Visually Grounded Concept Composition

Bowen Zhang [*]
USC

Hexiang Hu [†]
Google Research

Linlu Qiu [‡]
Google Research

Peter Shaw
Google Research

Fei Sha
Google Research

Abstract

We investigate ways to compose complex concepts in texts from primitive ones while grounding them in images. We propose Concept and Relation Graph (CRG), which builds on top of constituency analysis and consists of recursively combined concepts with predicate functions. Meanwhile, we propose a concept composition neural network called Composer to leverage the CRG for visually grounded concept learning. Specifically, we learn the grounding of both primitive and all composed concepts by aligning them to images and show that learning to compose leads to more robust grounding results, measured in text-to-image matching accuracy. Notably, our model can model grounded concepts forming at both the finer-grained sentence level and the coarser-grained intermediate level (or word-level). Composer leads to pronounced improvement in matching accuracy when the evaluation data has significant compound divergence from the training data.

1 Introduction

Visually grounded text expressions denote the images they describe. These expressions of visual concepts are naturally organized hierarchically in sub-expressions. The organization reveals structural relations that do not manifest when the sub-expressions are studied in isolation. For example, the phrase "a soccer ball in a gift-box" is a compound of two shorter phrases, *i.e.*, "a soccer ball" and "a gift-box", but carries the meaning of the spatial relationship "something in something" that goes beyond the two shorter phrases separately. The compositional structure of the grounded expression requires a concept learner to understand what primitive concepts are visually appearing and how the compound relating multiple primitives modifies their appearance.

Existing approaches (Kiros et al., 2014; Faghri et al., 2017; Lu et al., 2019; Chen et al., 2020, 2021) tackle visual grounding via end-to-end learning, which typically learns to align image and text information using neural networks without explicitly modeling their compositional structures. While neural networks have shown strong generalization capabilities in test examples that are i.i.d to the training distribution (Devlin et al., 2019), they often struggle in dealing with out-of-domain examples of novel compositional structures, in many tasks such as Visual Reasoning (Johnson et al., 2017; Bahdanau et al., 2019; Pezzelle and Fernández, 2019), Semantic Parsing (Finegan-Dollak et al., 2018; Keysers et al., 2020), and (Grounded) Command Following (Lake and Baroni, 2018; Chaplot et al., 2018; Hermann et al., 2017; Ruis et al., 2020).

In this work, we investigate how complex concepts, composed of simpler ones, are grounded in images at sentences, phrases and tokens levels. In particular, we investigate whether the structures of how these concepts are composed can be exploited as a modeling prior to improve visual grounding. To this end, we design Concept & Relation Graph (CRG), which is derived from constituency parse trees. The resulting CRG is a graph-structured database where concept nodes encode language expressions of concepts and their visual denotations (*e.g.*, a set of images corresponding to the concept), and predicate nodes define how a concept is semantically composed from its child concepts. Our graph is related to the denotation graph (Young et al., 2014; Zhang et al., 2020) but differs in two key aspects. First, our graph extracts the concepts without specially crafted heuristic rules[1]. Secondly,

[*] Part of work done while at Google
[†] Part of work done while at USC
[‡] Work done as a Google AI resident.

[1] Our graph construction relies on constituency parsing thus it is more scalable than hand-written rules initially developed for denotation graphs. The technique of denotation graph has

Findings of the Association for Computational Linguistics: EMNLP 2021, pages 201–215
November 7–11, 2021.

CRG's predicate can encode richer information explicitly than the subsumption relationships implicitly expressed in the denotation graphs. An illustrative figure of the graph is shown in Figure 1.

In addition to CRG, we propose **C**oncept c**OMPOS**ition transform**ER** (COMPOSER) that leverages the structure of text expressions to recursively encode the grounded concept embeddings, from coarse-level such as the noun words that refer to objects, to finer-grained ones with multiple levels of compositions. Transformer (Vaswani et al., 2017) is used as a building block in our model, to encode the predicates, and perform grounded concept composition. We learn COMPOSER using the task of visual-semantic alignment. Unlike traditional approaches, we perform hierarchical learning of visual-semantic alignment, which aligns the image to words, phrases, and sentences, and preserves the order of matching confidences.

We conduct experiments on multi-modal matching and show that COMPOSER achieves strong grounding capability in both sentence-to-image and phrase-to-image retrieval on the popular benchmarks. We validate the generalization capability of COMPOSER by designing an evaluation procedure for a more challenging compositional generalization task that uses test examples with maximum compound divergence (MCD) to the training data (Shaw et al., 2020; Keysers et al., 2020). Experiments show that COMPOSER is more robust to the compositional generalization than other approaches.

Our contributions are summarized as below:

- We study the compositional structure of visually grounded concepts and design Concept & Relation Graph that reflects such structures.
- We propose **C**oncept c**OMPOS**ition transform**ER** (COMPOSER) that recursively composes concepts using the child concepts and the semantically meaningful rules, which leads to strong compositional generalization performances.
- We propose a new evaluation task to assess the model's compositional generalization performances on the task of text-to-image matching and conduct comprehensive experiments to evaluate both baseline models and COMPOSER.

been developed and evaluated on English language corpus, and its multilingual utility depends on the parsing techniques for those languages other than English.

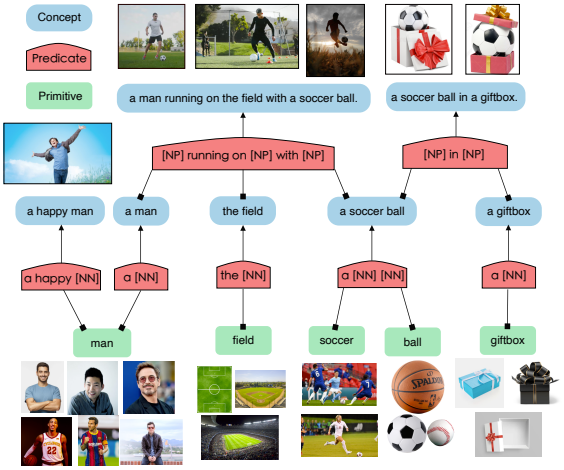


Figure 1: Concepts and their visual denotations organized by the Concept & Relation Graph

2 Concept & Relation Graph

We introduce multi-modal Concept and Relation Graph (CRG), a graph composed of concept and predicate nodes, which compose visually grounded descriptive phrases and sentences. Figure 1 provides an illustrative example. The concepts include sentences and intermediate phrases, shown as **blue nodes**. The primitives are the leaf nodes (typically noun words) that refer to visual objects, shown as **green nodes**. The predicates (**red nodes**) are n-ary functions that define the meaning of the concept composition. Their "signatures" consist of lexicalized templates, the number of arguments, and the syntactic type of the arguments. They combine primitives or simpler concepts into more complex ones.

Identifying concepts and relations. Given pairs of aligned image and sentence, we first parse a sentence into a constituency tree, using a state-of-the-art syntactic parser (Kitaev and Klein, 2018). We use the sentence's constituent tags to identify concepts and their relations. The set of relations are regarded as n-ary functions with placeholders denoted with constituency tags. We refer to such functions as predicates. Simpler concepts are arguments to the predicates, and the return values of the functions are complex concepts. The edges of the graph represent the relationship between predicates and their arguments. We restrict the type of constituents that can be concepts and how the predicates can be formed.

A concrete example is as follows: given an input concept "two dogs running on the grass", the algorithm extracts the predicate "`[NP] running`

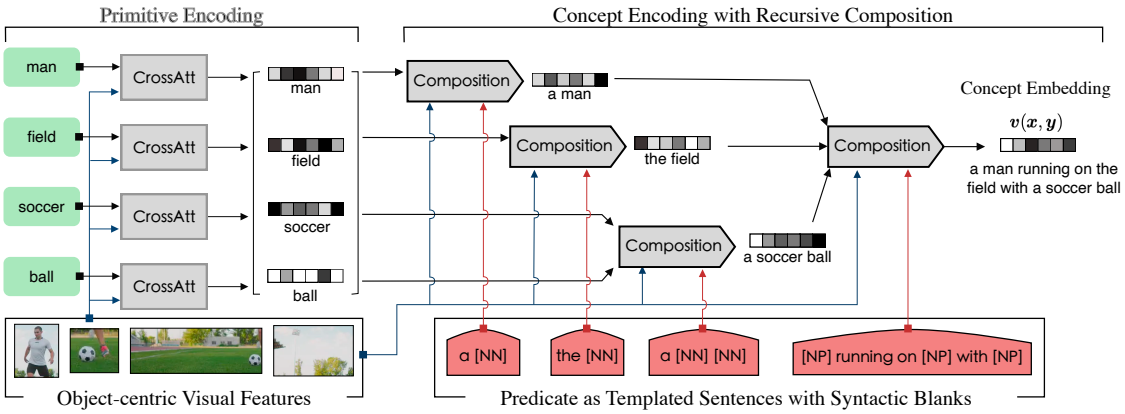


Figure 2: The overall design of the proposed COMPOSER model.

on [NP]" and the child concepts "two dogs" and "the grass". Here we use syntactic placeholders to replace the concept phrases. Details are in the Appendix. This idea is closely related to the semantically augmented parse trees (Ge and Mooney, 2009), though we focus on visually grounded concepts.

Finding visually grounded concepts. We take paired images and texts[2], and convert the texts into derived trees of predicates and primitives. With the generated text graph, we then group all images that refers to the same concept to form the image denotation, similar as Young et al. (2014) and Zhang et al. (2020). The image denotation is the set of images that contain the referred concept. For example, the image denotation of the concept "ball" is all the images that have the visual object category "ball". As a result, we associate the image denotation with each concept in the format of words, phrases, and sentences, which creates a multi-modal graph database as Figure 1.

3 COMPOSER: Recursive Modeling of the Compositional Structure

The main idea of COMPOSER is to recursively compose primitive concepts into sentences of complex structure, using composition rules defined by the predicates. Figure 2 presents a conceptual diagram of the high-level idea. Concretely, it first takes the primitive word embedding as the inputs and performs cross-modal attention to obtain their visually grounded word embeddings. Next, the COMPOSER calls the composition procedure to modify or combine primitive or intermediate concepts, according to the description of its predicates. At the end of this recursive procedure, we obtain the desired sentence concept embedding. In the rest of this section, we first discuss the notation and backgrounds, then introduce how primitives and predicates are encoded (§ 3.1), and present the recursive composition procedures in detail (§ 3.2). Finally, we discuss the learning objectives (§ 3.3).

Notation. We denote a paired image and sentence as $(\boldsymbol{x}, \boldsymbol{y})$ and the corresponding concepts and predicate for a tree $(\boldsymbol{x}, \boldsymbol{U}, \boldsymbol{E})$, where $\boldsymbol{U}, \boldsymbol{E}$ corresponds to the set of primitives and the set of predicates, respectfully. We also denote all concepts from a sentence $\boldsymbol{y}$ to be $\boldsymbol{C}$, where $\boldsymbol{U} \not\subset \boldsymbol{C}$ and $\boldsymbol{y} \in \boldsymbol{C}$.

Multi-head attention mechanism. Multi-Head Attention (MHA) (Vaswani et al., 2017) is the building block of our model. It takes three sets of input elements, *i.e.*, the key set K, the query set Q, and the value set V, and perform scaled dot-product attention as:

$$\texttt{MHA}(K, Q, V) = \texttt{FFN}\big(\texttt{Softmax}(\frac{Q^\top K}{\sqrt{d}}) \cdot V\big)$$

Here, d is the dimension of elements in K and Q. FFN is a feed-forward neural network. With different choices of K and V, MHA can be categorized as self-attention (SelfAtt) and cross-attention (CrossAtt), which corresponds to the variants with K and V including only the single-modality or cross-modality features.

3.1 Encoding Primitives and Predicates

Given a paired image and sentence $(\boldsymbol{x}, \boldsymbol{y})$, we parse the sentence as the tree of primitives and predicates $(\boldsymbol{x}, \boldsymbol{U}, \boldsymbol{E})$. Here, we represent the image as a set

[2] In this paper, texts refer to sentences.

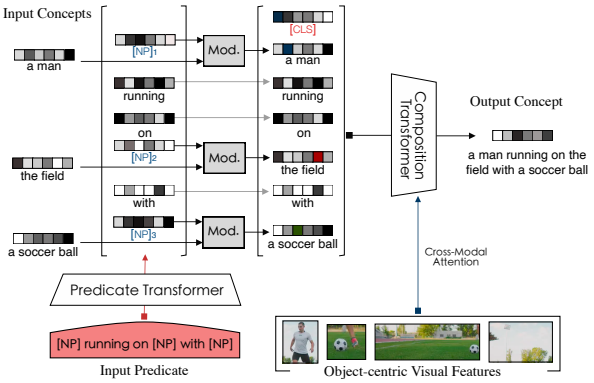


Figure 3: Details of the composition procedure.

of visual feature vectors $\{\boldsymbol{\phi}\}$, which are the object-centric features from an object detector (Anderson et al., 2018). Noted that we didn't use structural information beyond object proposals/regions. Our COMPOSER takes the primitives and predicates as input and output the visually grounded concept embeddings, with both the primitives and predicates as continuous vectors of different contextualization.

Representing primitives with visual context. The primitive concepts refer to tokens which can be visually grounded, and we represent them as word embeddings contextualized with visual features. As such, we use a one-layer Transformer with the `CrossAtt` mechanism, where K, V, and Q are linear transformations of $\boldsymbol{\phi}$, $\boldsymbol{\phi}$, and $\boldsymbol{u}$, respectively. This essentially uses the word embedding to query the visual features and outputs the grounded primitive embeddings $\hat{\boldsymbol{U}} = \{\hat{\boldsymbol{u}}\}$. Note that the output is always a single vector for each primitive as it is a single word.

Representing predicates as neural templates. A predicate $\boldsymbol{e}$ is a semantic n-place function that combines multiple concepts into one. We represent it as a **template sentence** with words and syntactic placeholders, such as "$\texttt{[NP]}_1$ running on $\texttt{[NP]}_2$", where those syntactic placeholders denote the positions and types of arguments. We encode such template sentences via `SelfAtt` mechanism, using a multi-layer Predicate Transformer (PT). The output of this model is a contextualized sequence of the words and syntactic placeholders as $\hat{\boldsymbol{e}}$.

3.2 Recursive Concept Composition

With the encoded primitives $\hat{\boldsymbol{U}}$ and predicates $\hat{\boldsymbol{E}}$, the COMPOSER then performs multiple recursive composition steps to obtain the grounded concept embedding, $\boldsymbol{v}(\boldsymbol{x}, \boldsymbol{y})$, representing the visual-linguistic embedding of the sentence and the image as shown in the Figure 2. To further illustrate this process, we detail the composition function in below, as shown in Figure 3.

Input concept modulation. We use a modulator to bind the arguments in the predicate to the input child concepts. Given a encoded predicate $\hat{\boldsymbol{e}} = \{\texttt{[NP]}_1, \texttt{running}, \texttt{on}, \texttt{[NP]}_2, \texttt{with}, \texttt{[NP]}_3\}$ and a input concept $\boldsymbol{c}_1 =$ "`a man`", the modulator is a neural network that takes the concept embedding $\boldsymbol{c}_1$ and its corresponding syntactic placeholder $\texttt{[NP]}_1$ as input and outputs a modulated embedding. This embedding is then reassembled with the embeddings of non-arguments in the predicate and used for the later stage. For example, the output sequence becomes $\{\texttt{Mod}(\texttt{[NP]}_1, \boldsymbol{c}_1), \texttt{running}, \texttt{on}, \texttt{Mod}(\texttt{[NP]}_2, \boldsymbol{c}_2), \texttt{with}, \texttt{Mod}(\texttt{[NP]}_3, \boldsymbol{c}_3)\}$ after the modulator processed each pair of input concept and syntactic placeholder. Various choices of neural networks are available for this modulator, such as a Multi-Layer Perceptron (MLP) or a Feature-wise Linear Modulation (FiLM) (Perez et al., 2018). COMPOSER uses FiLM for its strong empirical performance.

Contextualization with visual context. After concept modulation, we get a sequence of embeddings for non-argument words of the predicate and the binded child concepts, which is then fed as an input to a Composition Transformer (CT) model. This Transformer has multiple layers, with both `CrossAtt` layers that attends to the object-centric visual features and `SelfAtt` layers that contextualize between tokens. Please refer to Appendix for the detailed network architecture.

Given that our model is recursive by nature, the computation complexity of CT is proportional to the depth of the tree. We provide a comprehensive study in § 5.3 to show the correlation between the parameter/complexity and model's performances.

3.3 Learning COMPOSER with Visual-Semantic Alignments

With the composed grounded concept embedding $\boldsymbol{v}(\boldsymbol{x}, \boldsymbol{y})$, we use the visual-semantic alignment as the primary objective to learn COMPOSER. To this end, we compute the alignment score by learning an additional linear regressor θ:

$$\boldsymbol{s}(\boldsymbol{x}, \boldsymbol{y}) = \theta^\top \cdot \boldsymbol{v}(\boldsymbol{x}, \boldsymbol{y}) \propto p(\boldsymbol{x}, \boldsymbol{y}),$$

where $p(\boldsymbol{x}, \boldsymbol{y})$ is the probability that the sentence and image is a good match pair. Then we learn

the sentence to image alignment by minimizing the negative log-likelihood (NLL):

$$\ell_{\text{MATCH}} = -\sum_i \log \frac{\exp(s(\boldsymbol{x}_i, \boldsymbol{y}_i))}{\sum_{(\hat{\boldsymbol{x}},\hat{\boldsymbol{y}})\sim\mathcal{D}_i} \exp(s(\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}))}$$

with $\mathcal{D}_i = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\} \cup \mathcal{D}_i^-$. To properly normalize the probability, it is necessary to sample a set of negative examples to contrast. Thus, we generate $\mathcal{D}_i^-$ using the strategy of Lu et al. (2019).

Multi-level visual-semantic alignment (MVSA). Since COMPOSER composes grounded concepts recursively from the primitives, we obtain the embeddings of all the intermediate concepts automatically. Therefore, it is natural to extend the alignment learning objectives to all those intermediate concepts. We optimize the triplet hinge loss (Kiros et al., 2014):

$$\ell_{\text{MVSA}} = \sum_i \sum_{\boldsymbol{c}\in\boldsymbol{C}_i} [\alpha - s(\boldsymbol{x}_i, \boldsymbol{c}) + s(\boldsymbol{x}_i, \boldsymbol{c}^-)]_+ + [\alpha - s(\boldsymbol{x}_i, \boldsymbol{c}) + s(\boldsymbol{x}_i^-, \boldsymbol{c})]_+$$

where $[h]_+ = \max(0, h)$ denotes the hinge loss and α is the margin to be tuned. We derive the negative concepts $\boldsymbol{c}^-$ from the negative sentences in the $\mathcal{D}_i^-$. We observe that negative concepts at word/phrase levels are noisier than the ones at sentence level because many are common objects presented in the positive image and lead to ambiguity in learning. Therefore, we choose hinge loss over NLL because it is more robust to label noises (Biggio et al., 2011).

Learning to preserve orders in the tree. Finally, we use an order-preserving objective proposed by Zhang et al. (2020), to ensure that a fine-grained concept (closer to sentence) can produce a more confident alignment score than a coarse-grained concept (closer to primitive):

$$\ell_{\text{ORDER}} = \sum_i \sum_{\boldsymbol{e}_{jk}} [\beta - s(\boldsymbol{x}_i, \boldsymbol{c}_j) + s(\boldsymbol{x}_i, \boldsymbol{c}_k)]_+$$

Here, $\boldsymbol{e}_{jk}$ represents a predicate connecting the $\boldsymbol{c}_j$ and $\boldsymbol{c}_k$, with $\boldsymbol{c}_j$ to be the fine-grained parent concept which is closer to the sentence and $\boldsymbol{c}_k$ to be the coarse-grained child concept which is closer to the primitives. β is the margin that sets the constraint on how hard the order of embeddings should be reserved.

The complete learning objective is a weighted combination of three individual losses defined above, with the loss weights $\lambda_1 = 1$ and $\lambda_2 = 1$:

$$\ell = \ell_{\text{MATCH}} + \lambda_1 \cdot \ell_{\text{MVSA}} + \lambda_2 \cdot \ell_{\text{ORDER}}$$

The details of model optimization and hyperparameter setting are included in the Appendix.

4 Related Work

Generalization in grounded language understanding. Many evaluation methods are proposed to assess the model's generalization capabilities in grounded language understanding. Johnson et al. (2017) proposes a synthetic dataset, *i.e.* CLEVR, to evaluate the generalization of visual question answering models to novel objects and attributes. Misra et al. (2017) proposes to evaluate compositional generalization capability of visual models w.r.t. short phrases consist of attributes and objects. Chaplot et al. (2018) and Hermann et al. (2017) evaluate RL agents' capability to generalize to a novel composition of shape, size, and color in 3D simulators, which shows that RL agents generalize poorly. gSCAN (Ruis et al., 2020) perform a systematic benchmark to assess command following in a grounded environment. In this work, we focus on assessing model composition generalization under the visual context.

Compositional networks. State-of-the-art visually grounded language learning typically use deep Transformer models (Vaswani et al., 2017) such as ViLBERT (Lu et al., 2019), LXMERT (Tan and Bansal, 2019) and UNITER (Chen et al., 2020). Though being effective for data over i.i.d distribution, these models do not explicitly exploit the structure of the language and are thus prone to fail on compositional generalization. In contrast, another thread of works (Andreas et al., 2016; Yi et al., 2018; Mao et al., 2019; Shi et al., 2019; Wang et al., 2018) parse the language into an executable program composed as a graph of atomic neural modules, where each module is designed to perform atomic tasks and are learned end-to-end. Such models show almost perfect performances on synthetic benchmarks (Johnson et al., 2017) but perform subpar on the real-world data (Young et al., 2014; Chen et al., 2015) that are noisy and highly variable. Unlike them, we propose using a compositional neural network based on the Transformer architecture, which extends state-of-the-art neural networks to explicitly exploits language structure.

Dataset	# concepts	# predicates	# primitives	Avg height
F30K	408,464	122,196	10,755	3.09
C30K	345,331	88,623	9,683	2.86

Table 1: Statistics of the concepts and predicates in the F30K and C30K datasets.

5 Experiment

In this section, we perform experiments to validate the proposed COMPOSER model on the tasks of sentence-to-image retrieval and phrase-to-image retrieval. We begin with introducing the setup in § 5.1 and then present the main results in § 5.2, comparing models for their in-domain, cross-dataset evaluation, and compositional generalization performance. Finally, we perform an analysis and ablation study of our model design in § 5.3.

5.1 Experiment Setup

Datasets. We perform experiments on the COCO-caption (COCO) (Chen et al., 2015) and Flickr30K (F30K) (Young et al., 2014) datasets. Each image of these two datasets is associated with five sentences. Flickr30K contains 31,000 images, and we use the same data split as (Faghri et al., 2017), where there are 29,000 training images, 1000 test images, and 1000 validation images. COCO contains 123,287 images in total. For fast iteration, we use a subset training data C30K, which contains the same amount of images as the F30K. Note that C30K is a training split. We also trained models on the full COCO training split. For COCO dataset, the results are evaluated on COCO 1K test split (Karpathy and Fei-Fei, 2015). We use COCO 1k test split for both in-domain (models trained on either C30K or full COCO training split and evaluate on COCO-caption) and cross-dataset transfer (models trained on F30K and evaluate on COCO-caption) evaluation. For both F30K and COCO 1K test split, there are 5,000 text queries and 1,000 candidate images to be retrieved. We report recall@1 (R1) and recall@5 (R5) as the primary retrieval metric.

Compositional generalization evaluation. To generate evaluations of compositional generalization, we use a method similar to that of Shaw et al. (2020) and Keysers et al. (2020) which maximizes *compound divergence* between the distribution of *compounds* in the evaluation set and in the training set. Here compounds are defined based on the predicates occurring in captions. Following this method, we first calculate the overall divergence of compounds from the evaluation data to the training data using predicates from all the sentences. Then, for each sentence in the evaluation data, we calculate a compound divergence with this specific example removed. We rank those sentences based on the difference of the compound divergence. Finally, we choose the top-K sentences with the largest compound divergence differences and its corresponding images to form the evaluation splits.

Using this method, we generate evaluation splits with 1,000 images and 5,000 text queries, COCO-MCD and F30K-MCD, to assess models trained on F30K and COCO, respectively. Therefore, these splits assess both compositional generalization and cross-dataset transfer. Defining such splits across datasets is also helpful to achieve greater compound divergence than is otherwise possible, given the small amount of available in-domain test data. More details are included in Appendix.

CRG construction. We constructed two CRGs on the F30K and C30K datasets, using the procedure mentioned in § 2. The key statistics of the graph we generated as shown in Table 1.

Baselines and our approach. We compare COMPOSER to two strong baseline methods, *i.e.*, ViLBERT (Lu et al., 2019) and VSE (Kiros et al., 2014). We make sure all models are using the same object-centric visual features extracted from the Up-Down object detector (Anderson et al., 2018) for fair comparison. For the texts, both ViLBERT and the re-implemented VSE use the pre-trained BERT model as initialization. For the COMPOSER, we only initialize the predicate Transformer with the pre-trained BERT, which uses the first six layers. Note that the ViLBERT results are re-produced using the codebase from its author. ViLBERT is **not pre-trained** on any additional data of image-text pairs to prevent information leak in both cross-dataset evaluation and compositional generalization. Therefore, we used the pre-trained BERT models provided by HuggingFace to initalize the text stream of ViLBERT, and then followed the rest procedure in the original ViLBERT paper. Please refer to Appendix for complete details.

5.2 Main Results

We compare the COMPOSER with ViLBERT (Lu et al., 2019) and VSE (Kiros et al., 2014) on

(a) Models trained on F30K

Eval on	F30K		COCO		COCO-MCD	
Method	R1	R5	R1	R5	R1	R5
● VSE	46.84	77.16	25.60	54.36	21.82	47.58
▲ ViLBERT	50.94	**80.86**	30.50	58.98	24.44	51.44
◆ COMPOSER	**54.02**	80.27	**33.81**	**63.19**	**29.20**	**57.13**

(b) Models trained on C30K

Eval on	COCO		F30K		F30K-MCD	
Method	R1	R5	R1	R5	R1	R5
● VSE	45.74	**81.22**	27.66	55.92	23.44	47.90
▲ ViLBERT	**48.08**	81.10	31.12	58.88	24.02	49.34
◆ COMPOSER	47.87	80.93	**34.29**	**61.00**	**26.91**	**51.46**

Table 2: Text-to-Image retrieval results.

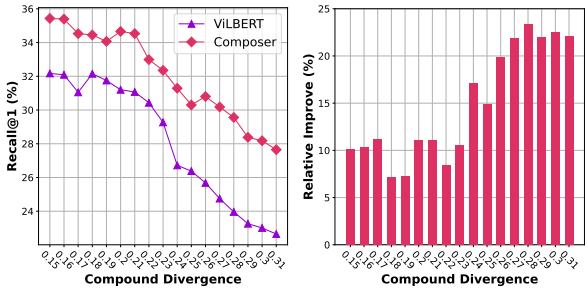


Figure 4: COMPOSER's results on generalization splits of different compound divergence over text description (evaluated under the F30K→COCO setting).

F30k and COCO for in-domain, zero-shot cross-dataset transfer, and compositional generalization (*e.g.* F30K→COCO-MCD). The notation A→B means that the model is trained on A and evaluated on B. We report the results of sentence-to-image retrieval in the main paper and defer more ablation study results to the Appendix.

In-domain performance. Table 2 presents the in-domain performance on both F30k and COCO datasets. First, we observe that both COMPOSER and ViLBERT consistently outperform VSE, which is expected as ViLBERT contains a cross-modal transformer with stronger modeling capacity. Comparing to ViLBERT, the COMPOSER performs on par.

Zero-shot cross-dataset transfer. We also consider zero-shot cross dataset transfer where we evaluate models on a dataset that is different from the training dataset. In this setting, the COMPOSER outperforms ViBLERT and VSE significantly. Concretely, on the F30k→COCO setting, the COMPOSER improves R1 and R5 by 11.0% and 7.0% over the ViLBERT, relatively. There are 10.0% and 4.2% relative improvements on R1 and R5 on the other transfer direction.

	F30K→F30K		F30K→COCO	
`CrossAtt?`	R1	R5	R1	R5
✗	52.38	79.09	33.33	60.97
✓	**54.02**	**80.27**	**33.81**	**63.19**

Table 3: Study of different primitive encodings.

	F30K→F30K		F30K→COCO	
Modulation	R1	R5	R1	R5
`Replace`	52.84	79.79	32.63	61.61
`MLP`	52.92	79.89	33.39	61.41
`FiLM`	**54.02**	**80.27**	**33.81**	**63.19**

Table 4: Study of different modulators.

Compositional generalization. On the max compound divergence (MCD) split, COMPOSER outperforms baselines by a margin for both F30K and C30K trained models (shown as Table 2). To further characterize the performance on compositional generalization, we create 16 test splits on each dataset with different compound divergence (from 0.15 to 0.31, where 0.31 is the max CD) and present the results in Figure 4. With the increases of CD, we observe the performance of COMPOSER and ViLBERT decreases. Compared to ViLBERT, we observe that COMPOSER is relatively more robust to this distribution shift, as the relative performance improvement is increasing with CD increases.

5.3 Analysis and Ablation Study

We perform several ablation studies to analyze COMPOSER, and provide qualitative results to demonstrate the model's interpretability.

Is `CrossAtt` in primitive encoding useful? Table 3 compares variants of COMPOSER with and without `CrossAtt` for primitive encoding, and shows that `CrossAtt` improves all metrics in in-domain and cross-dataset evaluation.

Which modulator works better? We consider three modulators to combine input concepts with the syntax token embeddings for later composition, which are `Replace`, `MLP`, and `FiLM`. The `Replace` directly replaces the syntax embedding with the input concept embedding. This is an inferior approach by design as it ignores the relative

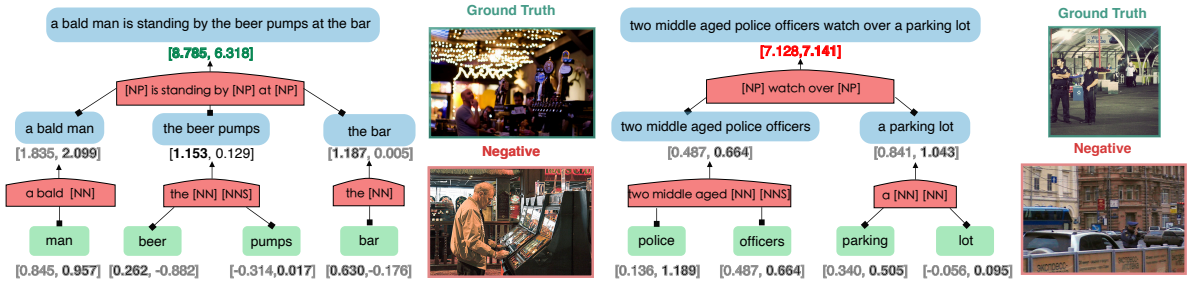


Figure 5: Interpreting the COMPOSER using visual-semantic alignment scores, formatted as $[s_{\texttt{GT}}, s_{\texttt{Negative}}]$. The left figure corresponds to a correct example, and the right figure corresponds to an incorrect one.

Method	F30K→F30K Sentence	F30K→F30K Phrase	F30K→COCO Sentence	F30K→COCO Phrase
▲ ViLBERT	50.94	18.34	30.50	15.00
+ **MVSA**	48.90	**23.55**	29.90	18.73
◆ COMPOSER	52.52	21.04	32.87	18.29
+ **MVSA**	**54.02**	22.70	**33.81**	**18.81**

Table 5: Comparison between ViLBERT and COMPOSER on multi-level visual-semantic alignment supervision (MVSA). All results are reported in R1.

PT	CT	F30K→F30K R1	F30K→F30K R5	F30K→COCO R1	F30K→COCO R5	# Param	FLOPS
2	5	51.72	79.71	33.67	60.38	$129M$	$35.40G$
4	5	53.32	79.73	**33.83**	61.61	$143M$	$37.11G$
6	5	**54.02**	80.27	33.81	**63.19**	$157M$	$38.40G$
6	3	47.92	76.85	25.74	51.45	$136M$	$29.40G$
6	1	34.47	62.61	21.25	43.86	$115M$	$19.98G$
ViLBERT		50.94	**80.86**	30.50	58.98	$235M$	$24.44G$

Table 6: Results on COMPOSER of different complexity. All results are reported in R1. (PT: Predicate Transformer, CT: Composition Transformer)

position of each concept. `MLP` model applies multi-layer neural networks on the concatenated syntax and input concept embeddings. `FiLM` model uses the syntax embedding to infer the parameter of an affine transformation, which is then applied to the input concepts. We show the results in Table 4. `Replace` achieves the worst performance, indicating the importance of identifying the position of input concepts. COMPOSER chooses `FiLM` as the modulator given its strong performance over all metrics.

Is MVSA supervision useful? We evaluate the influence of multi-level visual-semantic alignment on sentence and phrase to image retrieval. In the phrase-to-image experiments, we sample 5 non-sentence concepts from the CRG for each annotation in the corresponding test data and use them as the query to report results (in R1). Table 5 presents the results. With the MVSA, COMPOSER outperforms ViLBERT on both sentence and phrase-based retrieval by a noticeable margin, indicating the advantage of capturing mid-level alignment in our model design. Secondly, MVSA improves both COMPOSER and ViLBERT on the phrase to image retrieval over their counterparts. However, adding MVSA on ViLBERT leads to a degradation of sentence-to-image retrieval, showing that ViLBERT is incapable of mastering visual alignments for both sentences and phrases simultaneously. COMPOSER with MVSA improves itself on both sentence and phrase, showing strong multi-granular visual-semantic alignment ability.

Performance vs. complexity trade-off. We compare variants of COMPOSER with different parameter and computation budgets, which uses different numbers of layers for the Predicate Transformer (PT) and Composition Transformer (CT). The results are shown in Table 6. First, We keep the size of CT fixed and vary the size of PT. It shows a marginal performance decrease occurring as the # of layers of PT goes down. Then we keep the size of PT fixed and decrease the capacity of CT, which presents a significant performance drop, showing the essential role CT is playing. Besides having superior results, COMPOSER has *(at least 33%) fewer parameters* than the ViLBERT model, which indicates a potential performance gain could be achieved with a larger COMPOSER model.

For computation complexity, we observe that the full COMPOSER model is 50% less efficient to a ViLBERT model, due to its recursive nature. Meanwhile, we notice that the increase in the # of CT layers contributes a significant amount to the total computation time as every two additional layers adds $\sim 10G$ FLOPS.

	F30K→F30K		F30K→COCO	
Pruning Probability	R1	R5	R1	R5
Un-pruned	**54.02**	**80.27**	**33.81**	**63.19**
Probability=0.1	49.12	76.92	31.12	58.86
Probability=0.3	48.46	76.60	30.40	58.16
Probability=0.5	47.44	76.24	30.38	57.62

Table 7: Performance under different parsing qualities.

Performance under different parsing qualities. CRG is generated based on constituent parser. We investigate the performance of COMPOSER with CRG under different parsing qualites. Given a parsing tree, We randomly remove its branches randomly with a probability of 0.1, 0.3, or 0.5 to generate a tree with degraded parsing quality. We evaluate COMPOSER on the resulting CRGs. We summarized the results in Table 7. When parsing quality drops, both in-domain and cross-dataset transfer performance drops. The performance degrades by 12%, when half of the parse could be missing. We expect with better parsing quality, COMPOSER can achieve stronger performance.

Interpreting COMPOSER's decision. Despite the solid performance, COMPOSER is also highly interpretable. Specifically, we visualize its alignment scores along with the concept composition procedure in Figure 5. Empirically, we observe that most failures are caused by visually grounding mistakes at the primitive concepts level. The error then propagates "upwards" towards concept composition.

For instance, the left example shows that COMPOSER is confusing between the ground truth and negative image when only the text of shared visual concept "a bold man" is presented. With more information are given, it gets clarified immediately as it notices that the target sentence is composed not only with the above subject, but also with the prepositional phrases "by the beer pumps at the bar" that reflects the visual environment.

Scalability to full COCO dataset. Finally, we trained our model (PT=6, CT=5) on the full COCO training split and evaluated for both in-domain and cross-dataset transfer task. We use the same hyperparameters as C30K. However, COMPOSER underperforms the ViLBERT in this setting, as it achieves 56.06% and 44.24% in R1 for the in-domain task (COCO→COCO) and cross-dataset evaluation tasks (COCO→F30k), while ViLBERT obtains 56.83% and 46.62%, respectively. We hypothesize that this negative result is largely due to the limited model capacity of the proposed COMPOSER, as it has relatively 33% less parameters comparing to ViLBERT. Meanwhile, it is also observed that COMPOSER performs worse than ViLBERT in fitting training data. We observe that doubling the training epoch would increase both in-domain and out-of-domain performance by 2% relatively. Increasing the layer of Composition Transformer (CT) to 7 would also improves R1 by 2.5% relatively. Further scaling up COMPOSER may resolve this issue but requires more computational resources, and we leave this for future research.

6 Conclusion

In this paper, we propose the concept and relation graph (CRG) to explore the compositional structure in visually grounded text data. We further develop a novel concept composition neural network (COMPOSER) on top of the CRG, which leverages the explicit structure to compose concepts from word-level to sentence-level. We conduct extensive experiments to validate our model on image-text matching benchmarks. Comparing with prior methods, COMPOSER achieves significant improvements, particularly in zero-shot cross-dataset transfer and compositional generalization. Despite these highlights, there are also many challenges that COMPOSER does not address in the scope of this paper. First, it requires high-quality parsing results to achieve strong performances, which may not be readily available in languages beyond English. Moreover, similar to other recursive neural networks, COMPOSER is also computationally resource demanding, which sets a limit to its scalability to large-scale data.

Acknowledgements

This work is partially supported by NSF Awards IIS-1513966/ 1632803/1833137, CCF-1139148, DARPA Award#: FA8750-18-2-0117, FA8750-19-1-0504, DARPA-D3M - Award UCB-00009528, Google Research Awards, gifts from Facebook and Netflix, and ARO# W911NF-12-1-0241 and W911NF-15-1-0484. We thank anonymous EMNLP reviewers for constructive feedback. Additionally, we would like to thank Jason Baldbridge for reviewing an early version of this paper, and Kristina Toutanova for helpful discussion.

References

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-up and top-down attention for image captioning and visual question answering. In *CVPR*.

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. 2016. Neural module networks. In *CVPR*.

Dzmitry Bahdanau, Shikhar Murty, Michael Noukhovitch, Thien Huu Nguyen, Harm de Vries, and Aaron Courville. 2019. Systematic generalization: What is required and can it be learned? In *ICLR*.

Battista Biggio, Blaine Nelson, and Pavel Laskov. 2011. Support vector machines under adversarial label noise. In *ACML*.

Devendra Singh Chaplot, Kanthashree Mysore Sathyendra, Rama Kumar Pasumarthi, Dheeraj Rajagopal, and Ruslan Salakhutdinov. 2018. Gated-attention architectures for task-oriented language grounding. In *AAAI*.

Jiacheng Chen, Hexiang Hu, Hao Wu, Yuning Jiang, and Wang Changhu. 2021. Learning the best pooling strategy for visual semantic embedding. In *CVPR*.

Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C Lawrence Zitnick. 2015. Microsoft coco captions: Data collection and evaluation server. *arXiv preprint:1504.00325*.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020. Uniter: Learning universal image-text representations. *ECCV*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

Fartash Faghri, David J Fleet, Jamie Ryan Kiros, and Sanja Fidler. 2017. Vse++: Improved visual-semantic embeddings. In *BMVC*.

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 *ACL*.

Ruifang Ge and Raymond Mooney. 2009. Learning a compositional semantic parser using an existing syntactic parser. In *ACL-IJCNLP*.

Karl Moritz Hermann, Felix Hill, Simon Green, Fumin Wang, Ryan Faulkner, Hubert Soyer, David Szepesvari, Wojciech Marian Czarnecki, Max Jaderberg, Denis Teplyashin, et al. 2017. Grounded language learning in a simulated 3d world. *arXiv preprint arXiv:1706.06551*.

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 *CVPR*.

Andrej Karpathy and Li Fei-Fei. 2015. Deep visual-semantic alignments for generating image descriptions. In *CVPR*.

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 *ICLR*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint :1412.6980*.

Ryan Kiros, Ruslan Salakhutdinov, and Richard S Zemel. 2014. Unifying visual-semantic embeddings with multimodal neural language models. *NeurIPS Workshop Deep Learning*.

Nikita Kitaev and Dan Klein. 2018. Constituency parsing with a self-attentive encoder. In *ACL*.

Brenden Lake and Marco Baroni. 2018. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *ICML*.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *ICCV*.

Jiayuan Mao, Chuang Gan, Pushmeet Kohli, Joshua B Tenenbaum, and Jiajun Wu. 2019. The neuro-symbolic concept learner: Interpreting scenes, words, and sentences from natural supervision. In *ICLR*.

Ishan Misra, Abhinav Gupta, and Martial Hebert. 2017. From red wine to red tomato: Composition with context. In *CVPR*.

Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. 2018. Film: Visual reasoning with a general conditioning layer. In *AAAI*.

Sandro Pezzelle and Raquel Fernández. 2019. Is the red square big? malevic: Modeling adjectives leveraging visual contexts. In *EMNLP*.

Laura Ruis, Jacob Andreas, Marco Baroni, Diane Bouchacourt, and Brenden M Lake. 2020. A benchmark for systematic generalization in grounded language understanding. In *NeurIPS*.

Peter Shaw, Ming-Wei Chang, Panupong Pasupat, and Kristina Toutanova. 2020. Compositional generalization and natural language variation: Can a semantic parsing approach handle both? In *ACL*.

Haoyue Shi, Jiayuan Mao, Kevin Gimpel, and Karen Livescu. 2019. Visually grounded neural syntax acquisition. *ACL*.

Hao Tan and Mohit Bansal. 2019. LXMERT: Learning cross-modality encoder representations from transformers. In *EMNLP*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NeurIPS*.

Yu-Siang Wang, Chenxi Liu, Xiaohui Zeng, and Alan Yuille. 2018. Scene graph parsing as dependency parsing. *NAACL*.

Kexin Yi, Jiajun Wu, Chuang Gan, Antonio Torralba, Pushmeet Kohli, and Josh Tenenbaum. 2018. Neural-symbolic VQA: Disentangling reasoning from vision and language understanding. In *NeurIPS*.

Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. 2014. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *TACL*, 2:67–78.

Bowen Zhang, Hexiang Hu, Vihan Jain, Eugene Ie, and Fei Sha. 2020. Learning to represent image and text with denotation graph. In *EMNLP*.

Appendix

In the Appendix, we provide details omitted from the main text due to the limited space, including:

- § A describes the implementation details for extracting primitives & predicates from the constituency tree (§ 2 of the main text).
- In § B, we describes the details of generating the compositional evaluation splits (§ 5.1 of the main text).
- § C contains training and architecture details for COMPOSER and baselines (§ 5.1 of the main text).
- § D includes the ablation studies on learning objectives and margin of MVSA (§ 5.2 of the main text).

A Extracting Primitives & Predicates from the Constituency Tree

As mentioned in the main paper, we parse the sentence and convert it into a tree of concepts and primitives. Particularly, we first perform constituency parsing using the self-attention parser (Kitaev and Klein, 2018). Table 8 provides the visualization for two examples of the syntax sub-trees. Next, we perform a tree search (*i.e.*, breadth-first search) on the constituency tree of the current input concept to extract the sub-concepts and predicate functions. Note that this step is applied recursively until we can no longer decompose a concept into any sub-concepts. On a single step of the extraction, we enumerate each node in the constituency tree of current input text expression and examine whether a constituent satisfies the criterion that defines the visually grounded concept.

The concept criterion defined for the Flickr30K and COCO dataset contains several principles: (1) If the constituent is a word, it is a primitive concept if its Part-of-Speech (POS) tag is one of the following: {`[NN]`,`[NNS]`,`[NNP]`,`[NNPS]`}; (2) If the constituent is a phrase (with two words or more), it would be a concept when this constituent contains a primitive word (*i.e.*, satisfying condition (1)) and its constituency tag is one of the following: {`[S]`, `[SBAR]`, `[SBARQ]`, `[SQ]`, `[SINV]`, `[NP]`, `[NX]`}. After all the concepts are extracted, we take the remaining words in the current input text expression as the predicate that combines those concepts and use the tag to represent syntactic blank. Concrete examples can be found in the Table 8. For instance, in the first example, we search the text "two dogs are running on the grass" and extract two noun constituents, "two dogs" and "the grass" as the concepts. We use the remaining text "`[NP]` is running on `[NP]`" as the predicate that indicates the semantic meaning of how these two sub-concepts composes into the original sentence.

B Details on Generation of Compositional Evaluation Splits

As mentioned in the main text, we generate compositional generalization (CG) splits with 1,000 images and 5,000 text queries, maximizing the Compound Divergence (MCD) as Shaw et al. (2020)[3], to assess models' capability in generalizing to the data with different predicate distribution. Concretely, we select Flickr30K training data to generate the F30K-MCD split. First, we remove all F30K test data that has unseen primitive concepts to the COCO training data. Next, we collect and count the predicates for each image among all the remaining data over the five associated captions. These predicates correspond to the "compounds" defined in (Keysers et al., 2020; Shaw et al., 2020), and the objective is to maximize the divergence between compound distribution of the evaluation data to the training data. As a result of this step, we end up with a data set formed with pairs of (image, predicates counts), which are then used for computing the overall compound divergence ($\texttt{CD}_{\text{ALL}}$) to the training dataset. Afterwards, we enumerate over each pair of data, and again compute the compound divergence to the training dataset but with this specific data is removed. We denote the change of compound divergence as $\Delta_i = \texttt{CD}_i - \texttt{CD}_{\text{ALL}}$, and use it as an additional score to associate every data. Finally, we sort all the data with regard to the difference of compound divergence Δ_i, and use the top ranking one thousand examples as the maximum compound divergence (MCD) split. The process for generating the COCO-MCD split is symmetrical to the above process, except the data is collected from COCO val+test splits (as it is sufficiently large). Similarly, to generate different CDs for making Figure 4 of the main text, we can also make use of the above data sorted by Δ_i. Concretely, we put a sliding window with 1,000 examples and enumerate over the sorted data to obtain a massive combination of data (we can take a

[3]We adopt the released code here for the computing compound divergence: https://github.com/google-research/language/tree/master/language/nqg/tasks

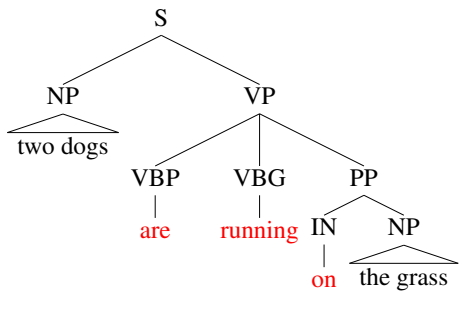
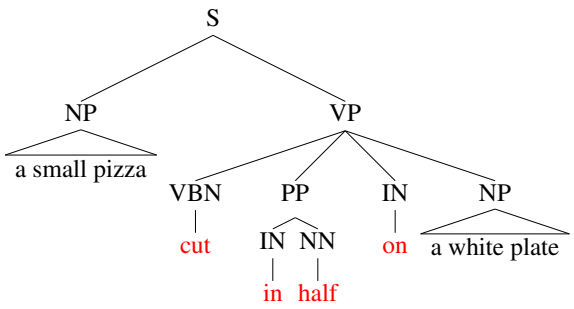
Syntax Tree		
Concept	`two dogs are running on the grass`	`a small pizza cut in half on a white plate`
Predicate	`[NP] are running on [NP]`	`[NP] cut in half on [NP]`
Sub-Concepts	`NP1="two dogs"` `NP2="the grass"`	`NP1="a small pizza"` `NP2="a white plate"`

Table 8: Explanatory example of extracting predicates and sub-concepts from a concept

stride to make this computation sparser.) For each window of data, we measure the compound divergence and only take the windows that are at the satisfaction to our criteria. In Figure 4, we keep the windows that has the closest CD values to desired X-axis values for plotting.

C Implementation Details of COMPOSER and Baselines

Visual feature pre-processing We follow ViLBERT (Lu et al., 2019) that extracts the patch-based ResNet feature using the Bottom-Up Attention model. The image patch feature has a dimension of 2048. A 5-dimension position feature that describes the normalized up-top and bottom-down position is extracted alongside the image patch feature. Therefore, each image region is described by both the image patch feature and the position feature. We extracted features from up to 100 patches in one image.

Text pre-processiong Following BERT (Devlin et al., 2019), we tokenize the text using the uncased WordPiece tokenizer. Specifically, we first lowercase the text and use the uncased tokenizer to extract tokens. The tokenizer has a vocabulary size of 30,522. The tokens are then transformed into word embeddings with 768 dimensions. Besides the word embedding, a 768-dimension position embedding is extracted. Both position embedding and word embedding are added together to represent the embedding of tokens.

Training details We use Adam optimizer (Kingma and Ba, 2014) to optimize the parameter of our model. All the models are trained with a mini-batch size of 64. We employ a warm-up training strategy as suggested by ViLBERT (Lu et al., 2019). Specifically, the learning rate is linearly increasing from 0 to $4e-5$ in the first 2 epochs. Then the learning rate decays to $4e-6$ and $4e-7$ after 10 epochs and 15 epochs, respectively. The training stopped at 20 epochs.

Detials of baseline approaches. The text encoder for both models contains 12 layers of transformers and is initialized from BERT pretrained model using the checkpoint provided by HuggingFace. For ViLBERT, we use the `[CLS]` embedding from the last layer as text representation $\boldsymbol{y}$. We use the average of contextualized text embedding from the last layer as $\boldsymbol{y}$ in the VSE model. The visual encoder of VSE contains an MLP model with the residual connection. It transforms the image patch feature into a joint image-text space. The output of the visual encoder is the mean of the transformed image patch features. Unlike VSE, ViLBERT contains 6 layers of transformers for the image encoder and 6 layers of the cross-modal transformer to model the text and image features jointly. We use the embedding of `[V-CLS]` token from the last layer of the image encoder as the image feature $\boldsymbol{x}$.

Details of COMPOSER. The composer contains four primary learning sub-modules: (1) the `CrossAtt` model in primitive encoding; (2) the

Predicate Transformer (PT) model; (3) the modulator; (4) the Composition Transformer (CT). The details of this sub-modules are list as what follows:

- **Primitive encoding.** We implement the `CrossAtt` model as a one-layer multi-head cross-modal Transformer that contains 768 dimension with 12 attention heads. The query set Q is the sub-word token embeddings of the primitive word, and the key and value set K and V are the union of sub-word token embeddings and the object-centric visual features (which is linearly transformed to have the same dimensionality). We use the average of the contextualized sub-word token embeddings as the final primitive encoding.
- **Predicate Transformer (PT).** We use 6 layers text Transformers with 768 hidden dimension and 12 attention heads to instantiate the Predicate Transformer. This network is initialized with the first 6 layers of a pre-trained BERT model.
- **Modulator.** We use FiLM (Perez et al., 2018) as the modulator. Specifically, it contains two MLP models with a hidden dimension size of 768 to generate the scale $\boldsymbol{a}$ and bias vectors $\boldsymbol{b}$, using the syntactic placeholders as input. The scale $\boldsymbol{a}$ and bias $\boldsymbol{b}$ are then used to transform the input concept embedding $\boldsymbol{c}$ as $\boldsymbol{a} \odot \boldsymbol{c} + \boldsymbol{b}$. Here $\odot$ represents the element-wise multiplication. This modulated concept embedding is then projected by another MLP with 768 hidden dimensions, and used for reassembling with the predicate sequence.
- **Composition Transformer (CT).** We follow the architecture of ViLBERT (Lu et al., 2019) to design the Composition Transformer (shown in Figure 6). Specifically, it has interleaved `SelfAtt` Transformer and `CrossAtt` Transformer in the network. For example, if we consider a three-layer Composition Transformer, we have a `SelfAtt` Transformer at the beginning for both modality, followed with a `CrossAtt` Transformer that interchanges the information between the modality, and then another `SelfAtt` Transformer that only operates on the text modality. The output embedding of this last text `SelfAtt` Transformer is then used for computing the visual-semantic alignment scores using the linear regressor $\boldsymbol{\theta}$. Thus, when we consider shallower or deeper network, we add or remove the two layers of interleaved `SelfAtt` and `CrossAtt` Transformers. The hidden dimension of `SelfAtt` Transformer is 768, and there is 12 attention heads. The hidden dimension of `CrossAtt` Transformer is 1024, and there is 8 attention heads.

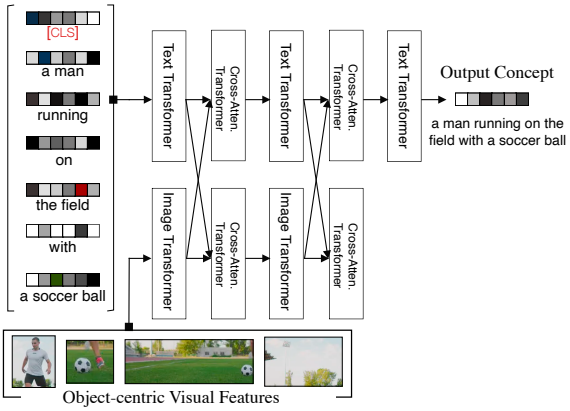


Figure 6: Details of the Composition Transformer model.

α	β	F30K→F30K R1	F30K→F30K R5	F30K→COCO R1	F30K→COCO R5
COMPOSER w/ different α					
0.4	0.2	53.54	80.51	33.73	61.67
0.6	0.2	53.44	80.21	33.89	61.05
0.8	0.2	54.02	80.27	33.81	63.19
COMPOSER w/ different β					
0.8	0	53.66	80.39	33.33	61.15
0.8	0.2	54.02	80.27	33.81	63.19
0.8	0.4	53.50	80.55	33.87	61.15

Table 9: Ablation Study on COMPOSER with Different Margin for MVSA and Order Objectives.

D Additional Experiments on COMPOSER

We report additional ablation studies that are omitted in the main paper due to space limitation. In this section, we study COMPOSER performance under different MVSA objectives, Negative Log-Likelihood and Hinge loss. Then we study COMPOSER performance under different margins of MVSA and Order objectives.

MVSA Objective. The MVSA objectives can be implemented using NLL loss or Hinge loss. We study the performance of COMPOSER under different losses for MVSA in Table 10. The models are trained with both MVSA and order objectives. We set the margin of order objectives $\beta = 0.2$. For the hinge loss, we set the margin $\alpha = 0.8$. COMPOSER trained with hinge loss in MVSA achieves better performance than the NLL loss in all metrics

	F30K→F30K		F30K→COCO	
Loss Function	R1	R5	R1	R5
NLL	52.42	79.47	33.41	62.17
Hinge Loss	54.02	80.27	33.81	63.19

Table 10: Ablation Study MVSA Objective: Comparing NLL to Hinge Loss.

across both in-domain and cross-dataset generalization settings. Therefore, for all the experiments training with MVSA, we use hinge loss instead.

Ablation study on α and β. We study COMPOSER performance on the different margin of MVSA and Order objectives. First, we fix the margin of order objectives β and tune the margin for MVSA α. COMPOSER with a larger margin for MVSA achieves better R1 in-domain performance. Alternatively, by fixing the α and tuning β, COMPOSER achieves the best R1 in-domain performance and best R5 in cross-dataset generalization setting with $\beta = 0.2$.