Disentangling Visual Embeddings for Attributes and Objects

Nirat Saini           Khoi Pham           Abhinav Shrivastava
University of Maryland, College Park

Abstract

We study the problem of compositional zero-shot learning for object-attribute recognition. Prior works use visual features extracted with a backbone network, pre-trained for object classification and thus do not capture the subtly distinct features associated with attributes. To overcome this challenge, these studies employ supervision from the linguistic space, and use pre-trained word embeddings to better separate and compose attribute-object pairs for recognition. Analogous to linguistic embedding space, which already has unique and agnostic embeddings for object and attribute, we shift the focus back to the visual space and propose a novel architecture that can disentangle attribute and object features in the visual space. We use visual decomposed features to hallucinate embeddings that are representative for the seen and novel compositions to better regularize the learning of our model. Extensive experiments show that our method outperforms existing work with significant margin on three datasets: MIT-States, UT-Zappos, and a new benchmark created based on VAW. The code, models, and dataset splits are publicly available at https://github.com/nirat1606/OADis.

1. Introduction

Objects in the real world can appear with different properties, i.e., different color, shape, material, etc. For instance, an apple can be red or green, cut or peeled, raw or ripe, and even dirty or clean. Understanding object properties can greatly benefit various applications, e.g., robust object detection [5, 14, 15, 26], human object interaction [7, 49, 51], and activity recognition [1, 3, 4, 16, 18, 34]. Since the total number of possible attribute-object pairs in the real world is prohibitively large, it is impractical to collect image examples and train multiple classifiers. Prior works proposed compositional learning, i.e., learning to compose knowledge of known attributes and object concepts to recognize a new attribute-object composition. Datasets such as MIT-States [24] and UT-Zappos [56] are commonly used to study this task, with joint attribute-object recognition for a diverse, yet limited set of objects and attributes.

Compositional learning refers to combining simple primitive concepts to understand a complex concept. This idea dates back to Recognition and Composition theory by Biederman [6], and early work in the visual domain by Hoffman [22], which proposed recognition by parts for pose estimation. Prior works explore compositionality to a certain degree, e.g., via feature sharing and shared embeddings space. Among them, most works use linguistically inspired losses to separate attributes and objects in the shared embedding space, then use that primitive knowledge to compose new complex pairs. Using linguistic embeddings is helpful since: (1) there is a clear distinction between attribute and object in the embedding space, and (2) these embeddings already contain semantic knowledge of similar objects and attributes, which is helpful for composition. However, unlike word embedding, it is difficult to discrimi-
erate the object and attribute in the visual embedding space. This is due to the fact that image feature extractor is usually pre-trained for object classification, often along with image augmentation (e.g., color jitter) that tends to produce attribute-invariant image representation, thus does not learn objects and attributes separately. In this paper, we propose a new direction that focuses on visual cues, instead of using linguistic cues explicitly for novel compositions.

Analogous to linguistic embedding, our work focuses on disentangling attribute and object in the visual space. Our method, Object Attribute Disentanglement (OADis), learns distinct and independent visual embeddings for peeled apple and sliced apple from the visual feature of peeled apple. As shown in Figure 1, for image I of peeled apple, we use two other images: one with same object and different attribute I_{db} (e.g., sliced apple), and one with same attribute and different object I_{at} (e.g., peeled orange). OADis takes I and I_{db} and learns the similarity (apple) and dissimilarity (sliced) of the second image with respect to the first one. Similarly, using I and I_{at}, the commonality between them (peeled) and the left out dissimilarity (orange) can also be extracted. Further, composition of these extracted visual primitives are used to hallucinate seen and unseen pair, peeled apple and sliced orange respectively.

For compositional learning, it is necessary to decompose first before composing new unseen attribute-object pairs. As humans, we have the ability to imagine an unseen complex concept using previous knowledge of its primitive concepts. For example, if someone has seen a clown and a unicycle, they can imagine clown on a unicycle even if they have never seen this combination in real life [20, 43]. This quality of imagination is the basis of various works such as GANs [12], CLIP [47] and DALL-E [48]. However, these works rely on larger datasets and high computation power for training. We study this idea of imagination for a smaller setup by composing newer complex concepts using disentangled attributes and object visual features. Our work focuses on answering the question, can there be visual embedding of peeled apple, disentangled separately from visual feature of peeled apple? Our contributions are as follows:

- We propose a novel approach, OADis, to disentangle attribute and object visual features, where visual embedding for peeled is distinct and independent of embedding for apple.
- We compose unseen pairs in the visual space using the disentangled features. Following Compositional Zero-shot Learning (CZSL) setup, we show competitive improvement over prior works on standard datasets [24, 56].
- We propose a new large-scale benchmark for CZSL using an existing attribute dataset VAW [45], and show that OADis outperforms existing baselines.

2. Related Work

Visual Attributes. Visual attributes have been studied widely to understand visual properties and low-level semantics of objects. These attributes help further improve on various downstream tasks such as object detection [5, 11, 14, 15, 26, 36], action recognition [1, 3, 4, 16, 18, 34], image captioning [25, 40], and zero-shot and semi-supervised classification [2, 10, 11, 27, 39, 41, 50]. Similar to multi-class classification for objects, initial work for attribute understanding used discriminative models [26, 42], without understanding attributes. Other works [8, 15, 23, 32] explored the relation between the same attributes and different objects, to learn visual attributes. Particularly, disentangling object features from attribute features are explored in [17, 19]. Although, these works use clustering and probabilistic models to learn the attributes of objects.

Compositional Zero-shot Learning. Concept of compositional learning was first introduced in Recognition by Parts [22]. Initially, [35] employed this concept for objects and attributes. Unlike zero-shot learning (ZSL), CZSL requires the model to learn to compose unseen concepts from already learned primitive components. [8, 35] proposed separate classifiers for primitive components, and merged all into a final classifier. Most prior works use linguistically inspired auxiliary loss terms to regularize training for embedding space, such as: [38] models attributes as a linear transformation of objects, [30] uses rules of symmetry for understanding states, and [55] learns composition and decomposition of attributes hierarchically. Another set of studies uses language priors to learn unseen attribute-object pairs, either in feature space or with multiple networks [31, 46, 52]. Other recent works use graph structure to leverage information transfer between seen to unseen pairs using Graph Convolutional Networks [33, 37], and [54] uses key-query based attention, along with modular network with message passing for learning relation between primitive concepts.

3. Object Attribute Disentanglement (OADis)

Contrary to prior works [30, 37, 38, 55], we explicitly focus on separating attributes and object features in the visual space. More precisely, TMN [46] uses word embeddings to generate attention layers to probe image features corresponding to a given pair. GraphEmbedding [37] exploits the dependency between word embeddings of the labels, and HiDC [55] mainly uses word embeddings to compose novel pairs and generate more examples for their triplet loss. To the best of our knowledge, none of the existing works have explored visual feature disentanglement of attributes and objects. We hypothesize that attribute and object visual features can be separated when considering visual feature similarities and differences between image pairs. Composing these disentangled elements help regularize the common embedding space to improve recognition performance.
More concretely, we take cues from [17] and [35, 55], to learn to compose unseen attribute-object pairs leveraging visual attributes based on auxiliary losses.

3.1. Task Formulation

We follow the conventional Compositional Zero-shot Learning (CZSL) setup, where distinct attribute-object compositions are used at training and testing. Each image \( I \) is labeled with \( y = y_{\text{attr}, \text{obj}} \in \mathcal{Y} \), where \( y_{\text{attr}} \) and \( y_{\text{obj}} \) are respectively the attribute and object label. The dataset is divided into two parts, seen pairs \( Y^s \) and unseen pairs \( Y^u \), such that \( Y^s = Y^s \cup Y^u \), \( Y^s \cap Y^u = \emptyset \). Although \( Y^u = y_{\text{attr}, \text{obj}} \in \mathcal{Y}^u \) consists of attribute \( y_{\text{attr}} \) and object \( y_{\text{obj}} \) that are never seen together in training, they are separately seen. We employ the Generalized CZSL setup defined in [46], which has seen \( Y^s \) and unseen pairs \( Y^u \) in the validation and test sets as detailed in Table 1. As shown in Figure 2, for image \( I \), with label peeled apple, we choose two additional images: one with same object and different attribute \( I_{\text{obj}} \) (e.g., sliced apple), and another image with same attribute and different object \( I_{\text{attr}} \) (e.g., peeled orange). Note that the subscript of image symbol, e.g., \( \text{attr} \) in \( I_{\text{attr}} \), shows similarity with \( I \), whereas superscript denotes seen and unseen sets.

3.2. Disentangling Visual Features

We extract image and label embedding features from pre-trained networks (ResNet [21] and GloVe [44]). As seen in Figure 2, we use Image Encoder (IE) and Object Conditioned Network (OCN), for image and word embedding features respectively. Similar to [38], we use Label Embedder (LE) as an additional FC-Layer for the image feature. LE and OCN learn image and word embeddings and embed those in a common pair embedding space. Next, visual similarity between \( I \) and \( I_{\text{obj}} \) is computed using Object Affinity Network, which extracts visual features for object, \( v_{\text{obj}} \). Whatever is not similar is considered dissimilar. Hence, visual features of \( I_{\text{obj}} \) that are least similar to visual features of \( I \) are considered as the attribute feature \( v'_{\text{attr}} \) in \( I_{\text{obj}} \), which is sliced in this example. Similarly, Attribute Affinity Network takes \( I \) and \( I_{\text{attr}} \) and extracts visual similarity feature \( v_{\text{attr}} \) for peeled, and dissimilar visual features of \( I_{\text{obj}} \), as object feature \( v'_{\text{obj}} \) for orange. The disentangled features are then used to compose seen and unseen pairs. We discuss the details in the following sections:

Image Encoder (IE). We use the second last layer before AveragePool of an ImageNet-pretrained ResNet-18 [13, 21] to extract features for all images. IE is a single convolutional layer that is shared across images \( I \), \( I_{\text{attr}} \) and \( I_{\text{obj}} \) to generate their image features, represented as \( f, f_{\text{attr}} \) and \( f_{\text{obj}} \) respectively, where each \( f \in \mathbb{R}^{49} \times n \) and \( n \) is the output dimension of IE.

Label Embedder (LE). Inspired by [38], our LE inputs spatial feature from ResNet [21], AveragePools and passes through a linear layer to extract final feature \( v_{\text{attr}, \text{obj}} \) for pair embedding, which has same dimension as the word embedding final feature \( v'_{\text{attr}, \text{obj}} \) extracted from Object Conditioned Network (OCN) (Figure 2). This is the main branch, and is used for input image \( I \) only.

Object Conditioned Network (OCN). This takes word embeddings of attribute \( \text{emb}_{\text{attr}} \) and object \( \text{emb}_{\text{obj}} \), concatenates the features and passes through multiple layers. Object-conditioned is named because a residual connection for the object feature is concatenated with the final attribute feature, and the output feature is \( v'_{\text{attr}, \text{obj}} \in \mathcal{Y} \). We discuss...
Our main contribution is the proposed affinity modules and compositional losses. Inspired by image captioning [9, 28, 29], OADis uses image similarities and differences to identify visual features corresponding to attributes and objects. Object Affinity Network (OAN) uses \( f \) and \( f_{obj} \), whereas Attribute Affinity Network (AAN) uses \( f \) and \( f_{attr} \). For brevity, we explain the AAN, while the OAN follows the same architecture. Reminded that both \( f \) and \( f_{attr} \) are visual features.

**Cosine Classifier (CosCls).** Analogous to compatibility function used in [33, 37], we use cross-entropy along with cosine similarity to get the final score for each pair. For visual features \( v_{attr, obj} \) (from LE), and composed word embeddings \( w_{attr, obj} \) (from OCN), CosCls provides logits for an image \( I \). For instance, let us assume \( v : X \rightarrow Z \) and \( w : Y \rightarrow Z \). Z is the common embedding space for word embeddings \( w \) and visual embeddings \( v \). Then classifier unit CosCls gives the score for label \( y \in Y^s \) is \( C \):

\[
\begin{align*}
    h(v, w) &= \cos(v, w) = \delta - \frac{v^T w}{\|v\| \|w\|} \\
    C(v, w) &= \sum_{y \in Y^s} e^{h(v, y)}
\end{align*}
\]

where \( \delta \) is the temperature variable. Each loss function uses same CosCls score evaluator, with different inputs.

**Object and Attribute Similarity Modules.** Similar to [53], which computes attention between word concepts with corresponding visual blocks, we compute attention between two images \( I \) and \( I_{attr} \). Since both images have the same attribute, i.e., peeped, our affinity network learns visual similarity between the images, which represents the attribute. Similarity matrix \( S \) is the cosine similarity between \( f \) and \( f_{attr} \), such that \( S \in \mathbb{R}^{49 \times 49} \) as:

\[
S = \frac{f^T f_{attr}}{\|f\|_2 \|f_{attr}\|_2} \quad (3)
\]

where element \( s_{ij} \) represents the similarity between \( i \)th element of \( f \) with \( j \)th element of \( f_{attr} \). Moreover, let \( s_{ij} \) and \( s_{ij} \) represent the \( i \)th row and \( j \)th column of \( S \) respectively. Then, \( s_{ij} \) captures the similarity of all the elements in \( f_{attr} \) with respect to \( j \)th element of \( f \). To know the most similar element among \( f_{attr} \) with respect to \( i \)th element of \( f \), we can take a row-wise softmax over \( S \). Similarly, for \( j \)th element of \( f_{attr} \), column \( s_{ij} \) represents the similarity with all the elements of \( f \). Using a column-wise softmax, we can interpret the most similar and least similar element of \( f \) with respect to \( j \)th element of \( f_{attr} \), as shown in Figure 3. Therefore, by applying column-wise and row-wise softmax, we get two matrices, \( A \) and \( A' \) (where \( A, A' \in \mathbb{R}^{d \times d}, d = 49 \)),

\[
A_i = \sum_{j=1}^{d} e^{\lambda s_{ij}} \quad \text{and} \quad A_j' = \sum_{i=1}^{d} e^{\lambda s_{ij}}, \quad (4)
\]

where \( \lambda \) is the inverse temperature parameter. We compute row and column sum for \( A \) and \( A' \) respectively, to get final
similarity maps, \( m \) and \( m_{\text{attr}} \),

\[
m_j = \sum_{i=1}^{d} A_{ij} \quad \text{and} \quad m_{\text{attr}} = \sum_{j=1}^{d} A'_{ij}. \tag{5}
\]

Similarly, the difference between these two images \( f \) and \( f_{\text{attr}} \) is the object label, \( y_{\text{obj}} \). Hence, we use the negative of \( S \) as the image difference, denoted as \( S' \). Then, difference of \( f_{\text{attr}} \) with respect to \( f \) would be row-wise softmax of difference matrix, denoted by \( D \). Hence, by performing column-sum over \( D \), we get difference map, \( m_{\text{obj}} \),

\[
D_j = \frac{e^{v'_{\text{obj}}}}{\sum_{i=1}^{d} e^{v'_{\text{obj}}}} \quad \text{and} \quad m'_{\text{obj}} = \sum_{j=1}^{d} D_{ij}. \tag{6}
\]

The final disentangled features for attribute \( v'_{\text{attr}} \) and object \( v'_{\text{obj}} \), for both AAN and OAN, can be computed as:

\[
v'_{\text{attr}} = m \cdot f_{\text{attr}} + m_{\text{attr}} \cdot f \quad \text{and} \quad v'_{\text{obj}} = m'_{\text{obj}} \cdot f_{\text{attr}}, \tag{7}
\]

More details using a toy example can be seen in Figure 3. Using concatenation of \( v'_{\text{attr}} \) and \( v'_{\text{obj}} \) along with a single linear layer, composes the pair peeled apple, represented by \((v'_{\text{attr}}, v'_{\text{obj}})\). Similarly, the disentangled visual features \( v'_{\text{attr}} \) and \( v'_{\text{obj}} \) are used to compose unseen pair sliced orange, and is represented as \((v'_{\text{attr}}, v'_{\text{obj}})\).

### 3.3. Embedding Space Learning objectives

As shown in Figure 3b, we learn three embedding spaces: (1) attribute space, (2) object space, and (3) attribute-object pair space. The attribute and object spaces are used for disentangling the two, whereas pair embedding is used for final pair composition and inference. OADis has separate loss functions for disentangling and composing. All loss functions are expressed in terms of CosCls defined previously.

The loss function for main branch, \( \mathcal{L}_{\text{cls}} \) uses combined visual feature \( v_{\text{attr,obj}} \) from LE and word embedding feature \( w_{\text{attr,obj}} \) from OCN. \( \mathcal{L}_{\text{cls}} \) is used for the pair embedding space. Similarly, \( \mathcal{L}_{\text{attr}} \) and \( \mathcal{L}_{\text{obj}} \) are used to learn the visual attribute and object feature, in their respective embedding spaces. \( \mathcal{L}_{\text{attr}} \) pushes the visual feature of attribute, closer to the word embedding. \( \mathcal{L}_{\text{obj}} \) does the same for objects in object embedding space Figure 3b. These losses cover the concept of disentanglement, and can be represented as:

\[
\begin{align*}
\mathcal{L}_{\text{cls}} &= C(v_{\text{attr,obj}}, w_{\text{attr,obj}}) \\
\mathcal{L}_{\text{attr}} &= C(v_{\text{attr}}, w_{\text{attr}}) \\
\mathcal{L}_{\text{obj}} &= C(v_{\text{obj}}, w_{\text{obj}}) \tag{8}
\end{align*}
\]

For composition, we use \( \mathcal{L}_{\text{seen}} \) and \( \mathcal{L}_{\text{unseen}} \). Among the images seen (\( I_1, I_{\text{attr}}, \) and \( I_{\text{obj}} \)), disentangled features \( v_{\text{obj}} \) and \( v_{\text{attr}} \) composes the same pair as \((v_{\text{attr}}, v_{\text{obj}})\), which we refer to as the seen composition. Note that \((v_{\text{attr}}, v_{\text{obj}})\) is different from \(v_{\text{attr,obj}}\), as the former is hallucinated feature with combination of disentangled attribute and object visual features, and latter is the combined visual feature extracted with LE. Here, we use \( \mathcal{L}_{\text{seen}} \) which takes the composition of disentangled features and learns to put the composition closer to \( w_{\text{attr,obj}} \). Moreover, the dissimilarity aspect from OAN and AAN extracts \( v'_{\text{attr}} \) and \( v'_{\text{obj}} \), which composes an unseen pair \((v'_{\text{attr}}, v'_{\text{obj}})\). We use \( \mathcal{L}_{\text{unseen}} \) as unseen loss since the hallucinated composition is never seen among \( I_1, I_{\text{attr}}, \) and \( I_{\text{obj}} \),

\[
\begin{align*}
\mathcal{L}_{\text{seen}} &= C((v_{\text{attr}}, v_{\text{obj}}), w_{\text{attr,obj}}) \\
\mathcal{L}_{\text{unseen}} &= C((v'_{\text{attr}}, v'_{\text{obj}}), w'_{\text{attr,obj}}) \tag{9}
\end{align*}
\]

The combined loss function \( \mathcal{L} \) is minimized over all the training images, to train OADis end-to-end. The weights for each loss (\( \alpha \)) are empirically computed:

\[
\mathcal{L} = \mathcal{L}_{\text{cls}} + \alpha_1 \mathcal{L}_{\text{attr}} + \alpha_2 \mathcal{L}_{\text{obj}} + \alpha_3 \mathcal{L}_{\text{seen}} + \alpha_4 \mathcal{L}_{\text{unseen}}.
\]

### 4. Experiment

#### 4.1. Datasets and Metrics

We show results on three datasets: MIT-states [24], UT-Zappos [56], and a new benchmark for evaluating CZSL on images of objects in-the-wild, referred as VAW-CZSL. VAW-CZSL is created based on images with object and attribute labels from the VAW dataset [45]. Both MIT-states [24] and UT-Zappos [56] are common datasets used for this task in previous studies. MIT-states covers wide range of objects (\(i.e.,\) laptop, fruits, fish, room, etc.) and attributes (\(i.e.,\) mossy, dirty, raw, etc.), whereas UT-zappos has fewer objects (\(i.e.,\) shoes type: boots, slippers, sandals) and fine-grained attributes (\(i.e.,\) leather, fur, etc.).

**Proposed New Benchmark.** While experimenting with MIT-states [24] and UT-Zappos [56], we found several shortcomings with these datasets and discovered issues across all baselines using these datasets:

- Both datasets are small, with a maximum of 2000 attribute-object pairs and 30k images, leading to overfitting fairly quickly.
- Random seed initialization makes performance fluctuate significantly (0.2-0.4% AUC). Moreover, [2] found 70% noise in human-annotated labels on MIT-States [24].
- A new dataset C-GQA was introduced in [37], but the dataset is still small and we found a lot of discrepancies (kindly refer to the suppl.).
Table 2. We show results on MIT-states [24] and UT-Zappos [56]. Following [37, 46], we use AUC in % between seen and unseen compositions with different bias terms, along with Val, Test, attribute and object accuracy. HM is Harmonic Mean. OADis consistently outperforms on most categories with significant increment.

<table>
<thead>
<tr>
<th>Model</th>
<th>Val@1</th>
<th>Test@1</th>
<th>HM</th>
<th>Seen</th>
<th>Unseen</th>
<th>Attribute</th>
<th>Object</th>
</tr>
</thead>
<tbody>
<tr>
<td>AttrOp [38]</td>
<td>2.5</td>
<td>2.0</td>
<td>10.7</td>
<td>16.6</td>
<td>18.4</td>
<td>22.9</td>
<td>24.7</td>
</tr>
<tr>
<td>LabelEmbed+ [38]</td>
<td>3.5</td>
<td>2.3</td>
<td>11.5</td>
<td>16.2</td>
<td>21.2</td>
<td>25.6</td>
<td>27.5</td>
</tr>
<tr>
<td>TMN [46]</td>
<td>3.3</td>
<td>2.6</td>
<td>11.8</td>
<td>22.7</td>
<td>17.1</td>
<td>21.3</td>
<td>24.2</td>
</tr>
<tr>
<td>Symnet [30]</td>
<td>4.5</td>
<td>3.4</td>
<td>13.8</td>
<td>24.8</td>
<td>20.0</td>
<td>26.1</td>
<td>25.7</td>
</tr>
<tr>
<td>CompCos [33]</td>
<td>6.9</td>
<td>4.8</td>
<td>16.9</td>
<td>26.9</td>
<td>24.5</td>
<td>28.3</td>
<td>31.9</td>
</tr>
<tr>
<td>GraphEmb [37]</td>
<td>7.2</td>
<td>5.3</td>
<td>18.1</td>
<td>28.9</td>
<td>25.0</td>
<td>27.2</td>
<td>32.5</td>
</tr>
</tbody>
</table>

OADis | 7.6 | 5.9 | 18.9 | 31.1 | 25.6 | 28.4 | 33.2 |

OADis 7.6 5.9 18.9 31.1 25.6 28.4 33.2

Table 3. We show results on VAW-CZSL. Since it is a much more challenging dataset, with significantly large number of compositions, to discriminate performance among different baseline, we show top-3 and top-5 AUC (in %) for Val and Test sets.

<table>
<thead>
<tr>
<th>Model</th>
<th>Val@1</th>
<th>Test@1</th>
<th>HM</th>
<th>Seen</th>
<th>Unseen</th>
<th>Attribute</th>
<th>Object</th>
</tr>
</thead>
<tbody>
<tr>
<td>AttrOp [38]</td>
<td>1.4</td>
<td>2.5</td>
<td>1.4</td>
<td>2.6</td>
<td>9.1</td>
<td>16.4</td>
<td>11.7</td>
</tr>
<tr>
<td>LabelEmbed+ [38]</td>
<td>1.5</td>
<td>2.8</td>
<td>1.6</td>
<td>2.8</td>
<td>9.8</td>
<td>16.2</td>
<td>13.2</td>
</tr>
<tr>
<td>TMN [46]</td>
<td>2.3</td>
<td>3.9</td>
<td>2.3</td>
<td>3.9</td>
<td>12.2</td>
<td>19.1</td>
<td>15.8</td>
</tr>
<tr>
<td>Symnet [30]</td>
<td>2.2</td>
<td>3.9</td>
<td>2.3</td>
<td>4.0</td>
<td>11.9</td>
<td>19.9</td>
<td>15.4</td>
</tr>
<tr>
<td>CompCos [33]</td>
<td>3.1</td>
<td>5.6</td>
<td>3.2</td>
<td>5.6</td>
<td>14.2</td>
<td>23.9</td>
<td>18.0</td>
</tr>
<tr>
<td>GraphEmb [37]</td>
<td>2.7</td>
<td>5.3</td>
<td>2.9</td>
<td>5.1</td>
<td>13.0</td>
<td>23.4</td>
<td>16.8</td>
</tr>
</tbody>
</table>

OADis | 3.5 | 6.0 | 3.6 | 6.1 | 15.2 | 24.9 | 18.7 |

OADis 3.5 6.0 3.6 6.1 15.2 24.9 18.7

4.2. Results and Discussion

Baselines. We compare with related recent and prominent prior works: AttrOp [38], LabelEmbed+ [38], TMN [46], Symnet [30], CompCos [33] and GraphEmb [37]. We do not compare with BMP [54], since it uses the concatenation of features from all four ResNet blocks (960-d features), resulting in higher input features and the number of network parameters than all other setups. Moreover, GraphEmb [37] is state-of-the-art; hence, comparing with that makes our work comparable to other baselines that [37] already outperforms. To be consistent, we state the performance of all models (including GraphEmb [37]) using frozen backbone ResNet without fine-tuning the image features, and using GloVe [44] for the object and attribute word embeddings. Before passing through backbone, training images are augmented with horizontal flip and random crop. Compared to other baselines, OADis uses convolutional features rather than AvgPooled, since it is easier to segregate visual features in the spatial domain for attributes and objects. Moreover, other studies [33,37] have also used additional FC layers on top of IE, which we argue makes it fair for us to use pre-pooled features for OADis.

Results on MIT-States. MIT-states has considerable label noise [2], but still is a standard dataset for this task. We show significant improvement on this dataset (reported in Table 2), from previous state-of-the-art GraphEmb, which has 7.2 Val AUC and 5.3 Test AUC. Note that we do not report GraphEmb results with fine-tuning backbone, as we find it incomparable with other baselines that did not incorporate fine-tuning as part of their proposed methods. Overall, our model performs significantly better than GraphEmb on all metrics.

Results on UT-Zappos. Similar improvement trends hold for UT-Zappos as well (see Table 2). Although, as explained for GraphEmb, it is difficult to balance the best performance for Val and Test set in this dataset. The problem is that $7/36$ ($\sim 20\%$) attributes in Test set do not appear in Val set. Hence, improving Val set AUC, does not necessarily improve Test AUC for UT-Zappos. Similar trend can be seen for other baselines: CompCos has best Val AUC, but does not perform well on Test set, compared to TMN and Symnet. Even GraphEmb in their final table show the frozen backbone network has much lower performance than TMN. However, OADis performs well on UT-Zappos overall, with $\sim 4.0$ improvement for Val and Test AUC, HM, unseen and object accuracy.
Is visual disentangling actually happening? Visual disentanglement in feature space is challenging to visualize since: (a) parts of an image for attributes and objects are hard to distinguish, as attributes are aspects of an object; (b) OADIs is end-to-end trained with losses to disentangle features for attribute and object embeddings, which is separate from pair embedding space. Inspired by [30, 38], we show a few qualitative results in Figure 5. Using all training images, prototype features $V_{attr}$ for each attribute can be computed by averaging features for all images containing that attributes $v_{attr}$ using AAN. Similarly, with OAN, prototype object features are also computed. For each test image, we find top-3 nearest neighbors from these prototype fea-
Figure 5. Qualitative results showing top 3 attributes and objects from test images, using prototype disentangled features computed on training data.

Limitations. Despite OADis outperforming prior works on all benchmarks, we still notice some outstanding deficiencies in this problem domain. First, similar to [37], OADis often struggles on images containing multiple objects, where it does not know which object to make prediction on. One possible solution is to utilize an object-conditioned attention that allows the model to focus and possibly output attribute for multiple objects. Second, from qualitative studies on VAW-CSZL, we notice there are multiple cases where OADis makes the correct prediction but is considered incorrect by the image label. This is due to the fact that objects in-the-wild are mostly multi-label (containing multiple attributes), which none of the current single-label benchmarks have attempted to address.

4.3. Ablation Studies

In this section, we show experiments to support our design choices for OADis. All the ablations are done for MIT-states [24], for one random seed initialization, and are consistent for other datasets as well. Empirical results for λ, δ and different word embeddings can be found in suppl.

Why Object-Conditioned Network? Label Embedder [38] uses a linear layer and concatenates word embeddings for attributes and objects. We experiment with other networks: MLP with more parameters with two layers and ReLU and Object-conditioned network that uses a residual connection for object embedding. Our intuition is that same attribute contributes differently to each object, i.e., ruffled bag is very different from ruffled flower. Hence, attributes are conditioned on object. Adding a residual connection for object embeddings to the final attribute embedding helps condition the attribute. We empirically demonstrate that object-conditioning helps in Table 5 (refer to the suppl.).

To augment or not to augment? Augmentation is a common technique to reduce over-fitting and improve generalization. Surprisingly, prior works do not use any image augmentation. OADis without augmentation gives 6.7% AUC on Val and 5.1% AUC on Test set for MIT-states. Hence, we use augmentation for OADis and re-implemented rest of the baselines in Table 2, showing that augmentation helps improving all methods ~1.0-1.5% AUC. We use horizontal flip and random crop as augmentation.

4.4. Qualitative results

To qualitatively analyze our hallucinated compositions, we perform a nearest neighbor search on all three datasets. We pick the unseen compositions composed using the disentangled features, and find their top-5 nearest neighbors from the validation and test set. Figure 4(a) illustrates a few of our results. Note that these pairs are never seen in training. Based on the hallucinated compositions of disentangled attributes and objects, we are able to retrieve samples from these unseen compositions.

In Figure 4(b), we show the top-3 predictions of OADis on VAW-CSZL. Column 1 shows results for seen, and columns 2 and 3 show unseen compositions, with the ground-truth label on top (bold black). In all examples, our top-3 predictions describe the visual content of the images accurately, even though in many cases the ground-truth label is not predicted in top-1. For column 3, we purposely show examples where our model predictions totally differ from the ground-truth label, but still correctly describe the visual information in each image. Similar to [37], this explains the multi-label nature of object-attribute recognition, and why we report top-3 and top-5 metrics for the VAW-CSZL benchmark.

5. Conclusion

In this work, we demonstrated the ability to disentangle object and attribute in the visual feature space, that are used for hallucinating novel complex concepts, as well as regularizing and obtaining a better object-attribute recognition model. Through extensive experiments, we show the efficacy of our method, and surpass previous methods across three different benchmarks. In addition, we also propose a new benchmark for the compositional zero-shot learning task with images of objects in-the-wild, which we believe can help shift the focus of the community towards images in more complex scenes. Finally, we also highlight limitations of our work, including the notable problem of multi-label in object attributes, which we hope would encourage future works to start tackling CSZL for more realistic scenarios.

Acknowledgements. This work was supported by the Air Force (STTR awards FA865019P6014, FA864920C0010), DARPA SAILON program (W911NF2020009) and gifts from Adobe collaboration support fund.


[38] Tushar Nagarajan and K. Grauman. Attributes as operators: Factorizing unseen attribute-object compositions. In *ECCV*, 2018. 2, 3, 6, 7, 8


