benchmark with machine translation in mind, our methodology is general and applicable to other settings such as semantic parsing. For instance, we could take a relatively small set of annotated user queries as our training set and create a generalization challenge from a large pool of unlabeled user queries.

Filtering Out-of-Vocabulary Atoms Compositional generalization involves generalizing to *new* compositions of *known* atoms. The WMT corpus includes many new semantic and syntactic atoms that are not attested in IWSLT. A large number of these are out-of-vocabulary (OOV) words which are by definition unknown and out of scope for compositional generalization. We thus discard WMT examples with words occurring less than 3 times in the IWSLT training set which gives us approximately a pool of 1.3M examples. For simplicity, we do not consider any other types of new atoms such as unseen word senses or syntactic patterns.

Measuring Compositionality How to define the notion of compositional generalization is a central question in creating a benchmark. Previous definitions have mostly centered around linguistic notions such as constituent or context-free grammars (Kim and Linzen, 2020; Keysers et al., 2020; Li et al., 2021). These notions are appropriate for synthetic examples or logical forms as their underlying hierarchical structures are well-defined and can be obtained with ease.

Since we do not wish to synthesize artificial examples but rather detect them in real-world utterances, relying on the notion of constituent might be problematic. Sentences in the wild are often noisy and ungrammatical and it is far from trivial to analyze their syntactic structure so as to reliably identify new compositions of known constituents. We overcome this problem by devising a metric based on n-gram matching which assesses how compositional a certain example is with respect to a training corpus.

Specifically, we first create a lookup dictionary of atomic units by extracting all n-grams that occur more than 3 times in the training corpus. Given a candidate sentence, we search the dictionary for the minimum number of n-grams that can be composed to form the sentence. For example, for sentence "$x_1x_2x_3x_4x_5$" and dictionary $(x_1, x_2, x_3x_4, x_5, x_1x_2, x_3x_4x_5,)$, the minimum set of such n-grams is $(x_1x_2, x_3x_4x_5)$. A sentence's *compositional degree* with respect to the training corpus is defined as the ratio of the minimum number of n-grams to its length (e.g., 2/5 = 0.4 for the above example). We select the top 60,000 non-overlapping examples with the highest compositional degree as our *candidate pool*. As we discuss in Section 6, compositional degree further allows us to examine at a finer level of granularity how model performance changes as test examples become increasingly compositional.

Estimating Uncertainty Examples with the same compositional degree could pose more or less difficulty to neural sequence models (see last two utterances in Table 1). Ideally, we would like to identify instances that are compositional in terms of surface form *and* hard in terms of the underlying generalization (see third example in Table 1). We detect such examples using a metric based on *uncertainty estimation* and orthogonal to compositional degree. We quantify predictive uncertainty based on model ensembles, a method which has been successfully applied to detecting misclassifications and out-of-distribution examples (Lakshminarayanan et al., 2017; Malinin and Gales, 2021).

We follow the uncertainty estimation framework introduced in Malinin and Gales (2021) for sequence prediction tasks. Specifically, we train 10 Transformer models with different random initializations on IWSLT (our training corpus), and run inference over the candidate pool created in the previous stage; for each example in this pool, we measure the disagreement between ensemble models using *reverse mutual information*, a novel measure (Malinin, 2019; Malinin and Gales, 2021) which quantifies *knowledge uncertainty*, i.e., a model's uncertainty in its prediction due to lack of understanding of the data rather than any intrinsic uncertainty associated with the task (e.g., a word could have multiple correct translations). We use the token-level approximation of knowledge uncertainty.

		Comp	Word n-gram		POS n-gram	
Dataset	# examples	Degree	2	3	2	3
COGS	21,000	0.392	6,097	24,275	12	27
CoGnition	10,800	0.502	1,865	13,344	1	38
CFQ	11,968	0.268	168	2,736	8	30
ReaCT	3,000	0.811	19,315	33,652	76	638

Table 2: Dataset Statistics: unique novel n-grams computed over words and parts of speech in ReaCT, and test partitions of COGS, CoGnition, and CFQ benchmarks.

We empirically find that the most uncertain examples are extremely noisy and barely legible (e.g., they include abbreviations, typos, and nonstandard spelling). We therefore throw away the top 2,000 uncertain examples and randomly sample 3,000 instances from the next 18,000 most uncertain examples in an attempt to create a generalization test set with diverse language patterns and different levels of uncertainty.

Analysis We analyze the compositional nature of ReaCT by comparing it to several popular benchmarks. Specifically, for all datasets, we count the number of novel test set n-grams that have not been seen in the training. We extract n-grams over words and parts of speech (POS); word-based n-grams represent more superficial lexical composition while n-grams based on POS tags reflect more of syntactic composition.

As shown in Table 2, despite being considerably smaller compared to other benchmarks (see # examples column), ReaCT presents substantially more diverse patterns in terms of lexical and syntactic composition. It displays a much bigger number of novel word n-grams, which is perhaps not surprising. Being a real-world dataset, ReaCT has a larger vocabulary and more linguistic variation. While

Train	Test
• and i can 't believe you 're here and that i 'm meeting you here at ted . (PRP RB IN NN .) • you see , this is what india is today . the ground reality is based on (DT NN NN VBZ VBN IN) a cyclical world view .	the account data is provided to you directly via e-mail .
• a couple of hours (DT NN IN NNS) later , the sun will shine on the next magnifying glass . • but this could also be used for good . (MD RB VB VBN IN NN .)	both setting of tasks must successfully be mastered under supervision .
• the national science foundation , other countries (JJ NN NN , JJ NN) are very interested in doing this • no , they are full of misery . (VBP JJ IN NN .)	its warm water temperature , small depth are convenient for bathing .

Table 3: Novel syntactic compositions in ReaCT test set (syntactic atoms of same type are color coded). POS-tag sequences for these atoms are shown in parentheses (PRP:pronoun, RB:adverb, IN: preposition, NN/S:noun singular/plural, DT: determiner, JJ: adjective, MD:modal, VBZ: verb, 3rd person singular, present tense, VBP: verb, present tense, other than third person singular, VBN: verb past participle.)

our dataset creation process does not explicitly target novel syntactic patterns (approximated by POS n-grams), ReaCT still includes substantially more compared to other benchmarks. This suggests that it captures the complexity of real-world compositional generalization to a greater extent than what is achieved when examples are synthesized artificially. We show examples with novel POS n-gram compositions in Table 3).

5 Experimental Setup

Datasets We evaluated R-Dangle on two machine translation datasets and one semantic parsing benchmark which we selected to maximally reflect natural language variations and real-world generalization challenges. These include: (a) **ReaCT**, the machine translation benchmark developed in this paper; we used the IWSLT 2014 De→En test set as an in-domain test set and created an out-of-distribution test set from the WMT'14 De→En training corpus; (b) **CoGnition** (Li et al., 2021) is a semi-natural machine translation benchmark focusing on English-Chinese sentence pairs; source sentences were taken from the Story Cloze Test and ROCStories Corpora (Mostafazadeh et al., 2016, 2017), and target sentences were constructed by post-editing the output of a machine translation engine; (c) **SMCalFlow-CS** (Andreas et al., 2020) is a semantic parsing dataset for task-oriented dialogue, featuring real-world human-generated utterances about calendar management; following previous work (Yin et al., 2021; Qiu et al., 2022), we report experiments on the compositional skills split, considering a few-shot learning scenario (with 6, 16, and 32 training examples). See Appendix A for more details on these datasets.

Models On machine translation, our experiments evaluated two variants of R-Dangle depending on whether keys and values are shared (R-Dangle$_{\text{shr}}$) or separate (R-Dangle$_{\text{sep}}$). We implemented all machine translation models with fairseq (Ott et al., 2019). We compared R-Dangle against a vanilla Transformer (Vaswani et al., 2017) and the original Dangle model (Zheng and Lapata, 2022) which used the popular fairseq configuration transformer_iwslt_de_en. We also implemented bigger variants of these models using 12 encoder layers and 12 decoder layers which empirically led to better performance.

R-Dangle$_{\text{shr}}$ and R-Dangle$_{\text{sep}}$ also use a 12-layer decoder. We tuned the number of layers of the adaptive components ($k_1 = 2$ and $k_2 = 10$) on the development set. For R-Dangle$_{\text{sep}}$, we adopted a 10-layer value encoder and a 10-layer key encoder ($k_1 = 2$ and $k_2 = 8$), with the top 8 layers in the two encoders being shared. This configuration produced 12 differently parametrized transformer encoder layers, maintaining identical model size to comparison systems.

Previous work (Qiu et al., 2022) has shown the advantage of pre-trained models on the SMCalFlow-CS dataset. For our semantic parsing experiments, we therefore built R-Dangle on top of BART-large (Lewis et al., 2020). We only report results with R-Dangle$_{\text{shr}}$ as the R-Dangle$_{\text{sep}}$ architecture is not compatible with BART. We again set $k_1 = 2$ and $k_2 = 10$. We provide more detail on model configurations in Appendix B.

6 Results

Disentangling Keys and Values Improves Generalization Table 4 reports the BLEU score (Papineni et al., 2002) achieved by the two R-Dangle

CoGnition	1	2	4	8
R-Dangle$_{\text{shr}}$	62.5	62.3	62.3	61.9
R-Dangle$_{\text{sep}}$	63.4	63.1	62.3	62.1
ReaCT	1	2	4	8
R-Dangle$_{\text{shr}}$	11.8	11.9	11.8	11.6
R-Dangle$_{\text{sep}}$	12.3	12.2	11.9	11.7

Table 4: BLEU score for R-Dangle variants (with different re-encoding intervals) on CoGnition and ReaCT compositional generalization test sets. Note that R-Dangle$_{\text{shr}}$ with interval 1 is Dangle.

variants on ReaCT and CoGnition, across different re-encoding intervals. R-Dangle$_{\text{sep}}$ is consistently better than R-Dangle$_{\text{shr}}$ which confirms that representing keys and values separately is beneficial. We also observe that smaller intervals lead to better performance (we discuss this further later).

Table 5 compares R-Dangle$_{\text{sep}}$ (with interval 1) against baseline models. In addition to BLEU, we report novel compound translation error rate, a metric introduced in Li et al. (2021) to quantify the extent to which novel compounds are mistranslated. We compute error rate over instances and an aggregate score over contexts. R-Dangle$_{\text{sep}}$ delivers compositional generalization gains over Dangle and vanilla Transformer models (both in terms of BLEU *and* compound translation error rate), even though their performance improves when adopting a larger 12-layer network. R-Dangle$_{\text{sep}}$ achieves a new state of the art on CoGnition (a gain of 0.9 BLEU points over Dangle and 1.5 BLEU points over the Transformer baseline). R-Dangle$_{\text{sep}}$ fares similarly on ReaCT; it is significantly superior to the Transformer model by 0.9 BLEU points, and Dangle by 0.5 BLEU points. Moreover, improvements on compositional generalisation are not at the expense of in-domain performance (R-Dangle obtains similar performance to the Transformer and Dangle on the IWSLT2014 in-domain test set).

R-Dangle Can Handle Long-tail Compositional Patterns Better We next examine model performance on real-world examples with diverse language and different levels of composition. Specifically, we train R-Dangle$_{\text{sep}}$ (interval=1) and a Transformer on the IWSLT14 corpus and test on the pool of 1.3M WMT examples obtained after filtering OOV words. Figure 1a plots the difference in BLEU between the two models against compositional degree. This fine-grained evaluation reveals that they perform similarly on the majority of less compositional examples (BLEU difference is around zero), however, the performance gap becomes larger with more compositional examples (higher difference means higher BLEU for R-Dangle$_{\text{sep}}$). This indicates that R-Dangle is particularly effective for handling long-tail compositional patterns.

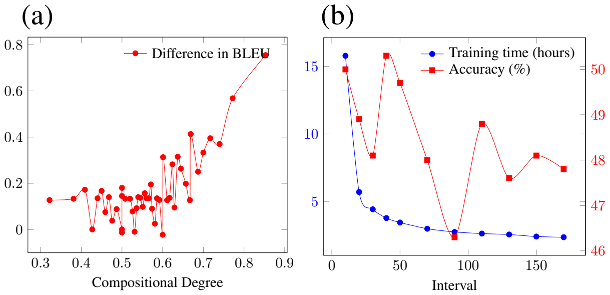

Figure 1: (a) Difference in BLEU score between R-Dangle$_{\text{sep}}$ (interval = 1) and Transformer vs compositional degree. A positive score means R-Dangle$_{\text{sep}}$ is better than Transformer. Each data point is computed on 30K WMT examples. R-Dangle shows increasing performance improvements as test examples become more compositional. (b) Training cost (hours) and test accuracy vs interval length. R-Dangle$_{\text{shr}}$ was trained on SMCalFlow-CS (16-ℂ) using 4 A100 GPUs.

R-Dangle Boosts the Performance of Pretrained Models The "pre-train and fine-tune" paradigm (Peters et al., 2018; Devlin et al., 2019; Raffel et al., 2020; Lewis et al., 2020) has been widely adopted in NLP, and semantic parsing is no exception (Shin et al., 2021; Qiu et al., 2022). We further investigate R-Dangle's performance when combined with a pre-trained model on the SMCalFlow-CS dataset (across the three cross-domain settings). Table 6 shows that R-Dangle$_{\text{shr}}$ boosts the performance of BART-large, which suggests that generalization improvements brought by R-Dangle are complementary to generalization benefits afforded by large-scale pre-training (see Zheng and Lapata 2022 for a similar conclusion). The proposed model effectively marries pre-training with disentangled representation learning to achieve better generalization.

In Table 6, we also compare R-Dangle with other top-performing models on SMCalFlow-CS. These include: (a) a sequence-to-sequence model with a BERT encoder and an LSTM decoder using a copy mechanism (BERT2SEQ; Yin et al. 2021); (b) the coarse-to-fine (C2F) model of Dong and Lapata (2018) which uses a BERT encoder and a structured decoder that factorizes the generation of a program into sketch and value predictions; (c) and combi-

Models	CoGnition				ReaCT	
	↓ $\text{ErrR}_{\text{Inst}}$	↓ $\text{ErrR}_{\text{Aggr}}$	↑ ind-test	↑cg-test	↑IWSLT14	↑cg-test
Transformer (Zheng and Lapata, 2022)	30.5	63.8	69.2	59.4	34.4	9.5
Dangle (Zheng and Lapata, 2022)	22.8	50.6	69.1	60.6	—	—
Transformer (our implementation)	23.4	53.7	70.8	61.9	36.0	11.4
Dangle (our implementation)	19.7	47.0	70.6	62.5	36.1	11.8
R-$\text{Dangle}_{\text{sep}}$ (interval = 1)	16.0	42.1	70.7	63.4	36.0	12.3

Table 5: **Machine Translation Results:** we compare R-Dangle to baseline models on CoGnition and ReaCT. For CoGnition, we report instance-wise and aggregate compound translation error rates (ErrR) on the compositional generalization test set (cg-test) and BLEU on both in-domain test set (ind-test) and cg-test. For ReaCT, we report BLEU on the in-domain IWSLT 2014 De→En test set and the compositional generalization test set (cg-test) created in this paper. Results are averaged over 5 random runs on CoGnition and 3 random runs on ReaCT.

System	8-$\mathbb{C}$	16-$\mathbb{C}$	32-$\mathbb{C}$
BERT2SEQ	—	33.6	53.5
BERT2SEQ+SS	—	46.8	61.7
C2F	—	40.6	54.6
C2F+SS	—	47.4	61.9
T5	34.7	44.7	59.0
T5+CSL	51.6	61.4	70.4
BART	32.1	47.2	61.9
+R-$\text{Dangle}_{\text{shr}}$ (interval = 6)	36.3	50.6	64.1

Table 6: **Semantic Parsing Results:** we compare R-Dangle to various systems on SMCalFlow-CS. *-$\mathbb{C}$ are different settings with 8, 16, and 32 cross-domain examples added to the training set. Results for BERT and C2F models are from Yin et al. (2021). Results for T5 models are from Qiu et al. (2022). Results for BART and R-Dangle are averaged over 3 random runs.

nations of these two models with span-supervised attention (+SS; Yin et al. 2021). We also include a T5 model and variant thereof trained on additional data using a model called Compositional Structure Learner (CSL) to generate examples for data augmentation (T5+CSL; Qiu et al. 2022). R-Dangle with BART performs best among models that do *not* use data augmentation across compositional settings. Note that our proposal is orthogonal to CSL and could also benefit from data augmentation.

Larger Re-encoding Intervals Reduce Training Cost The results in Table 4 indicate that re-encoding correlates with R-Dangle's generalization ability, at least for machine translation. Both model variants experience a drop in BLEU points when increasing the re-encoding interval to 8. We hypothesize that this sensitivity to interval length is task-related; target sequences in machine translation are relatively short and representative of real language, whereas in SMCalFlow-CS, the average length of target sequences (in formal language) is 99.5 and the maximum length is 411. It is computationally infeasible to train R-Dangle with small intervals on this dataset, however, larger intervals still produce significant performance gains.

Figure 1b shows how accuracy and training time vary with interval length on SMCalFlow-CS with the 16-$\mathbb{C}$ setting. Larger intervals substantially reduce training cost with an optimal speed-accuracy trade off in between 10 and 50. For instance, interval 40 yields a 4x speed-up compared to interval 10 while achieving 50.3% accuracy. Finding a tradeoff between generalization and efficiency is an open research problem which we leave to future work.

7 Related Work

The realization that neural sequence-to-sequence models struggle with compositional generalization has led to numerous research efforts aiming to precisely define this problem and explore possible solutions to it. A line of research focuses on benchmarks which capture different aspects of compositional generalization. Finegan-Dollak et al. (2018) repurpose existing semantic parsing benchmarks for compositional generalization by creating more challenging splits based on logical form patterns. In SCAN (Lake and Baroni, 2018b) compositional generalization is represented by unseen combinations of seen actions (e.g., JUMP LTURN). Keysers et al. (2020) define compositional generalization as generalizing to examples with maximum compound divergence (e.g., combinations of entities and relations) while guaranteeing similar atom distribution to the training set. Kim and Linzen (2020) design five linguistic types of compositional generalization such as generalizing phrase nesting to unseen depths. In ReaCT, our definition of compositional generalization is dependent on the data distribution of the candidate corpus, which determines what compositional patterns are of practical interest and how frequently they occur.

Another line of work focuses on modeling solutions, mostly ways to explicitly instil compositional bias into neural models. This can be achieved by adopting a more conventional grammar-based approach (Herzig and Berant, 2021) or incorporating a lexicon or lexicon-style alignments into sequence models (Akyurek and Andreas, 2021; Zheng and Lapata, 2021). Other work employs heuristics, grammars, and generative models to synthesize examples for data augmentation (Jia and Liang, 2016; Akyürek et al., 2021; Andreas, 2020; Wang et al., 2021; Qiu et al., 2022) or modifies standard training objectives with new supervision signals like attention supervision or meta-learning (Oren et al., 2020; Conklin et al., 2021; Yin et al., 2021). Our work builds on Dangle (Zheng and Lapata, 2022), a disentangled sequence-to-sequence model, which tries to tackle compositional generalization with architectural innovations. While Dangle is conceptually general, our proposal is tailored to the Transformer and features two key modifications to encourage more disentangled representations and better computational efficiency.

8 Conclusions

In this paper we focused on two issues related to compositional generalization. Firstly, we improve upon Dangle, an existing sequence-to-sequence architecture that generalizes to unseen compositions by learning specialized encodings for each decoding step. We show that re-encoding keys periodically, at some interval, improves both efficiency and accuracy. Secondly, we propose a methodology for identifying compositional patterns in real-world data and create a new dataset which better represents practical generalization requirements. Experimental results show that our modifications improve generalization across tasks, metrics, and datasets and our new benchmark provides a challenging testbed for evaluating new modeling efforts.

Limitations

Our machine translation experiments revealed that optimal generalization performance is obtained with small interval values. However, R-Dangle with small intervals still runs much slower than an equivalent Transformer model. Despite our modifications, large R-Dangle models with small intervals on large datasets remain computationally expensive. In this paper, we only explored a simple periodic re-encoding strategy. However, more complex and flexible ways of re-encoding could be used to further improve computational efficiency. For instance, we could adopt a dynamic strategy which *learns* when re-encoding is necessary.

Acknowledgments

We thank our anonymous reviewers for their feedback. We also wish to thank Tom Hosking and Biao Zhang for helpful discussions. Finally, we gratefully acknowledge the support of the UK Engineering and Physical Sciences Research Council (grant EP/W002876/1).

References

Ekin Akyürek, Afra Feyza Akyurek, and Jacob Andreas. 2021. Learning to recombine and resample data for compositional generalization. In *Proceedings of the 9th International Conference on Learning Representations*, Online.

Ekin Akyurek and Jacob Andreas. 2021. Lexicon learning for few shot sequence modeling. 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 4934–4946, Online. Association for Computational Linguistics.

Jacob Andreas. 2020. Good-enough compositional data augmentation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7556–7566, Online. Association for Computational Linguistics.

Jacob Andreas, John Bufe, David Burkett, Charles Chen, Josh Clausman, Jean Crawford, Kate Crim, Jordan DeLoach, Leah Dorner, Jason Eisner, Hao Fang, Alan Guo, David Hall, Kristin Hayes, Kellie Hill, Diana Ho, Wendy Iwaszuk, Smriti Jha, Dan Klein, Jayant Krishnamurthy, Theo Lanman, Percy Liang, Christopher H. Lin, Ilya Lintsbakh, Andy McGovern, Aleksandr Nisnevich, Adam Pauls, Dmitrij Petters, Brent Read, Dan Roth, Subhro Roy, Jesse Rusak, Beth Short, Div Slomin, Ben Snyder, Stephon Striplin, Yu Su, Zachary Tellman, Sam Thomson, Andrei Vorobev, Izabela Witoszko, Jason Wolfe, Abby Wray, Yuchen Zhang, and Alexander Zotov. 2020. Task-Oriented Dialogue as Dataflow Synthesis. *Transactions of the Association for Computational Linguistics*, 8:556–571.

Noam Chomsky. 2014. *Aspects of the Theory of Syntax*, volume 11. MIT press.

Henry Conklin, Bailin Wang, Kenny Smith, and Ivan Titov. 2021. Meta-learning to compositionally generalize. 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 3322–3335, Online. Association for Computational Linguistics.

Verna Dankers, Elia Bruni, and Dieuwke Hupkes. 2022. The paradox of the compositionality of natural language: A neural machine translation case study. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4154–4175, Dublin, Ireland. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Li Dong and Mirella Lapata. 2018. Coarse-to-fine decoding for neural semantic parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 731–742, Melbourne, Australia. Association for Computational Linguistics.

Catherine Finegan-Dollak, Jonathan K. Kummerfeld, Li Zhang, Karthik Ramanathan, Sesh Sadasivam, Rui Zhang, and Dragomir Radev. 2018. Improving text-to-SQL evaluation methodology. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 351–360, Melbourne, Australia. Association for Computational Linguistics.

Jonathan Herzig and Jonathan Berant. 2021. Span-based semantic parsing for compositional generalization. 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 908–921, Online. Association for Computational Linguistics.

Zhiheng Huang, Davis Liang, Peng Xu, and Bing Xiang. 2020. Improve transformer models with better relative position embeddings. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3327–3335, Online. Association for Computational Linguistics.

Robin Jia and Percy Liang. 2016. Data recombination for neural semantic parsing. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12–22, Berlin, Germany. Association for Computational Linguistics.

Daniel Keysers, Nathanael Schärli, Nathan Scales, Hylke Buisman, Daniel Furrer, Sergii Kashubin, Nikola Momchev, Danila Sinopalnikov, Lukasz Stafiniak, Tibor Tihon, Dmitry Tsarkov, Xiao Wang, Marc van Zee, and Olivier Bousquet. 2020. Measuring compositional generalization: A comprehensive method on realistic data. In *Proceedings of the 8th International Conference on Learning Representations*, Addis Ababa, Ethiopia.

Najoung Kim and Tal Linzen. 2020. 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, Online. Association for Computational Linguistics.

Brenden M. Lake and Marco Baroni. 2018a. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2879–2888. PMLR.

Brenden M. Lake and Marco Baroni. 2018b. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 2879–2888, Stockholm, Sweden. PMLR.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. 2017. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Yafu Li, Yongjing Yin, Yulong Chen, and Yue Zhang. 2021. On compositional generalization of neural machine translation. 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 4767–4780, Online. Association for Computational Linguistics.

Andrey Malinin. 2019. *Uncertainty Estimation in Deep Learning with Application to Spoken Language Assessment*. Ph.D. thesis, University of Cambridge.

Andrey Malinin and Mark Gales. 2021. Uncertainty estimation in autoregressive structured prediction. In *Proceedings of the 9th International Conference on Learning Representations*, Online.

Richard Montague. 1970. Universal grammar. *Theoria*, 36(3):373–398.

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.

Nasrin Mostafazadeh, Michael Roth, Annie Louis, Nathanael Chambers, and James Allen. 2017. LSDSem 2017 shared task: The story cloze test. In *Proceedings of the 2nd Workshop on Linking Models of Lexical, Sentential and Discourse-level Semantics*, pages 46–51, Valencia, Spain. Association for Computational Linguistics.

Inbar Oren, Jonathan Herzig, Nitish Gupta, Matt Gardner, and Jonathan Berant. 2020. Improving compositional generalization in semantic parsing. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2482–2495, Online. Association for Computational Linguistics.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. 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 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Linlu Qiu, Peter Shaw, Panupong Pasupat, Pawel Nowak, Tal Linzen, Fei Sha, and Kristina Toutanova. 2022. Improving compositional generalization with latent structure and data augmentation. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4341–4362, Seattle, United States. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Peter Shaw, Ming-Wei Chang, Panupong Pasupat, and Kristina Toutanova. 2021. Compositional generalization and natural language variation: Can a semantic parsing approach handle both? 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 922–938, Online. Association for Computational Linguistics.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 464–468, New Orleans, Louisiana. Association for Computational Linguistics.

Richard Shin, Christopher Lin, Sam Thomson, Charles Chen, Subhro Roy, Emmanouil Antonios Platanios, Adam Pauls, Dan Klein, Jason Eisner, and Benjamin Van Durme. 2021. Constrained language models yield few-shot semantic parsers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7699–7715, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Bailin Wang, Wenpeng Yin, Xi Victoria Lin, and Caiming Xiong. 2021. Learning to synthesize data for semantic parsing. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2760–2766, Online. Association for Computational Linguistics.

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.

Hao Zheng and Mirella Lapata. 2021. Compositional generalization via semantic tagging. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1022–1032, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Hao Zheng and Mirella Lapata. 2022. Disentangled sequence to sequence learning for compositional generalization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4256–4268, Dublin, Ireland. Association for Computational Linguistics.

A Dataset Details

We evaluated our model on two machine translation datasets, and one semantic parsing benchmark which we selected to maximally reflect natural language variations and real-world generalization challenges. We describe these in detail below.

ReaCT is the real-world machine translation benchmark developed in this paper for compositional generalization. The IWSLT 2014 De→En dataset consists of approximately 170K sequence pairs. We used the fairseq script `prepare-iwslt14.sh` to randomly sample approximately 4% of this dataset as validation set and kept the rest as training set. Following standard practice, we created an in-domain test set, the concatenation of files dev2010, dev2012, tst2010, tst2011, and tst2012. We created an out-of-distribution test sets from the WMT'14 De→En training corpus following the uncertainty selection method based on sequences.

CoGnition is another machine translation benchmark targeting compositional generalization (Li et al., 2021). It also contains a synthetic test set to quantify and analyze compositional generalization of neural MT models. This test set was constructed by embedding synthesized novel compounds into training sentence templates. Each compound was combined with 5 different sentence templates, so that every compound can be evaluated under 5 different contexts. A major difference between ReACT and CoGnition is the fact that test sentences for the latter are not naturally occurring. Despite being somewhat artificial, CoGnition overall constitutes a realistic benchmark which can help distinguish subtle model differences compared to purely synthetic benchmarks. For example, Zheng and Lapata (2022) showed that their encoder-only Dangle variant performed badly on this dataset in spite of impressive performance on synthetic semantic parsing benchmarks (Kim and Linzen, 2020; Keysers et al., 2020).

SMCalFlow-CS (Andreas et al., 2020) is a large-scale semantic parsing dataset for task-oriented dialogue, featuring real-world human-generated utterances about calendar management. Yin et al. (2021) proposed a compositional skills split of SMCalFlow (SMCalFlow-CS) that contains single-turn sentences from one of two domains related to creating calendar events (e.g., *Set up a meeting with Adam*) or querying an org chart (e.g., *Who are in Adam's team?*), paired with LISP programs. The training set $\mathbb{S}$ consists of samples from single domains while the test set $\mathbb{C}$ contains compositions thereof (e.g., *create a meeting with* *Adam and his team*). Since zero-shot compositional generalization is highly non-trivial due to novel language patterns and program structures, we follow previous work (Yin et al., 2021; Qiu et al., 2022) and consider a few-shot learning scenario, where a small number of cross-domain examples are included in the training set. We report experiments with 6, 16, and 32 examples.

B Implementation Details

Machine Translation Models We implemented all translation models with fairseq (Ott et al., 2019). Following previous work (Li et al., 2021; Zheng and Lapata, 2022), we compared with the baseline machine translation models Dangle and Transformer using the popular fairseq configuration `transformer_iwslt_de_en`. We also implemented a bigger variant of these models using a new configuration, which empirically obtained better performance. We used 12 encoder layers and 12 decoder layers. We set the dropout to 0.3 for attention weights and 0.4 after activations in the feed-forward network. We also used pre-normalization (i.e., we added layer normalization before each block) to ease optimization. Following Zheng and Lapata (2022), we used relative position embeddings (Shaw et al., 2018; Huang et al., 2020) which have demonstrated better generalization performance.

Hyperparameters for R-Dangle were tuned on the respective validation sets of CoGnition and ReaCT. Both R-Dangle$_{\text{shr}}$ and R-Dangle$_{\text{sep}}$ used a 12-layer decoder. For R-Dangle$_{\text{shr}}$, we tuned the number of layers of the two adaptive components k_1 and k_2, and set k_1 and k_2 to 2 and 10, respectively. For R-Dangle$_{\text{sep}}$, we shared some layers of parameters between the value encoder and the adaptive key decoder and experimented with different sharing strategies. Finally, we adopted a 10-layer value encoder and a 10-layer key encoder ($k_1 = 2$

and $k_2 = 8$). The top 8 layers in the two encoders were shared. This configuration produced 12 differently parametrized transformer encoder layers, thus maintaining identical model size to the baseline.

Semantic Parsing Models Qiu et al. (2022) showed the advantage of pre-trained sequence-to-sequence models on SMCalFlow-CS. We therefore built R-Dangle on top of BART-large (Lewis et al., 2020), which is well supported by fairseq. We used BART's encoder and decoder to instantiate the adaptive encoder and decoder in our model. For compatibility, we only employ the R-Dangle$_{\mathrm{shr}}$ architecture. We also set k_1 and k_2 to 2 and 10, respectively.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
the limitations section

☑ A2. Did you discuss any potential risks of your work?
the limitations section

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Left blank.

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

The section 4 and 5

☑ B1. Did you cite the creators of artifacts you used?
The section 5

☒ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
We were unable to find the license for the dataset we used

☒ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Left blank.

☒ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
We reuse existing datasets to create our benchmark.

☒ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Left blank.

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
the section 4

C ☑ Did you run computational experiments?

the section 5 and 6

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
the section 6

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
the section 5 and appedix

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
the section 6

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
the section 5

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

□ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

□ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

□ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

□ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

□ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.