

---

# LEARNING TO COMPOSE SOFT PROMPTS FOR COMPOSITIONAL ZERO-SHOT LEARNING

Nihal V. Nayak\*, Peilin Yu\*, Stephen H. Bach

Department of Computer Science

Brown University

Providence, RI 02906, USA

{nnayak2, pyu12, sbach}@cs.brown.edu

## ABSTRACT

We introduce compositional soft prompting (CSP), a parameter-efficient learning technique to improve the zero-shot compositionality of large-scale pretrained vision-language models (VLMs) without the overhead of fine-tuning the entire model. VLMs can represent arbitrary classes as natural language prompts in their flexible text encoders but they underperform state-of-the-art methods on compositional zero-shot benchmark tasks. To improve VLMs, we propose a novel form of soft prompting. We treat the attributes and objects that are composed to define classes as learnable tokens of vocabulary and tune them on multiple prompt compositions. During inference, we recombine the learned attribute-object vocabulary in new combinations and show that CSP outperforms the original VLM on benchmark datasets by an average of 14.7 percentage points of accuracy. CSP also achieves new state-of-the-art accuracies on two out of three benchmark datasets, while only fine-tuning a small number of parameters. Further, we show that CSP improves generalization to higher-order attribute-object compositions and combinations of pretrained attributes and fine-tuned objects. The code is available at <https://github.com/BatsResearch/csp>.

## 1 INTRODUCTION

Compositionality is the long-standing goal of artificial intelligence of creating new concepts by combining existing primitive concepts [5, 8, 14, 19, 29]. The practical advantage of compositionality for deep neural networks lies in the ability to build new classifiers by combining existing classifiers. In this work, we consider compositional zero-shot learning, a classification task where the model learns to predict unseen or novel compositions of the classes [31, 32, 35]. Since we have innumerable combinations of classes, the problem is made tractable by considering attribute-object compositions such as `old tiger` and `young tiger` where `old`, `young`, and `tiger` are primitive concepts. Attributes are visual concepts understood by humans and shared across categories. For example, the attribute `old` can be used to describe `old tiger` as well as `old cat` where `tiger` and `cat` are the object categories.

Existing methods for compositional zero-shot learning typically map attributes and objects to pretrained word embeddings and use a pretrained image encoder backbone to jointly align the image and the attribute-object text representations to learn compositionality [25, 27, 28, 30, 31, 32, 35]. However, the pretraining of the word embeddings and image encoder is disjoint and isolated from each other, i.e., these methods learn to align image and text representations from scratch. These task-specific architectures also are limited in flexibility. For example, to adapt these methods to higher-order compositions with multiple attributes and objects such as `small furry cat` or `old white tiger`, the original architecture needs to be modified. The ability to generalize beyond the original training length is a key test for compositionality [14].

In contrast, we propose to build on large-scale pretrained vision-language models (VLMs), which are trained on massive amounts of aligned images and text [16, 17, 24, 38]. We focus on CLIP

---

\*Equal contribution

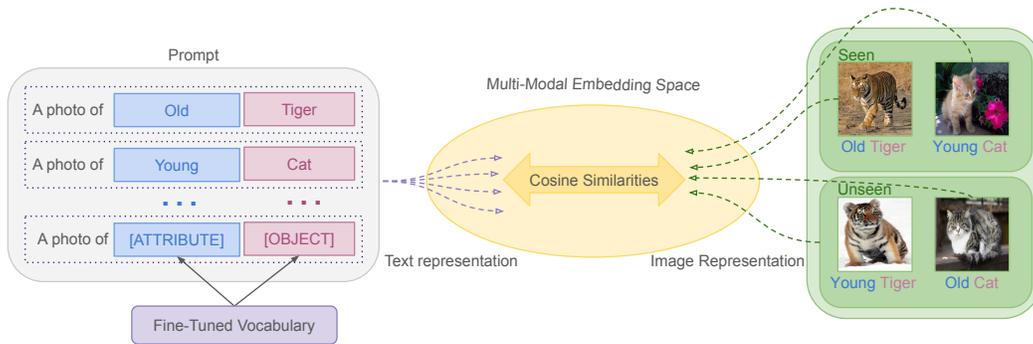


Figure 1: An overview of compositional zero-shot learning with CSP. We fine-tune the vocabulary for attributes and objects on the seen classes. Then we compose novel soft prompts to test on the unseen classes.

[38], a powerful vision-language model pretrained on 400 million image-text pairs. CLIP has two main components: the image encoder and the text encoder that produce vector representations for images and text in a multi-modal embedding space. The text encoder accepts a textual input, or a prompt such as `A photo of dog` to produce a vector representation for the class `dog`. Taking the cosine similarity with all the class prompts and the image, we get a compatibility score for the classes and pick the one with the highest score. However, our experiments show that CLIP without any fine-tuning underperforms state-of-the-art methods (Section 5.5).

To improve VLMs for compositional zero-shot learning, we introduce compositional soft prompting (CSP), a parameter-efficient learning technique that avoids the overhead of fine-tuning the entire model. Fine-tuning large pre-trained models such as CLIP requires huge amounts of compute that may be inaccessible to the community at large. This limitation of large pre-trained models has motivated several soft prompting techniques in both vision and language [21, 36, 45, 50]. These works tune a single prompt on a downstream supervised task, often in a few-shot setting. For instance, they typically represent the prompts as `A photo of [class]` and tune a single prefix `A photo of` on the entire dataset. In contrast, CSP is a novel way of soft prompting. We treat the attributes and objects that are composed to define classes as learnable tokens of vocabulary in a prompt as `A photo of [attribute] [object]`. We tune on multiple `[attribute]` and `[object]` prompt compositions, and then we recombine them into new prompts for zero-shot inference (Figure 1).

Our results show that CSP improves over the zero-shot performance of CLIP. CSP significantly improves over CLIP across three benchmark datasets by an average accuracy of 17.7 percentage points in the closed-world setting and 11.7 percentage points in the open-world setting (using the harmonic mean metric). CSP also achieves new state-of-the-art accuracies on two out of the three benchmark datasets in both the closed-world and open-world settings. We improve over the previous state-of-the-art methods on MIT-States by 4.7 points and C-GQA by 3.4 points on the harmonic mean metric in the closed world setting. In the open-world setting, we show improvement of 1.3 points on MIT-States and 1.2 points on C-GQA on the harmonic mean metric.

In addition to good accuracy, CSP has several other advantages. CSP fine-tunes orders-of-magnitude fewer parameters than existing work on compositional zero-shot learning since only the vocabulary is tuned. We show that training CSP with attribute-object compositions improves CLIP’s performance on attribute-attribute-object compositions without any changes or explicit training. We also show that CSP improves generalization to compositions of *unseen* attributes and seen objects. Prior work on compositional zero-shot learning typically only evaluates unseen compositions of seen attributes and seen objects.

In summary, we make the following contributions:

1. We introduce compositional soft prompting (CSP), a parameter-efficient learning technique to improve the compositionality of large-scale vision-language models (VLMs). The at-

- 
- tributes and objects that are composed to define classes are treated as learnable tokens of vocabulary. Unlike existing work on soft prompting, our prompts are tuned on multiple prompt compositions and then recomposed into new combinations for zero-shot inference.
2.  $\mathcal{CSP}$  improves the harmonic-mean accuracy of CLIP by an average of 14.7 percentage points across three benchmark datasets. We also report new state-of-the-art accuracies for MIT-States [15] and C-GQA [27] in the closed-world and open-world setting.
  3. We conduct additional experiments to analyze  $\mathcal{CSP}$ . We show that fine-tuning with attribute-object compositions improves CLIP’s performance on attribute-attribute-object compositions without any changes or explicit attribute-attribute-object supervision. We also show that  $\mathcal{CSP}$  learns to generalize to compositions of pretrained and fine-tuned vocabulary.

## 2 RELATED WORK

We describe the related work in compositional zero-shot learning and prompting.

### 2.1 COMPOSITIONAL ZERO-SHOT LEARNING

The growing interest in compositional zero-shot learning has contributed to several architectural innovations [25, 27, 28, 30, 31, 32, 35]. Early works compose attributes and objects with a transformation function [30, 32]. Recent work uses separate encoding layers for attributes and objects, and then combines them with late fusion using a linear layer or a multilayer perceptron [35]. The most successful methods represent the attribute and object relationship in a graph and learn their compositions via graph convolutional networks [18, 28, 31, 39, 47]. However, prior work learns to align the image and the attribute-object representations in the multi-modal embedding space from scratch. Instead, we build on CLIP, a vision-language model pretrained on image-text pairs. Recently, Radenović et. al. [37] pretrain an attribute-object model with large amounts of weakly labeled data, but their model is not publicly available.

Compositional zero-shot learning is also closely related to the broader goal of compositionality in artificial intelligence [5, 8, 14, 19, 29]. For more details, refer to [14]. Lastly, the compositional zero-shot learning task is a form of zero-shot learning [7, 48] where the classes are described using attribute-object pairs. Notable works in zero-shot learning include object classification [20], semantic segmentation [22], video action-recognition[9], and more [46].

### 2.2 PROMPTING

Prompting is a recent focus in the vision and language communities that has shown benefits in zero-shot and few-shot performance on a wide range of tasks [1, 2, 3, 21, 38, 36, 40, 45, 50]. Discrete prompts are typically hand-written text input that provide guidelines to large pre-trained models such as CLIP, GPT-3 [3], etc. for inference without updating the model parameters. While manually engineering prompts can help achieve better performance, it is often time-consuming and impractical to find the best prompt.

Soft prompting is an alternative to discrete prompts, where a part of the prompt is learned by back-propagating without fine-tuning the entire model [21, 26, 45, 50]. Several works using soft prompts show improved downstream performance compared to hand-crafted prompts [21, 23, 36, 41, 45, 50]. In all these works, soft prompts are a single input for the entire task. In contrast, we learn multiple prompt compositions for compositional zero-shot learning. We recombine them in new combinations to represent unseen classes for zero-shot inference. We show in Section 5.5 that traditional soft prompting can also improve CLIP on compositional zero-shot learning, but generally not as much as compositional soft prompting.

## 3 PRELIMINARIES

In this section, we formally introduce compositional zero shot-learning and large-scale pretrained vision-language models (VLMs). In particular, we describe the architecture of CLIP and how to use it for compositional zero-shot learning.

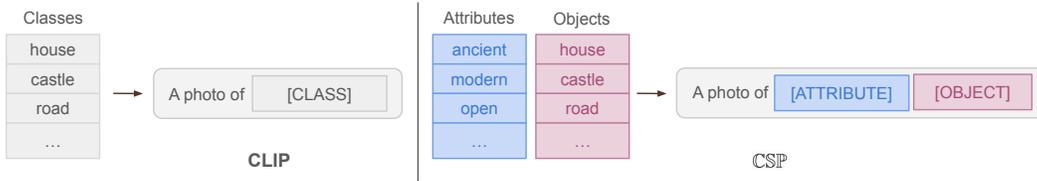


Figure 2: Comparison of the vocabulary composition process for prompting CLIP in a traditional zero-shot setting and CSIP in a compositional zero-shot setting.

### 3.1 PROBLEM SETUP

Here we formally define the task of compositional zero-shot learning. Let  $\mathbb{A} = \{a_0, a_1, \dots, a_n\}$  be the set of possible attributes and  $\mathbb{O} = \{o_0, o_1, \dots, o_m\}$  be the set of possible object categories. Let the label space  $\mathbb{Y}$  be the Cartesian product of the attribute set and the object category set,  $\mathbb{Y} = \mathbb{A} \times \mathbb{O}$ . We are given two disjoint label subsets such that  $\mathbb{Y}_{seen} \subset \mathbb{Y}$ ,  $\mathbb{Y}_{unseen} \subset \mathbb{Y}$ , and  $\mathbb{Y}_{seen} \cap \mathbb{Y}_{unseen} = \emptyset$  where  $\mathbb{Y}_{seen}$  and  $\mathbb{Y}_{unseen}$  are the set of the seen and unseen classes. At training time, we are given examples  $\mathbb{S}_{seen} = \{(x_1, y_1), \dots, (x_n, y_n)\}$  to learn some discriminative model  $f: \mathbb{X} \rightarrow \mathbb{Y}_{seen}$ .

During inference, we want the model to predict both seen and unseen classes in the test set, i.e.,  $f: \mathbb{X} \rightarrow \mathbb{Y}_{test}$ . In the closed-world evaluation, the test set is defined as  $\mathbb{Y}_{test} = \mathbb{Y}_{seen} \cup \mathbb{Y}_{unseen}$ . In the open-world evaluation, the model has to predict all possible permutations of the attribute-object compositions, i.e.,  $\mathbb{Y}_{test} = \mathbb{Y}$  and  $\mathbb{Y}_{unseen} = \mathbb{Y} - \mathbb{Y}_{seen}$ . For more details, see Section 5.2.

### 3.2 CLIP FOR COMPOSITIONAL ZERO-SHOT INFERENCE

VLMs have several variants depending on the task [10, 42, 43, 44]. In our experiments, we use Contrastive Language-Image Pre-Training (CLIP) [38]. CLIP is a large-scale vision-language model pretrained using approximately 400M text-image pairs sourced from the internet with a contrastive learning objective. The CLIP architecture has two key components: the image encoder and the text encoder. The image encoder is either a convolutional neural network [11] or vision transformers (ViTs) [6]. The text encoder consists of multiple transformer encoder layers. Both the components are trained jointly by optimizing the contrastive loss aligning images with their associated captions.

To use CLIP for zero-shot inference, the classes are transformed into natural language prompts such as `A photo of [class]` (Figure 2, left). The prompts are passed through the tokenizer to get tokens for each word in the prompt. Next, the embedding function maps the tokens to the vocabulary. Then, they are passed through the text encoder to compute the text representations. Similarly, the representation of a query image comes from the image encoder. We then take the cosine similarities between the query image representation and the text representations to get the final prediction.

Formally, let  $VLM_{\mathbf{T}}$  be the pretrained text encoder and  $VLM_{\mathbf{V}}$  be the pretrained image encoder in the vision-language model. Consider some image  $x$  with label set  $\mathbb{Y}$  where  $|\mathbb{Y}| = m$ . For each label  $y_j \in \mathbb{Y}$ , we have a tokenized prompt  $t_i$ . The tokenized prompt is passed through the embedding function  $\xi(\cdot)$  to map the tokens to the their vocabulary.

Finally, we compute the inference probability as follows:

$$\mathbf{x}_v = \frac{VLM_{\mathbf{V}}(x)}{\|VLM_{\mathbf{V}}(x)\|} \quad \mathbf{t}_i = \frac{VLM_{\mathbf{T}}(\xi(t_i))}{\|VLM_{\mathbf{T}}(\xi(t_{a,o}))\|} \quad (1)$$

$$p\left(\frac{y=i}{x}\right) = \frac{\exp(\mathbf{x}_v \cdot \mathbf{t}_i / \tau)}{\sum_{j=1}^m \exp(\mathbf{x}_v \cdot \mathbf{t}_j / \tau)} \quad (2)$$

where  $\mathbf{x}_v$  is the normalized image representation,  $\mathbf{t}_i$  is the normalized text representation, and  $\tau$  is a temperature parameter.

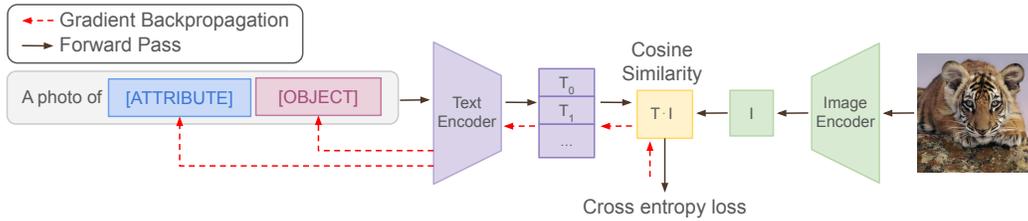


Figure 3: Training pipeline for CSP. The prompt with the attribute and object vocabulary is passed through the text encoder to get the text representation. Similarly, the example is passed through the image encoder for the image representation. Next, we take the cosine similarity for all the prompts with the image and compute the cross entropy loss. Finally, we backpropagate the loss through the text encoder and update the attribute and object vocabulary weights.

We can easily adapt CLIP to compositional zero-shot learning by changing the prompt style. Instead of using the class prompt format `a photo of [class]`, we use `a photo of [attribute] [object]` the candidate compositional classes. The change in prompt format allows us to represent pair of attribute and objects such as `a photo of young cat` in the text encoder without any changes. However, in Section 5.5, we see that this naive application of CLIP underperforms state-of-the-art methods on compositional zero-shot learning benchmark datasets.

## 4 COMPOSITIONAL SOFT PROMPTING

In this section, we introduce CSP, a parameter-efficient learning technique for fine-tuning large pre-trained models for better compositionality. We describe CSP, its training and inference procedures, and how to apply it in open-world evaluations. The goal of CSP is to improve VLMs such as CLIP on tasks that require composing concepts in novel ways. In Section 5.5, we show that CLIP does not perform well on compositional zero-shot learning benchmarks as state-of-the-art-methods. This is perhaps related to the fine-grained supervision on attributes and objects available in these benchmarks, which were likely not present when CLIP was pretrained on data crawled from the Web. To combine the benefits of CLIP’s pretraining and flexible architecture with the additional supervision for compositional zero-shot learning, it would be ideal to fine-tune the model. However, fine-tuning CLIP requires huge amounts of compute that may be inaccessible to the community at large.

To that end, we introduce CSP, a parameter-efficient learning technique to improve the compositionality of a large-scale pretrained vision language model without the overhead of fine-tuning the model. CSP treats the attributes and objects that are composed to define classes as learnable tokens of vocabulary and tune them on multiple prompt compositions. We tune multiple prompt compositions on the labeled dataset with attribute and object compositions as described in Section 3.1. Training and inference with CSP is very simple as we only need to swap the vocabulary of the attributes and objects for any desired composition in the prompt (Figure 2, right). As a result, our method tunes only  $|\mathbb{A} \cup \mathbb{O}| \times d$  parameters where  $d$  is the dimension of the vocabulary embedding. This is a novel form of soft prompting, because prior work [12, 21, 36] has only tuned a single prompt prefix, whereas we composed learned prompt tokens in new ways at test time.

### 4.1 TRAINING

Figure 3 shows the overall learning process for CSP. We train the composed soft prompts by fine-tuning the attribute and object vocabulary on the training dataset. First, we initialize the learnable vocabulary  $\theta = [\theta_{\mathbb{A}}; \theta_{\mathbb{O}}]$  where  $\theta \in \mathbb{R}^{|\mathbb{A} \cup \mathbb{O}| \times d}$ . We initialize  $\theta$  with pretrained embeddings from CLIP. Next, we construct a prompt with the attribute and object composition, tokenize, and map them to the vocabulary:

$$\xi(t_{a,o}) = \{\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_p, \mathbf{x}_a, \mathbf{x}_o\} \quad (3)$$

where  $\mathbf{x}_i \in \mathbb{R}^d$  has two parts:  $\{\mathbf{x}_0 \dots \mathbf{x}_p\}$  is the prefix context and the  $\mathbf{x}_a$  and  $\mathbf{x}_o$  is the attribute and object vocabulary for the composition  $(a, o)$ . The prompt format in our work is `a photo of`

[attribute] [object] where a photo of is the prefix context. We replace the pretrained attribute and object vocabulary with our learnable attribute and object embeddings in prompt to get a photo of [attribute] [object] as follows:

$$\psi^{\text{CSP}}(t_{a,o}) = \xi^{\text{CSP}}(\xi(t_{a,o})) = \{\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_p, \boldsymbol{\theta}_a, \boldsymbol{\theta}_o\} \quad (4)$$

where  $\xi^{\text{CSP}}$  is the embedding function to replace pretrained vocabulary with learnable parameters  $\boldsymbol{\theta}_a$  and  $\boldsymbol{\theta}_o$  for the attribute and the object in the prompt. For some image  $x \in \mathbb{X}$  and attribute-object pair  $(a, o)$ , we get the image and text representations:

$$\mathbf{x}_v = \frac{\text{VLM}_{\mathbf{V}}(x)}{\|\text{VLM}_{\mathbf{V}}(x)\|} \quad \mathbf{t}_{a,o} = \frac{\text{VLM}_{\mathbf{T}}(\psi^{\text{CSP}}(t_{a,o}))}{\|\text{VLM}_{\mathbf{T}}(\psi^{\text{CSP}}(t_{a,o}))\|} \quad (5)$$

Then, we compute the class probability  $p\left(\frac{y=(a,o)}{x;\boldsymbol{\theta}}\right)$  as follows:

$$p\left(\frac{y=(a,o)}{x;\boldsymbol{\theta}}\right) = \frac{\exp(\mathbf{x}_v \cdot \mathbf{t}_{a,o}/\tau)}{\sum_{(\hat{a},\hat{o}) \in \mathbb{Y}_{\text{seen}}} \exp(\mathbf{x}_v \cdot \mathbf{t}_{\hat{a},\hat{o}}/\tau)} \quad (6)$$

Finally, we learn the parameters by minimizing the cross entropy loss on the training dataset:

$$\mathcal{L} = -\frac{1}{|\mathbb{S}|} \sum_{(x,y) \in \mathbb{S}} \log p\left(\frac{y=(a,o)}{x;\boldsymbol{\theta}}\right) + \lambda \|\boldsymbol{\theta}\| \quad (7)$$

where  $\lambda$  is the weight decay.

## 4.2 INFERENCE

During inference, we recompose the fine-tuned attribute and object vocabulary in the prompt. We compose the candidate prompts with the tuned  $\boldsymbol{\theta}$  with the (attribute, object) pairs in the same way during training. In both closed-world and open-world settings, we only replace attribute and objects with the fine-tuned parameters in the prompt. Finally, we calculate the most likely attribute and object pair as follows:

$$\hat{y} = \arg \max_{y \in \mathbb{Y}_{\text{test}}} p\left(\frac{y=(a,o)}{x;\boldsymbol{\theta}}\right) \quad (8)$$

We include the pseudocode for inference in Appendix A.

## 4.3 FEASIBILITY CALIBRATION FOR OPEN-WORLD SETTING

The open-world setting is particularly challenging as the label space contains all possible permutations of attributes and objects in the dataset. For instance, the label space contains feasible compositions such as `young cat` and `eroded cliff` and infeasible compositions such as `eroded cat`. The model must output the correct class while ignoring the infeasible compositions. Existing work shows a significant drop in model performance from the closed-world setting to the open-world setting [27, 28].

Feasibility calibration aims to filter out infeasible compositions that might be present in the open-world setting.

To filter out infeasible compositions, we follow the post-training calibration from [28]. They conjecture that similar objects share similar attributes while dissimilar objects are unlikely to share attributes. For example, `cat` and `dog` can share the attribute `old` but `cat` and `cliff` do not share the attribute `eroded`.

We calculate the feasibility compositions for the composition  $(a, o)$  by computing the relationships between the objects and the attributes. First, we find the similarities between the objects:

$$\rho_o(a, o) = \max_{\hat{o} \in \mathbb{O}_{\text{seen}}} \frac{\phi(o) \cdot \phi(\hat{o})}{\|\phi(o)\| \|\phi(\hat{o})\|} \quad (9)$$

where  $\rho_o(\cdot)$  is the similarity between the object  $o$  with other objects  $\hat{o}$  and  $\phi(\cdot)$  is an embedding function that maps attributes to pretrained embedding. We compute similarities between the attributes in the same way.

Dataset	Composition			Train		Validation			Test		
	A	O	A × O	Y <sub>seen</sub>	X	Y <sub>seen</sub>	Y <sub>unseen</sub>	X	Y <sub>seen</sub>	Y <sub>unseen</sub>	X
MIT-States [15]	115	245	28175	1262	30338	300	300	10420	400	400	12995
UT-Zappos [49]	16	12	192	83	22998	15	15	3214	18	18	2914
C-GQA [27]	413	674	278362	5592	26920	1252	1040	7280	888	923	5098

Table 1: Summary statistics of the datasets used in our experiments.

Next, we combine the two similarities with a pooling function. In our case, we use mean pooling  $\mu$ :

$$\rho(a, o) = \mu(\rho_o(a, o), \rho_a(a, o)) \quad (10)$$

where  $\rho(a, o)$  is the feasibility score for the composition  $(a, o)$ . Finally, we filter out infeasible compositions by considering compositions above a threshold  $T$  calibrated on the validation set to get our final prediction:

$$\hat{y} = \arg \max_{y \in \mathbb{Y}_{test}, \rho(a, o) > T} p\left(\frac{y = (a, o)}{x; \theta}\right) \quad (11)$$

## 5 EXPERIMENTAL EVALUATION

In this section, we describe our experiments with  $\mathbb{CSP}$ . We include the datasets, evaluation metrics, training details, and baselines for our experiments. Then, we compare  $\mathbb{CSP}$  to state-of-the-art methods in the closed-world and open-world settings of compositional zero-shot learning. We include qualitative results for zero-shot image-text retrieval to compare  $\mathbb{CSP}$  and CLIP. Finally, we demonstrate that  $\mathbb{CSP}$  can generalize beyond these benchmarks to two modified settings: attribute-object composition and inference with unseen attributes. The code for our experiments has been released.<sup>1</sup>

### 5.1 DATASET

We experiment with three real-world attribute-object composition datasets: MIT-states [15], UT-Zappos [49], and C-GQA [31]. Table 1 summarizes the statistics of the datasets. MIT-states contains images of naturally occurring objects where each object is described by an adjective. UT-Zappos contains images of shoes paired with fine-grained states. For this dataset, we use the split suggested by Purushwalkam et. al. [35]. C-GQA, a newly introduced dataset derived from the Stanford GQA dataset [13], contains images of objects paired with states.

### 5.2 BENCHMARK EVALUATION

We follow the closed-world and the open-world evaluation protocols to test our models. In the closed-world setting, the seen and the unseen splits are defined in the dataset. In the open-world setting, the model predicts all possible attribute-object permutations present in the dataset.

Following prior work [27], we report the performance in the generalized zero-shot learning for both the closed-world and the open-world settings. In generalized zero-shot learning, we evaluate both the seen and unseen classes in test set. Several works [4, 39] have noted that zero-shot models are biased towards the seen classes even in the presence of unseen classes. To account for the bias, we add a scalar bias to the unseen classes. Next, we vary the bias from  $-\infty$  to  $+\infty$  such we get a curve indicating the seen accuracy on the x-axis and unseen accuracy on the y-axis. Following prior work, we report the area under the curve (AUC) and select the operating point with the best harmonic mean (H) between the seen and unseen accuracy. We also report the best seen accuracy (S) when bias is  $-\infty$  and the best unseen accuracy (U) when the bias is  $+\infty$ .

### 5.3 TRAINING DETAILS

We implement  $\mathbb{CSP}$  with a pretrained CLIP model in PyTorch [33]. CLIP is a large-scale vision-language model used for zero-shot inference. We use the CLIP model ViT-L/14 which is the largest

<sup>1</sup><https://github.com/BatsResearch/csp>

Method	MIT-States				UT-Zappos				CGQA			
	S	U	H	AUC	S	U	H	AUC	S	U	H	AUC
AoP[32]	14.3	17.4	9.9	1.6	59.8	54.2	40.8	25.9	17.0	5.6	5.9	0.7
LE+ [30]	15.0	20.1	10.7	2.0	53.0	61.9	41.0	25.7	18.1	5.6	6.1	0.8
TMN[35]	20.2	20.1	13.0	2.9	58.7	60.0	45.0	29.3	23.1	6.5	7.5	1.1
SymNet[25]	24.2	25.2	16.1	3.0	49.8	57.4	40.4	23.4	26.8	10.3	11.0	2.1
CompCos [27]	25.3	24.6	16.4	4.5	59.8	62.5	43.1	28.1	28.1	11.2	12.4	2.6
ProtoProp [39]	-	-	-	-	62.1	65.5	50.2	<b>34.7</b>	-	-	-	-
CGE [31]	32.8	28.0	21.4	6.5	<b>64.5</b>	<b>71.5</b>	<b>60.5</b>	33.5	<b>33.5</b>	15.5	16.0	4.2
Co-CGE [28]	31.1	5.8	6.4	1.1	62.0	44.3	40.3	23.1	32.1	2.0	3.4	0.5
CLIP [38]	30.2	46.0	26.1	11.0	15.8	49.1	15.6	5.0	7.5	25.0	8.6	1.4
COOP [50]	36.7	49.2	31.6	15.1	62.9	62.3	45.5	31.3	20.9	25.9	17.1	4.4
<b>CS<math>\mathbb{P}</math></b>	<b>46.6</b>	<b>49.9</b>	<b>36.3</b>	<b>19.4</b>	64.2	66.2	46.6	33.0	28.8	<b>26.8</b>	<b>20.5</b>	<b>6.2</b>

Table 2: Closed world results on MIT-States, UT-Zappos, and C-GQA. For COOP and CS $\mathbb{P}$ , we report the average performance of the models on 5 random seeds. See Appendix C for full results with standard errors. The results for AoP, LE+, TMN, SymNet, CompCos, CGE, and Co-CGE are obtained from [28] and ProtoProp from [39].

available model in our experiments.<sup>2</sup> Nonetheless, our method is agnostic to the choice of CLIP architecture. The pretrained CLIP ViT-L/14 model has a vision transformer (ViT) as the image encoder and a transformer as the text encoder. The input the text encoder is a prompt of the form `a photo of [attribute] [object]`. CLIP tokenizes attributes and objects like Nubuck, Faux Fur, etc. as multiple tokens. We simply average the token representations of multiple subwords to get a single representation for the attribute or object.

We train CS $\mathbb{P}$  by minimizing the cross entropy loss with the Adam optimizer over the seen split in the dataset for 20 epochs. We use a single NVIDIA RTX 3090 GPU and a single NVIDIA V100 card depending on their availability to train all our models. For each dataset, we choose the best hyperparameters based on the performance on the validation split. For more details, refer to the Appendix B.

During inference, we recombine the fine-tuned attribute and object vocabulary for represent the unseen classes. Following prior work, we compute feasibility calibration using GloVe embeddings [34] and filter out the infeasible attribute-object compositions based on the performance on the validation split. We report our results on the best performing epoch across all seeds on the validation data.

#### 5.4 BASELINES

We compare CS $\mathbb{P}$  to the existing compositional zero-shot learning methods [25, 28, 27, 32, 31, 35, 39] and CLIP-based methods [38, 50]. We consider the following compositional zero-shot learning methods: AoP [32], LE+ [30], TMN [35], SymNet [25], CompCos [27], CGE [31], and Co-CGE [28]. We also consider ProtoProp [39] in the closed-world setting. Apart from these task-specific methods, we compare CS $\mathbb{P}$  to pretrained CLIP [38] and COOP [50]. COOP is a soft-prompting method that learns the context with limited labeled examples in a few-shot setting. We adapt COOP for compositional zero-shot learning task and learn the context as vocabulary. For fair comparison, we use the same prompt format as CS $\mathbb{P}$  for both CLIP and COOP experiments. Finally, in the open-world setting, we apply feasibility calibration to CLIP and COOP predictions to filter out infeasible compositions.

#### 5.5 BENCHMARK RESULTS

Our results in Table 2 show that CS $\mathbb{P}$  sets the new state-of-the-art on MIT-States and C-GQA datasets. CS $\mathbb{P}$  significantly improves over CLIP on all datasets in the closed-world setting. We outperform CLIP in the best harmonic mean by 10.2 points on MIT-states, 31.0 point on UT-Zappos, and 11.9 points on C-GQA. Additionally, we show that CS $\mathbb{P}$  beats COOP, a soft-prompting method,

<sup>2</sup><https://github.com/openai/CLIP/blob/main/model-card.md>

Method	MIT-States				UT-Zappos				CGQA			
	S	U	H	AUC	S	U	H	AUC	S	U	H	AUC
AoP[32]	16.6	5.7	4.7	0.7	50.9	34.2	29.4	13.7	-	-	-	-
LE+ [30]	14.2	2.5	2.7	0.3	60.4	36.5	30.5	16.3	19.2	0.7	1.0	0.08
TMN[35]	12.6	0.9	1.2	0.1	55.9	18.1	21.7	8.4	-	-	-	-
SymNet[25]	21.4	7.0	5.8	0.8	53.3	44.6	34.5	18.5	26.7	2.2	3.3	0.43
CompCos [27]	21.4	7.0	5.8	0.8	53.3	44.6	34.5	18.5	26.7	2.2	3.3	0.43
CGE [31]	32.4	5.1	6.0	1.0	61.7	<b>47.7</b>	39.0	23.1	<b>32.7</b>	1.8	2.9	0.47
Co-CGE <sup>CW</sup> [28]	31.1	5.8	6.4	1.1	62.0	44.3	40.3	23.1	32.1	2.0	3.4	0.53
Co-CGE <sup>open</sup> [28]	30.3	11.2	10.7	2.3	61.2	45.8	<b>40.8</b>	<b>23.3</b>	32.1	3.0	4.8	0.78
CLIP [38]	30.1	14.3	12.8	3.0	15.7	20.6	11.2	2.2	7.5	4.6	4.0	0.27
COOP [50]	36.8	<b>16.5</b>	16.1	4.7	61.8	39.3	35.6	19.5	20.9	4.5	5.7	0.73
<b>CS<sub>P</sub></b>	<b>46.3</b>	15.7	<b>17.4</b>	<b>5.7</b>	<b>64.1</b>	44.1	38.9	22.7	28.7	<b>5.2</b>	<b>6.9</b>	<b>1.20</b>

Table 3: Open world results on MIT-States, UT-Zappos, and C-GQA. For COOP and CS<sub>P</sub>, we report the average performance of the models on 5 random seeds. See Appendix C for full results with standard errors. The results for AoP, LE+, TMN, SymNet, CompCos, CGE, and Co-CGE are obtained from [28].

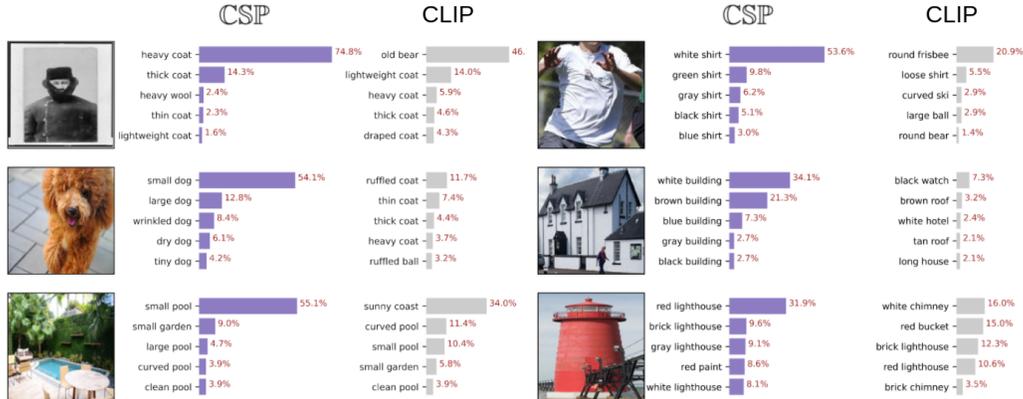


Figure 4: Qualitative comparison for image to text retrieval between CS<sub>P</sub> and CLIP on CGQA. Selected samples with concepts correctly identified and top-5 retrieval results by CS<sub>P</sub> are shown.

in the best harmonic mean by 4.7 points on MIT-States, 1.1 points on UT-Zappos, and 3.4 points on C-GQA. Finally, CS<sub>P</sub> improves over existing compositional zero-shot learning method by 14.9 points on MIT-states and 4.5 points on C-GQA on the harmonic mean metric.

Table 3 shows that CS<sub>P</sub> reports the new state-of-the-art on MIT-States and C-GQA datasets in the open-world setting. Our results show that CS<sub>P</sub> improves over CLIP by 4.6 points on MIT-States, 27.7 points on UT-Zappos, and 2.9 points on C-GQA in the harmonic mean metric. We outperform COOP on MIT-States by 1.3 points, UT-Zappos by 3.3 points, and C-GQA by 1.2 points on the best harmonic mean metric. We improve over the existing compositional zero-shot learning methods on MIT-States by 2.8 points and C-GQA by 2.5 points on the harmonic mean metric. Finally, improving the feasibility calibration could further reduce the gap in performance between closed-world setting and the open-world setting. We also conduct experiments with different backbones and the trend is consistent. We report the results in Appendix D.

We qualitatively evaluate the effectiveness of CS<sub>P</sub> in compositional zero-shot image to text retrieval tasks. Selected samples from CGQA in Figure 4 show that fine-tuning the vocabulary enables CS<sub>P</sub> to better identify composed concepts compared to CLIP. We include more qualitative comparisons in Appendix E.

### Generalization with Unseen Attributes

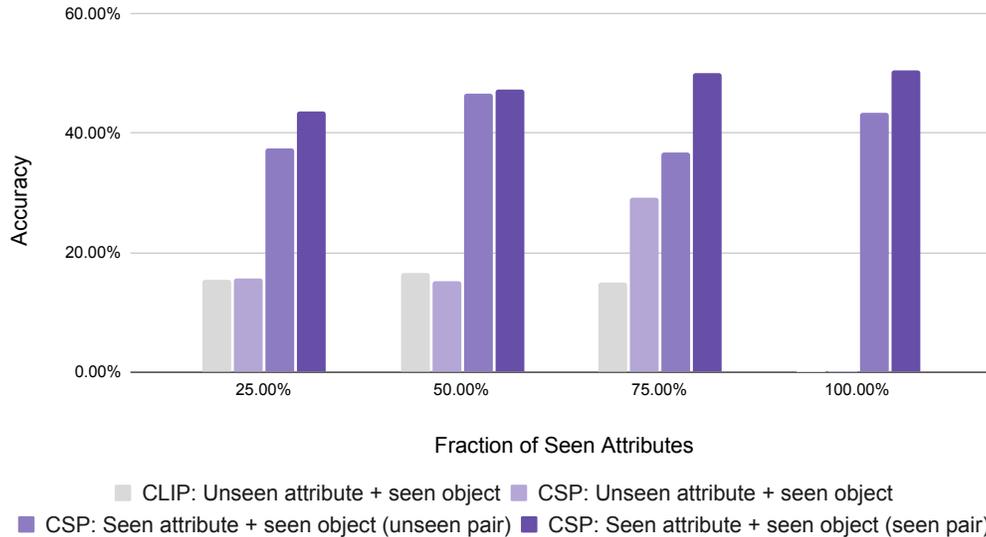


Figure 5: Results of CSP and CLIP with different fractions of pretrained and fine-tuned vocabulary. In each fraction, we report the average performance of CLIP and CSP on 5 random attribute splits.

### 5.6 GENERALIZATION TO HIGHER-ORDER COMPOSITIONS

To test the additional flexibility afforded by VLMs, we test if training CSP with attribute-object compositions can generalize to higher-order compositions such as attribute-attribute-object compositions. We annotate a novel challenge dataset: AAO-MIT-States, a subset derived from the MIT-States dataset. In this dataset, we annotate the images in the test split of the MIT-States dataset with an additional attribute, to get an attribute-attribute-object pair as the class label. More details on the annotation are included in Appendix F.

We compare CLIP and CSP to classify images with attribute-attribute-object classes and report the accuracy. Since these class compositions are not present during training, we treat them as unseen classes and calculate the unseen accuracy. We use the same best performing models for CSP from MIT-States and run inference on the challenge dataset.

Model	Accuracy
CLIP	62.7
CSP	<b>72.7 ± 0.5</b>

Table 4 shows that CSP improves over CLIP by an average 10 percentage points on unseen accuracy and generalizes to attribute-attribute-object compositions without any modifications or training. The results demonstrate that CSP improves the compositionality of CLIP’s vocabulary, even in ways that were not explicitly supervised.

Table 4: Unseen accuracy on 5 random seeds with standard error.

### 5.7 GENERALIZATION TO MIXED PRETRAINED AND FINE-TUNED VOCABULARY

The further test the additional flexibility afforded by VLMs, we also evaluate CSP on compositional zero-shot learning with a mixture of pretrained and fine-tuned vocabulary. This evaluation stems from the practical need to combine new unseen attributes with fine-tuned vocabulary. Evaluating in this setting will allow us to assess whether the benefits of CSP extend to classes including vocabulary not seen during fine-tuning. This setup goes beyond the above benchmarks, which include unseen combinations of attributes and objects, but all attributes and objects are seen during training. Now, we include completely unseen attributes.

We apply CSP on UT-Zappos with different fractions of attributes as seen attributes. We randomly select 25%, 50%, 75%, and 100% of the attributes and all the objects from the training set. Then,

---

we remove from the seen classes the attribute-object pairs that include an unseen attribute. Finally, we train the on the remaining seen attribute-object pairs with five random seed values.

For each split of the seen and unseen attributes, we evaluate CSP by dividing the classes into three buckets: (1) unseen attribute + seen object pairs, (2) seen (i.e., fine-tuned) attribute + seen object pairs in unseen combinations, and (3) seen attribute + seen object pairs in seen combinations. In this evaluation, we refer to the classes in the first and second buckets as the unseen classes and those in the third bucket as the seen classes. This evaluation is more general than typical CZSL settings, which only evaluate on classes in the second and third buckets. Similar to our evaluation in Section 5.2, we add a scalar bias to the unseen classes and vary the bias from  $-\infty$  to  $+\infty$ . As in other CZSL evaluations, we select the bias that maximizes the harmonic mean between accuracy on the unseen and seen classes. We then report accuracy on classes in each of the three buckets. To contextualize the performance of CSP, we report the accuracy of CLIP on the unseen attribute + seen object pairs.

Our results in Figure 5 show that the performance on unseen attribute + seen object pairs improves with CSP and sufficient training pairs. Initially, the performance of CLIP and CSP are comparable but by providing more combinations of supervision for the objects CSP significantly outperforms CLIP on the unseen attribute + seen object evaluation bucket. These results demonstrate that the fine-tuned vocabulary of CSP can improve the compositional zero-shot performance of pretrained vocabulary.

## 6 CONCLUSION

We present a new style of soft-prompting, CSP, for compositional zero-shot learning. We show that learning composable components of classes via soft prompting can improve downstream compositional zero-shot performance with a small number of parameters. We also demonstrate the importance of a rich and flexible textual encoder in generalizing to higher-order compositions and mixtures of seen and unseen vocabulary.

**Authors’ Note.** The first two authors contributed equally. Co-first authors can prioritize their names when adding this paper’s reference to their resumes.

## ACKNOWLEDGMENTS

We thank Andrew Delworth and Elise Carman for helping us annotate the AAO-MIT-States dataset. We appreciate the comments and advice from Cristina Menghini, Wasu Piriyaikulij and Zheng-Xin Yong on our drafts. This material is based on research sponsored by Defense Advanced Research Projects Agency (DARPA) and Air Force Research Laboratory (AFRL) under agreement number FA8750-19-2-1006. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of Defense Advanced Research Projects Agency (DARPA) and Air Force Research Laboratory (AFRL) or the U.S. Government. We gratefully acknowledge support from Google and Cisco. Disclosure: Stephen Bach is an advisor to Snorkel AI, a company that provides software and services for weakly supervised machine learning.

## REFERENCES

- [1] S. H. Bach, V. Sanh, Z.-X. Yong, A. Webson, C. Raffel, N. V. Nayak, A. Sharma, T. Kim, M. S. Bari, T. Fevry, Z. Alyafeai, M. Dey, A. Santilli, Z. Sun, S. Ben-David, C. Xu, G. Chhablani, H. Wang, J. A. Fries, M. S. Al-shaibani, S. Sharma, U. Thakker, K. Almubarak, X. Tang, D. Radev, M. T.-J. Jiang, and A. M. Rush. PromptSource: An integrated development environment and repository for natural language prompts. In *Meeting of the Association for Computational Linguistics (ACL) Demonstration, 2022*.
- [2] R. Bommasani, D. A. Hudson, E. Adeli, R. Altman, S. Arora, S. von Arx, M. S. Bernstein, J. Bohg, A. Bosselut, E. Brunskill, E. Brynjolfsson, S. Buch, D. Card, R. Castellon, N. S. Chatterji, A. S. Chen, K. Creel, J. Davis, D. Demszky, C. Donahue, M. Doumbouya, E. Durmus, S. Ermon, J. Etchemendy, K. Ethayarajh, L. Fei-Fei, C. Finn, T. Gale, L. E. Gillespie,

- 
- K. Goel, N. D. Goodman, S. Grossman, N. Guha, T. Hashimoto, P. Henderson, J. Hewitt, D. E. Ho, J. Hong, K. Hsu, J. Huang, T. F. Icard, S. Jain, D. Jurafsky, P. Kalluri, S. Karamcheti, G. Keeling, F. Khani, O. Khattab, P. W. Koh, M. S. Krass, R. Krishna, R. Kuditipudi, A. Kumar, F. Ladhak, M. Lee, T. Lee, J. Leskovec, I. Levent, X. L. Li, X. Li, T. Ma, A. Malik, C. D. Manning, S. P. Mirchandani, E. Mitchell, Z. Munyikwa, S. Nair, A. Narayan, D. Narayanan, B. Newman, A. Nie, J. C. Niebles, H. Nilforoshan, J. F. Nyarko, G. Ogut, L. Orr, I. Papadimitriou, J. S. Park, C. Piech, E. Portelance, C. Potts, A. Raghunathan, R. Reich, H. Ren, F. Rong, Y. H. Roohani, C. Ruiz, J. Ryan, C. R'e, D. Sadigh, S. Sagawa, K. Santhanam, A. Shih, K. P. Srinivasan, A. Tamkin, R. Taori, A. W. Thomas, F. Tramèr, R. E. Wang, W. Wang, B. Wu, J. Wu, Y. Wu, S. M. Xie, M. Yasunaga, J. You, M. A. Zaharia, M. Zhang, T. Zhang, X. Zhang, Y. Zhang, L. Zheng, K. Zhou, and P. Liang. On the opportunities and risks of foundation models. *ArXiv*, abs/2108.07258, 2021.
- [3] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. J. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners. *ArXiv*, 2020.
- [4] W.-L. Chao, S. Changpinyo, B. Gong, and F. Sha. An empirical study and analysis of generalized zero-shot learning for object recognition in the wild. In *European conference on computer vision (ECCV)*, 2016.
- [5] N. Chomsky. Three models for the description of language. *IRE Transactions on information theory*, 2(3):113–124, 1956.
- [6] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *ArXiv*, abs/2010.11929, 2021.
- [7] A. Farhadi, I. Endres, D. Hoiem, and D. A. Forsyth. Describing objects by their attributes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.
- [8] J. A. Fodor and Z. W. Pylyshyn. Connectionism and cognitive architecture: A critical analysis. *Cognition*, 28:3–71, 1988.
- [9] C. Gan, M. Lin, Y. Yang, Y. Zhuang, and A. G. Hauptmann. Exploring semantic inter-class relationships (sir) for zero-shot action recognition. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2015.
- [10] W. Hao, C. Li, X. Li, L. Carin, and J. Gao. Towards learning a generic agent for vision-and-language navigation via pre-training. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13134–13143, 2020.
- [11] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [12] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. de Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly. Parameter-efficient transfer learning for nlp. In *ICML*, 2019.
- [13] D. A. Hudson and C. D. Manning. Gqa: A new dataset for real-world visual reasoning and compositional question answering. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.
- [14] D. Hupkes, V. Dankers, M. Mul, and E. Bruni. Compositionality decomposed: How do neural networks generalise? *J. Artif. Intell. Res.*, 67:757–795, 2020.
- [15] P. Isola, J. J. Lim, and E. H. Adelson. Discovering states and transformations in image collections. In *CVPR*, 2015.
- [16] A. Jain, M. Guo, K. Srinivasan, T. Chen, S. Kudugunta, C. Jia, Y. Yang, and J. Baldridge. Mural: multimodal, multitask retrieval across languages. *arXiv preprint arXiv:2109.05125*, 2021.

- 
- [17] C. Jia, Y. Yang, Y. Xia, Y.-T. Chen, Z. Parekh, H. Pham, Q. V. Le, Y.-H. Sung, Z. Li, and T. Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *ICML*, 2021.
- [18] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2017.
- [19] B. M. Lake and M. Baroni. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *ICML*, 2018.
- [20] C. H. Lampert, H. Nickisch, and S. Harmeling. Attribute-based classification for zero-shot visual object categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 36(3):453–465, 2013.
- [21] B. Lester, R. Al-Rfou, and N. Constant. The power of scale for parameter-efficient prompt tuning. In *EMNLP*, 2021.
- [22] B. Li, K. Q. Weinberger, S. Belongie, V. Koltun, and R. Ranftl. Language-driven semantic segmentation. In *International Conference on Learning Representations*, 2022.
- [23] X. L. Li and P. Liang. Prefix-tuning: Optimizing continuous prompts for generation, 2021.
- [24] Y. Li, F. Liang, L. Zhao, Y. Cui, W. Ouyang, J. Shao, F. Yu, and J. Yan. Supervision exists everywhere: A data efficient contrastive language-image pre-training paradigm. *ArXiv*, abs/2110.05208, 2021.
- [25] Y.-L. Li, Y. Xu, X. Mao, and C. Lu. Symmetry and group in attribute-object compositions. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11313–11322, 2020.
- [26] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, and G. Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ArXiv*, abs/2107.13586, 2021.
- [27] M. Mancini, M. Naeem, Y. Xian, and Z. Akata. Open world compositional zero-shot learning. In *34th IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2021.
- [28] M. Mancini, M. F. Naeem, Y. Xian, and Z. Akata. Learning graph embeddings for open world compositional zero-shot learning. *ArXiv*, abs/2105.01017, 2021.
- [29] G. F. Marcus. *The algebraic mind: Integrating connectionism and cognitive science*. 2001.
- [30] I. Misra, A. K. Gupta, and M. Hebert. From red wine to red tomato: Composition with context. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1160–1169, 2017.
- [31] M. F. Naeem, Y. Xian, F. Tombari, and Z. Akata. Learning graph embeddings for compositional zero-shot learning. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 953–962, 2021.
- [32] T. Nagarajan and K. Grauman. Attributes as operators. *ECCV*, 2018.
- [33] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems* 32, 2019.
- [34] J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014.
- [35] S. Purushwalkam, M. Nickel, A. Gupta, and M. Ranzato. Task-driven modular networks for zero-shot compositional learning. *ICCV*, 2019.

- 
- [36] G. Qin and J. Eisner. Learning how to ask: Querying lms with mixtures of soft prompts. In *EMNLP*, 2021.
- [37] F. Radenović, A. Sinha, A. Gordo, T. L. Berg, and D. K. Mahajan. Large-scale attribute-object compositions. *ArXiv*, 2021.
- [38] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever. Learning transferable visual models from natural language supervision. In *ICML*, 2021.
- [39] F. Ruis, G. J. Burghouts, and D. Bucur. Independent prototype propagation for zero-shot compositionality. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, 2021.
- [40] V. Sanh, A. Webson, C. Raffel, S. Bach, L. Sutawika, Z. Alyafeai, A. Chaffin, A. Stiegler, A. Raja, M. Dey, M. S. Bari, C. Xu, U. Thakker, S. S. Sharma, E. Szczechla, T. Kim, G. Chhablani, N. Nayak, D. Datta, J. Chang, M. T.-J. Jiang, H. Wang, M. Manica, S. Shen, Z. X. Yong, H. Pandey, R. Bawden, T. Wang, T. Neeraj, J. Rozen, A. Sharma, A. Santilli, T. Fevry, J. A. Fries, R. Teehan, T. L. Scao, S. Biderman, L. Gao, T. Wolf, and A. M. Rush. Multi-task prompted training enables zero-shot task generalization. In *International Conference on Learning Representations*, 2022.
- [41] T. Shin, Y. Razeghi, R. L. Logan IV, E. Wallace, and S. Singh. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020.
- [42] W. Su, X. Zhu, Y. Cao, B. Li, L. Lu, F. Wei, and J. Dai. V1-bert: Pre-training of generic visual-linguistic representations. In *International Conference on Learning Representations*, 2020.
- [43] C. Sun, A. Myers, C. Vondrick, K. P. Murphy, and C. Schmid. Videobert: A joint model for video and language representation learning. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7463–7472, 2019.
- [44] H. Tan and M. Bansal. Lxmert: Learning cross-modality encoder representations from transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, 2019.
- [45] T. Vu, B. Lester, N. Constant, R. Al-Rfou, and D. M. Cer. Spot: Better frozen model adaptation through soft prompt transfer. *ArXiv*, 2021.
- [46] W. Wang, V. W. Zheng, H. Yu, and C. Miao. A survey of zero-shot learning: Settings, methods, and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2):1–37, 2019.
- [47] F. Wu, T. Zhang, A. H. d. Souza Jr, C. Fifty, T. Yu, and K. Q. Weinberger. Simplifying graph convolutional networks. In *International Conference on Machine Learning (ICML)*, 2019.
- [48] Y. Xian, C. H. Lampert, B. Schiele, and Z. Akata. Zero-shot learning—a comprehensive evaluation of the good, the bad and the ugly. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 41(9):2251–2265, 2018.
- [49] A. Yu and K. Grauman. Fine-grained visual comparisons with local learning. *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 192–199, 2014.
- [50] K. Zhou, J. Yang, C. C. Loy, and Z. Liu. Learning to prompt for vision-language models. *arXiv preprint arXiv:2109.01134*, 2021.

## A PSEUDOCODE

```

def inference(batch_images: nn.Tensor,
              test_pairs: List[List, List],
              model: nn.Module):
    """
    Function to run inference with the fine-tuned embeddings.
    Args:
        batch_images (torch.Tensor): minibatch of images [n, h, w, c]
        test_pairs (tuple): attribute-object pairs in the test
            split [m, 2]
        model (nn.Module): model with the fine-tuned embeddings
    Returns:
        torch.Tensor: cosine similarities of the minibatch images
            and attribute-object pairs [n, m]
    """
    prompt_template = "a photo of x x"
    tokenized_prompt = tokenize(prompt_template)
    tokenized_prompt = tokenized_prompt.repeat(len(test_pairs))
    token_tensor = model.token_embedding(tokenized_prompt)

    # fine-tuned embeddings
    attr, obj = zip(*test_pairs)
    attr_emb = model.soft_embedding(attr)
    obj_emb = model.soft_embedding(obj)

    # replace the "x x" in prompt template with fine-tuned embeddings
    token_tensor = replace_emb(token_tensor, attr_emb, obj_emb)

    # l2-normalized
    text_rep = model.text_encoder(token_tensor)
    image_rep = model.image_encoder(batch_images)

    logits = (image_rep @ text_rep) * model.logit_scale.exp()

    return logits

```

Figure 6: Torch-like pseudocode for inference with CSP.

Figure 6 shows the Torch-like pseudocode for inference with CSP. The function accepts the mini-batch of images, test pairs, and the clip model with the fine-tuned embeddings and returns the cosine similarities between the image representation and the text representation scaled by a constant scalar.

## B HYPERPARAMETERS

Hyperparameter	MIT-States	UT-Zappos	C-GQA
Learning rate	$5e - 05$	$5e - 04$	$5e - 05$
Batch size	128	128	128
Attribute dropout	0.3	0.2	0.3
Weight decay	$1e - 05$	$1e - 05$	$5e - 05$

Table 5: Hyperparameters for MIT-States, UT-Zappos, and C-GQA.

In our work, we find the best hyperparameters for training CSP via a grid search. We train the ViT-B/32 model for 50 epochs and use the same best performing hyperparameters to train all our models including ViT L/14. We run a grid search with the following hyperparameters: (1) learning rate:  $\{5e - 03, 5e - 04, 5e - 05\}$ , (2) batch size:  $\{128, 256\}$ , (3) attribute dropout:  $\{0.0, 0.1, 0.2, 0.3\}$ , and (4) weight decay:  $\{1e - 05, 5e - 05\}$ . We choose the hyperparameters for a dataset based best unseen accuracy on the validation split. We reduce the number of epochs to 20 with ViT L/14 as we found our models tend to converge earlier. Table 5 shows the hyperparameters used to train our models on all the datasets.

## C RESULTS WITH STANDARD ERROR

We include the extended results with standard errors for the closed-world and the open-world settings. Table 6 shows the results for the closed-world setting with the standard error. Table 7 shows the results for the open-world setting with the standard error.

Method	MIT-States				UT-Zappos				CGQA			
	S	U	H	AUC	S	U	H	AUC	S	U	H	AUC
AoP[32]	14.3	17.4	9.9	1.6	59.8	54.2	40.8	25.9	17.0	5.6	5.9	0.7
LE+ [30]	15.0	20.1	10.7	2.0	53.0	61.9	41.0	25.7	18.1	5.6	6.1	0.8
TMN[35]	20.2	20.1	13.0	2.9	58.7	60.0	45.0	29.3	23.1	6.5	7.5	1.1
SymNet[25]	24.2	25.2	16.1	3.0	49.8	57.4	40.4	23.4	26.8	10.3	11.0	2.1
CompCos [27]	25.3	24.6	16.4	4.5	59.8	62.5	43.1	28.1	28.1	11.2	12.4	2.6
ProtoProp [39]	-	-	-	-	62.1	65.5	50.2	<b>34.7</b>	-	-	-	-
CGE [31]	32.8	28.0	21.4	6.5	<b>64.5</b>	<b>71.5</b>	<b>60.5</b>	<b>34.7</b>	<b>33.5</b>	15.5	16.0	4.2
Co-CGE [28]	31.1	5.8	6.4	1.1	62.0	44.3	40.3	23.1	32.1	2.0	3.4	0.5
CLIP [38]	30.2	46.0	26.1	11.0	15.8	49.1	15.6	5.0	7.5	25.0	8.6	1.4
COOP [50]	36.7 ± 0.1	49.2 ± 0.1	31.6 ± 0.1	15.1 ± 0.1	62.9 ± 0.5	62.3 ± 0.6	45.5 ± 1.3	31.3 ± 1.1	20.9 ± 0.2	25.9 ± 0.2	17.1 ± 0.1	4.4 ± 0.0
CSP	<b>46.6 ± 0.1</b>	<b>49.9 ± 0.1</b>	<b>36.3 ± 0.1</b>	<b>19.4 ± 0.1</b>	64.2 ± 0.7	66.2 ± 1.2	46.6 ± 1.2	33.0 ± 1.3	28.8 ± 0.1	<b>26.8 ± 0.1</b>	<b>20.5 ± 0.1</b>	<b>6.2 ± 0.0</b>

Table 6: Closed world results on MIT-States, UT-Zappos, and C-GQA. For COOP and CSP, we report the average performance of the models on 5 random seeds with standard error. The results for AoP, LE+, TMN, SymNet, CompCos, CGE, and Co-CGE are obtained from [28] and ProtoProp from [39].

Method	MIT-States				UT-Zappos				CGQA			
	S	U	H	AUC	S	U	H	AUC	S	U	H	AUC
AoP[32]	16.6	5.7	4.7	0.7	50.9	34.2	29.4	13.7	-	-	-	-
LE+ [30]	14.2	2.5	2.7	0.3	60.4	36.5	30.5	16.3	19.2	0.7	1.0	0.08
TMN[35]	12.6	0.9	1.2	0.1	55.9	18.1	21.7	8.4	-	-	-	-
SymNet[25]	21.4	7.0	5.8	0.8	53.3	44.6	34.5	18.5	26.7	2.2	3.3	0.43
CompCos [27]	21.4	7.0	5.8	0.8	53.3	44.6	34.5	18.5	26.7	2.2	3.3	0.43
CGE [31]	32.4	5.1	6.0	1.0	61.7	<b>47.7</b>	39.0	23.1	<b>32.7</b>	1.8	2.9	0.47
Co-CGE <sup>CW</sup> [28]	31.1	5.8	6.4	1.1	62.0	44.3	40.3	23.1	32.1	2.0	3.4	0.53
Co-CGE <sup>open</sup> [28]	30.3	11.2	10.7	2.3	61.2	45.8	<b>40.8</b>	<b>23.3</b>	32.1	3.0	4.8	0.78
CLIP [38]	30.1	14.3	12.8	3.0	15.7	20.6	11.2	2.2	7.5	4.6	4.0	0.27
COOP [50]	36.8 ± 0.1	<b>16.5 ± 0.1</b>	16.1 ± 0.1	4.7 ± 0.0	61.8 ± 0.5	39.3 ± 1.3	35.6 ± 0.7	19.5 ± 0.6	20.9 ± 0.3	4.5 ± 0.2	5.7 ± 0.2	0.73 ± 0.0
CSP	<b>46.3 ± 0.3</b>	15.7 ± 0.1	<b>17.4 ± 0.1</b>	<b>5.7 ± 0.0</b>	<b>64.1 ± 0.7</b>	44.1 ± 0.3	38.9 ± 0.5	22.72 ± 0.4	28.7 ± 0.2	<b>5.2 ± 0.1</b>	<b>6.9 ± 0.1</b>	<b>1.20 ± 0.0</b>

Table 7: Open world results on MIT-States, UT-Zappos, and C-GQA. For COOP and CSP, we report the average performance of the models on 5 random seeds with standard error. The results for AoP, LE+, TMN, SymNet, CompCos, CGE, and Co-CGE are obtained from [28].

## D MODEL ABLATION

Table 8 shows that CSP with a larger backbone improves the state-of-the-art reported by CSP in Section 5.5. We note that CSP generally improves performance over CLIP. In particular, we see the gains are highest with ViT backbones.

Method	Backbone	MIT-States				UT-Zappos				CGQA			
		S	U	H	AUC	S	U	H	AUC	S	U	H	AUC
CLIP	ResNet-50	21.1	34.4	18.4	5.6	6.4	43.6	6.4	1.4	6.1	17.1	6.1	0.7
CLIP	ResNet-101	25.2	37.4	21.7	7.5	11.2	35.2	11.9	2.8	7.3	19.7	7.6	1.1
CLIP	ViT B/32	25.1	39.1	21.4	7.5	9.6	42.4	10.0	2.4	7.3	22.1	7.4	1.2
CLIP	ViT L/14	30.2	46.0	26.1	11.0	15.8	49.1	15.6	5.0	7.5	25.0	8.6	1.4
CSP	ResNet-50	35.0 ± 0.1	30.3 ± 0.1	23.0 ± 0.1	8.3 ± 0.1	21.8 ± 1.6	11.3 ± 1.7	10.2 ± 1.2	1.8 ± 0.3	17.9 ± 0.3	14.7 ± 0.2	10.2 ± 0.2	1.8 ± 0.0
CSP	ResNet-101	38.9 ± 0.1	32.1 ± 0.2	25.2 ± 0.1	9.9 ± 0.1	42.2 ± 1.2	9.1 ± 1.3	10.0 ± 1.1	2.6 ± 0.5	17.7 ± 0.1	17.0 ± 0.2	12.0 ± 0.1	2.3 ± 0.0
CSP	ViT B/32	36.4 ± 0.4	42.5 ± 0.2	28.6 ± 0.1	12.4 ± 0.1	57.1 ± 0.4	57.3 ± 0.6	39.3 ± 0.6	24.2 ± 0.4	<b>30.1 ± 0.1</b>	33.4 ± 0.2	19.4 ± 0.3	5.7 ± 0.1
CSP	ViT L/14	<b>46.6 ± 0.1</b>	<b>49.9 ± 0.1</b>	<b>36.3 ± 0.1</b>	<b>19.4 ± 0.1</b>	<b>64.2 ± 0.7</b>	<b>66.2 ± 1.2</b>	<b>46.6 ± 1.2</b>	<b>33.0 ± 1.3</b>	28.8 ± 0.1	<b>26.8 ± 0.1</b>	<b>20.5 ± 0.1</b>	<b>6.2 ± 0.0</b>

Table 8: Closed world ablation results with respect to different backbone architectures of CLIP.

## E ADDITIONAL QUALITATIVE EXAMPLES

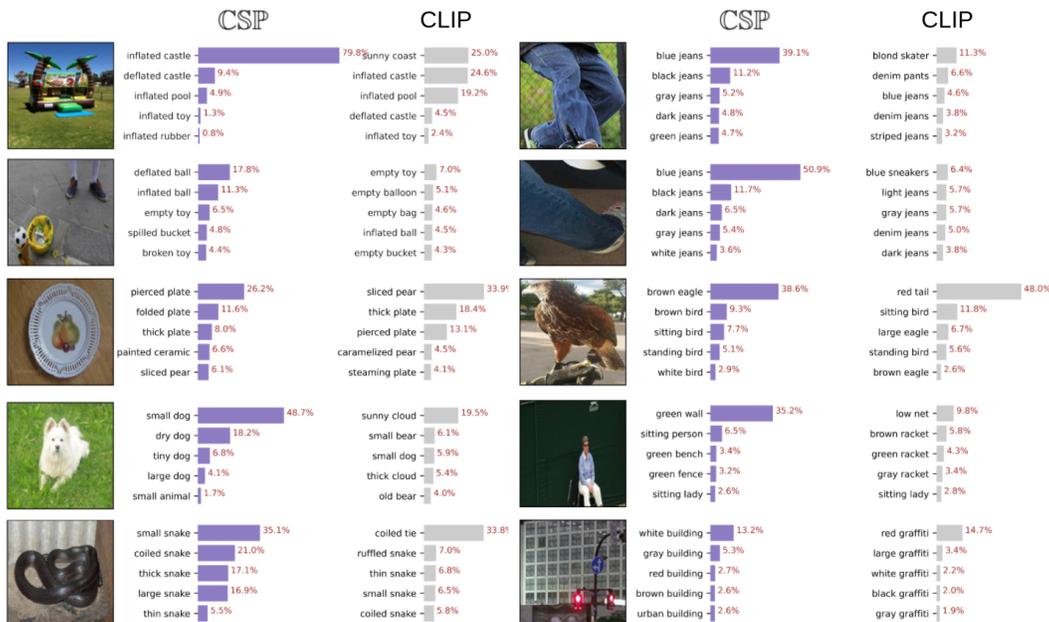


Figure 7: Additional qualitative comparison for image to text retrieval between CSP and CLIP on CGQA. Selected samples with concepts correctly identified and top-5 retrieval results by CSP are shown.

## F DATASET CREATION

We create AAO-MIT-States from the MIT-States dataset [15]. Below we include details on the annotation interface, annotators, and aggregation process for the annotations.

We annotate an additional attribute for the images paired with unseen classes in the test split of MIT-States. The interface has three main components: (1) general instructions, (2) image with caption, and (3) list of populated attributes. The general instructions provide the annotators with a detailed description of the annotation task. To the right of the instructions is a randomly sampled image from the test split of MIT-States. Additionally, we include a caption describing the image. For example, suppose we select an image of a wet cat, we ask the users: “which attribute best describes the cat in the image presented?”. Since we have a large number of attributes in the MIT-States dataset, we need a way to reduce the list of attributes the user observes while annotating a single image. We use CLIP to predict the attributes except for the original attribute in the image and choose the top-5 attributes as annotation candidates. We also include an option to select none of the above.

---

Which attributes best describe the stone in the image presented?



Select an option	
cracked	1
eroded	2
weathered	3
scratched	4
ancient	5
None of the Above	6

Figure 8: Example annotation interface.

The dataset was annotated by two of the authors and two undergraduate research assistants. We randomly sampled a total of 1200 images from test-split and asked the annotators to annotate from the interface. We received annotations for 1089 instances where each image received exactly three annotations. We aggregate examples for our dataset where all the three annotators agreed on the same attribute other than “None of the above”. The total number of examples in the final annotated dataset is 193. We have open-sourced the dataset in our code.