Effectively Leveraging Attributes for Visual Similarity Supplementary

Samarth Mishra*[1] Zhongping Zhang*[1] Yuan Shen[2] Ranjitha Kumar[2]
Venkatesh Saligrama[1] Bryan A. Plummer[1]
[1]Boston University [2]University of Illinois, Urbana-Champaign
[1]{samarthm, zpzhang, srv, bplum}@bu.edu [2]{yshen47, ranjitha}@illinois.edu

A. Similarity as edge prediction

To accommodate training with the graph encoder (GE), we formulate learning similarity as an edge prediction task on graph with nodes as images as done in [3]. Each node in the graph represents an image and the edges represent ground truth similarity information. Edges are stored as an adjacency matrix $\boldsymbol{A} \in \mathbb{R}^{N \times N}$ where $\boldsymbol{A}_{i,j} = 1$ if there exists an edge between node i and node j, *i.e.*, if the images corresponding to the nodes are labelled to be similar in the dataset. $\boldsymbol{A}_{i,j} = 0$ otherwise. Note that this is a general formulation and encoders other than the GE can be trained this way. The graph defined just does not play any role in the model's outputs in that case.

B. Similarity prediction for end tasks

Fashion Compatibility Prediction. The goal of this task is to predict whether a given set of items is compatible. We compute the compatibility score of an outfit (group of items) by averaging the likelihood of edge existence over all pairs of items in the outfit. Area under a receiver operating characteristic curve (AUC) is used here to evaluate the performance on this task. For this task, at inference time, a model with GE uses no ground truth context information in the form of edges since none is available.

Fill-in-the-blank (FITB). The FITB task is to select the best compatible item given a partial outfit and a set of candidate items. Concretely, a question consists of n items $q_1, q_2, ..., q_n$. Each question has m choices $o_1, o_2 ..., o_m$. Our models compute the compatibility score s between all item pairs and s_{ij} represents the score between q_i and o_j. The score of o_j is calculated as $\sum_{i=0}^{n} s_{ij}$. The item that obtains the highest score is chosen as our final candidate. Performance is measured in terms of answer accuracy. When a GE is used, edges are added to represent compatibility between the items in the question, *i.e.*, edges are added between each q_i and q_j for $i, j \in [n]$, at inference time.

Few shot classification. Each 5-way 5-shot classification episode has 5 support examples or 5 "shots" for each of the 5 classes and 16 query examples from each class. The task is to classify query examples into one of the 5 classes. For a given query example, the model predicts the probability of existence of an edge between it and the support examples for each of the classes. The average edge probability over support examples for a given class is treated as a score of belongingness to that class. The model then predicts the class of the given example as the one with the highest score. The accuracy of the model for an episode is its accuracy in classifying the 16 x 5 = 80 query examples.

C. Implementation details

In this section we describe the training details of our models and different baselines referred to in Section 4.1 in the main paper.

C.1. Pairwise Attribute-informed similarity Network (PAN)

Our PAN model uses image features encoded by a pretrained CNN. We use a Resnet-18 [5] feature extractor for few shot classification on CUB and a Resnet-50 for fashion compatibility prediction on the Polyvore Outfits dataset. The Resnet-50 used is pretrained on Imagenet [4]. However, we use the feature extractor from a Siamese Network (details below in section C.2) for CUB since the novel split of CUB shares some images with the Imagenet dataset. The size of the image input to these feature extractors is 224 x 224, and the features output from Resnet-18 are 512 dimensional while those output from Resnet-50 are 2048 dimensional.

For models evaluated on CUB, the graph encoder contains a 3 layer graph convolutional network (GCN) where the features output at each layer are 350 dimensional. The same for Polyvore Outfits is a 2-layer GCN, the features being 200 dimensional at the output of each layer.

In the training of models using the graph encoder, we use dropout with drop probability of 0.5 at each GCN layer. As an additional method of regularization, in each epoch of training, each edge in the adjacency matrix $\boldsymbol{A}$ is dropped with a probability 0.15.

For training the PAN models, we use an Adam [6] optimizer with learning rate 0.001. The models for few shot classification on CUB are trained for 1000 epochs and the ones for fashion compatibility prediction on Polyvore Outfits for 4000 epochs, validation being done after each epoch of training.

C.2. Siamese Network for few shot classification

We train a Siamese Network [1, 7] for few shot classification on CUB, primarily as a CNN feature extractor for the PAN models. Training examples are obtained by sampling triplets of images ($\mathbf{x}$, $\mathbf{y}$, $\mathbf{z}$), where $\mathbf{x}$ and $\mathbf{y}$ belong to the same class and $\mathbf{z}$ belongs to a different class. The network is trained to minimize the following triplet loss

$$\mathcal{L} = \max(||f(\mathbf{x}) - f(\mathbf{y})||_2 - ||f(\mathbf{x}) - f(\mathbf{z})||_2 + \alpha, 0) \tag{1}$$

where f is the ResNet-18 feature extractor, $||\cdot||_2$ is the ℓ_2 norm, and α is a margin parameter.

We use the same splits for CUB as [2], 100 classes in the base split and 50 each in the novel and val splits. In one epoch of training, the network sees all possible positive pairs of images in the training dataset (base split of CUB), with negative examples sampled randomly from one of the remaining classes. We trained the network for 200 epochs, validating every 2 epochs using average accuracy on 100 few shot classification episodes drawn from the val split. We used Adam [6] optimizer with a learning rate of 0.001 and the margin parameter α was chosen to be 0.2. A mini-batch size of 96 triplets was used for training. We used random resizing and cropping, color jittering and random horizontal flips as data augmentation for training.

C.3. Single batch training

When training with a single batch (*i.e.*, the entire dataset is used for training at once), fine-tuning the CNN with limited GPU memory is not possible. Thus, to minimize the GPU memory required for each image, we use a pretrained CNN to get fixed-length feature representations for images. For methods like a Siamese Network and "X + Attr Multitask" we train a classifier implemented as a fully connected layer to predict links between images, using a single batch consisting of pre-extracted features from the entire training set. These features are extracted from a CNN trained using mini-batches either as a Siamese Network or with an additional attribute prediction head, as is done for "X + Attr Multitask".

D. PAN — Behavior with different number of similarity conditions

In Figure 1 we plot the performance of the PAN-supervised and the PAN-unsupervised models with different numbers of similarity conditions. For supervision, we randomly shuffled the order of attributes and selected the first n.

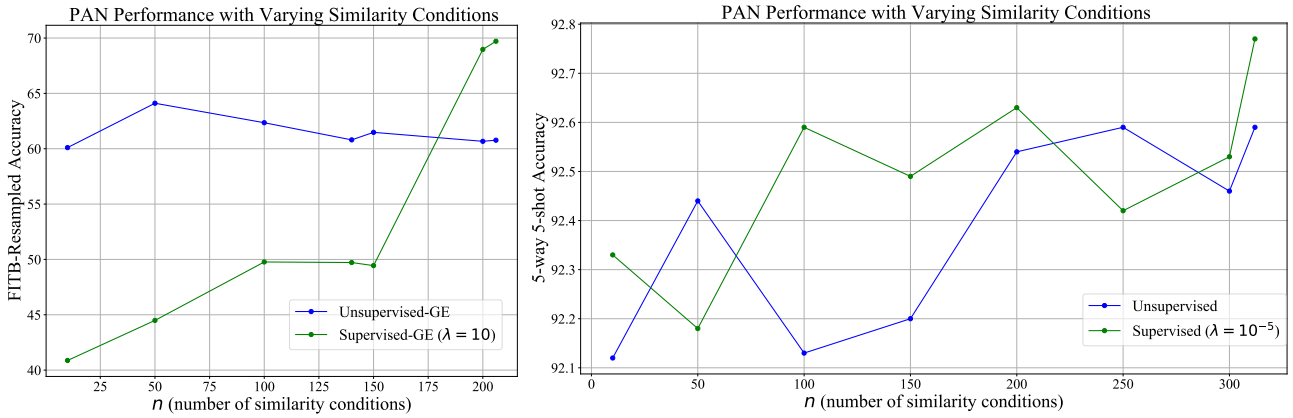
In Figure 1(a), where we plot only the FITB accuracy of the models on the *resampled* Polyvore Outfits split, we see that when there are few supervised attributes, the performance of the fully supervised model is poor. However, when the similarity conditions are allowed to be free (in the unsupervised model), the performance is higher. The supervised model starts performing better as the number of different attributes that are annotated increases. The increase in performance of the unsupervised model with increasing number of similarity conditions is much less pronounced, thus showcasing the role of supervision using attribute annotations in fashion compatibility prediction performance on Polyvore Outfits.

A different trend is seen in the case of few shot classification on CUB (in Figure 1(b)), where the unsupervised model has performance close to that of the fully supervised model. There is a general increase in performance with more similarity conditions in both the unsupervised and the supervised models, but the performance is high enough with a few similarity conditions. This indicates that good performance on the few shot classification task can be achieved with a few unsupervised similarity conditions, and supervision using attributes provides a small boost most of the time. The relatively good performance on this task may be the result of the relative simplicity of the task, where few-shot classification can be thought of as trying to simply match attributes. In comparison, in fashion compatibility, the relationship between attributes is far more complicated, as discussed in the introduction of our paper. Images with different attributes can be deemed highly similar (or more compatible), while some attribute combinations may indicate dissimilarity/incompatibility even if subsets of them would normally indicate similarity. This requires a far more complex function to reason about attributes, which our results seem to indicate is difficult to capture without supervision.

We also find that using supervised attributes often requires a critical mass, *i.e.*, a variety of attributes are required to outperform the unsupervised model reliably. Our results on Polyvore Outfits demonstrate that these need not be dense annotations. Attributes on that dataset were automatically labeled after curating a set of visual concepts manually from common words that appear in the items' description and/or title, resulting in a very sparsely annotated dataset.

E. Choosing number of similarity conditions for unsupervised models

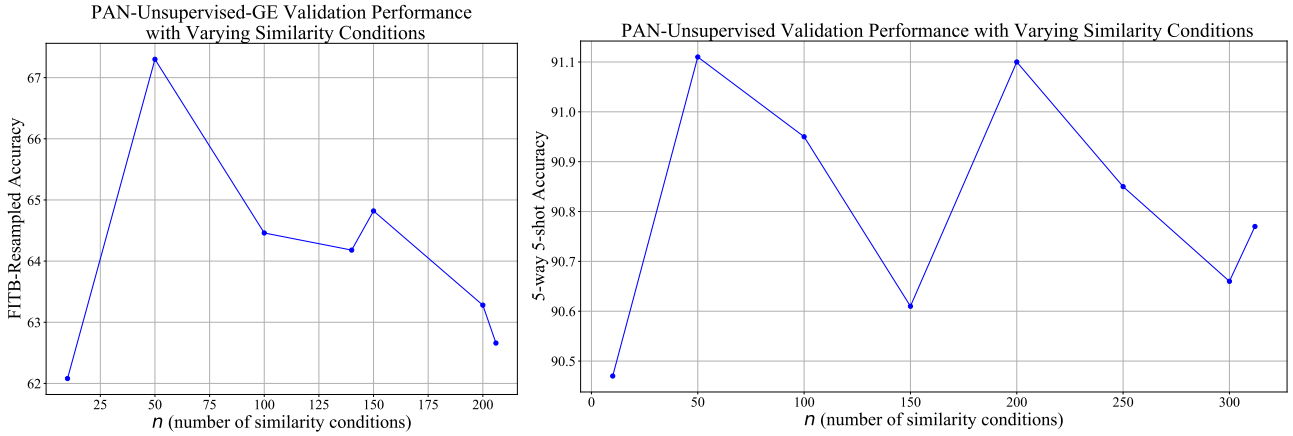
Figures 2(a) & 2(b), show the validation set performances of the unsupervised models in the two tasks—predicting compatibility on Polyvore Outfits and few shot



(a) FITB accuracy on Polyvore Outfits

(b) Few shot classification on CUB

Figure 1: **Comparing performance with different number of similarity conditions on the test set.** For providing supervision with n attributes (where n may be less than the total number of attributes labelled in the dataset), the first n of a fixed random order of attributes were chosen.



(a) FITB accuracy on Polyvore Outfits

(b) Few shot classification on CUB

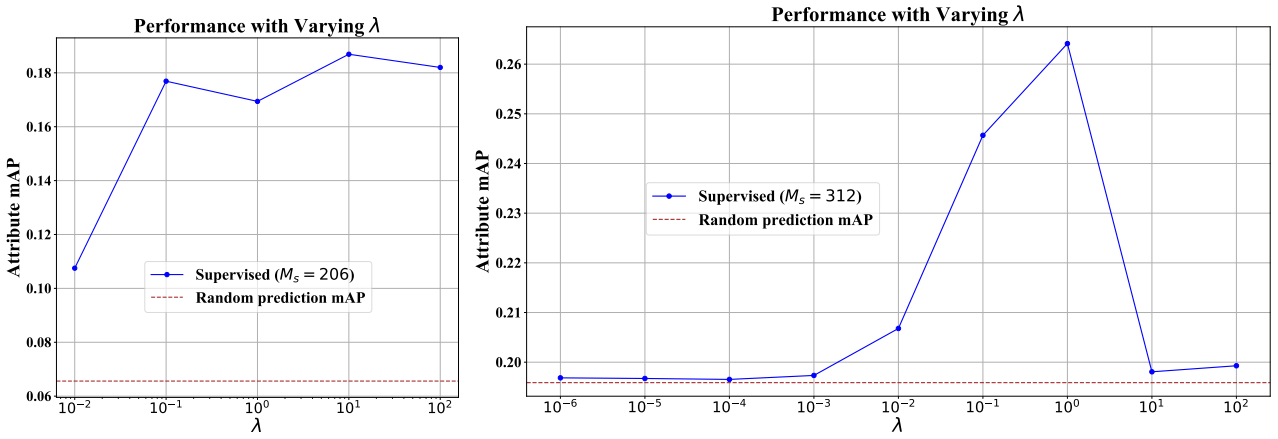
Figure 2: **Validation set performance** of the PAN models with unsupervised similarity conditions

classification on CUB. We chose the models with the best validation accuracy for comparison in Tables 1, 2 and 4 of the main paper. In particular, for both tasks, we found that the best performance was achieved at 50 unsupervised similarity conditions ($M = 50$) and beyond that the model seems to overfit.

F. Attribute Recognition Experiments

We train the PAN models using supervision from the logical OR of attributes, and have seen that this helps in improving similarity prediction performance. Here, we inspect if the supervision results in meaningful predictions along different supervised similarity conditions. On pairs of images from the test split of the data, we compute the average precision (AP) of the attribute scores output by our model where the ground truth comes from the OR of the attribute labels. Missing attribute labels and their score predictions are excluded. The mean AP (mAP) is then computed by averaging the APs over the different attributes.

In Figure 3, we compare the mAP values as described above for our PAN-Supervised model using different values of the hyperparameter λ. In both Figures 3(a) and 3(b), the brown dotted horizontal line is the mean average precision of a set of scores generated uniformly at random from the range $[0, 1]$. From both figures, we see that increasing the weight of attribute supervision increases the mAP. Com-



(a) Attribute recognition on Polyvore Outfits.

(b) Attribute recognition on CUB

Figure 3: **Mean average precision (mAP) of attribute prediction.**

bining the results of Figure 3 with those of Figure 3 in the main paper, on the Polyvore Outfits dataset we see a positive correlation between the end-task performance and the mAP on attributes. $\lambda = 10$ results in both the best end-task performance as well as the best mAP. This suggests that if we were to improve our model to better recognize joint attributes, we would improve compatibility predictions as well. On the CUB dataset however, we see a possible trade-off, where the attribute prediction performance is better for a higher lambda, but end-task performance is better for a lower value of lambda. This is in line with what we observe in Section D, where we see a smaller role of attribute supervision in improving few shot classification performance on CUB as compared to its effect in fashion compatibility prediction on Polyvore Outfits.

G. More questions

How useful are relevance weights to model performance? To find out, we experimented with an approach that predicts output similarity scores as PAN does and simply averages them to get the final similarity prediction between two images. On CUB, the 5-way 5-shot accuracy of this model was 85.25 ± 0.28 (over 3 runs with different random initializations) compared to 92.77 ± 0.30 with the relevance weighted sum. The FITB accuracy on the resampled split of PO, for it was 63.6 compared to 69.7 with the relevance weights. Thus simply averaging similarity predictions was found to perform poorer, highlighting the importance of using relevance weights.

What is the reason behind using the logical functions that were used for attribute combination f_a? We defined $f_a : \{0, 1\} \times \{0, 1\} \rightarrow [0, 1]$ as some function that maps two input attribute labels to a pair-wise label for supervision. The definition lends quite some flexibility in choosing what f_a can be, where possibilities include both binary (*i.e.* outputs only take on values 0 or 1), or fractional outputs (in the entire range $[0, 1]$). f_a can be either a non-parametric function, or could involve parameters (*e.g.* it could be a linear combination or could be non-linear like a multi-layer perceptron). To narrow down this range of choices, for our experiments, we restrict f_a to only binary values leaving the exploration of other choices to future work. Fractional outputs of the attribute combination f_a could have benefits in certain scenarios, for instance, it could be used for ranking images by some form of attribute strength as done in "relative attributes" [8].

Restricting f_a to binary outputs results in a total of 16 possibilities. Out of the 16, ruling out functions that are non-commutative, there are 8 possible choices remaining. This set largely consists of common logical functions. We can further narrow this down to 4 choices, which are the ones we experimented with in Section 5 of the main paper. Of the 8, the functions left out were the constants (always 0 or always 1) and $NAND$ and NOR. The first two do not provide information regarding the inputs, and the latter two are simply negatives of AND and OR respectively.

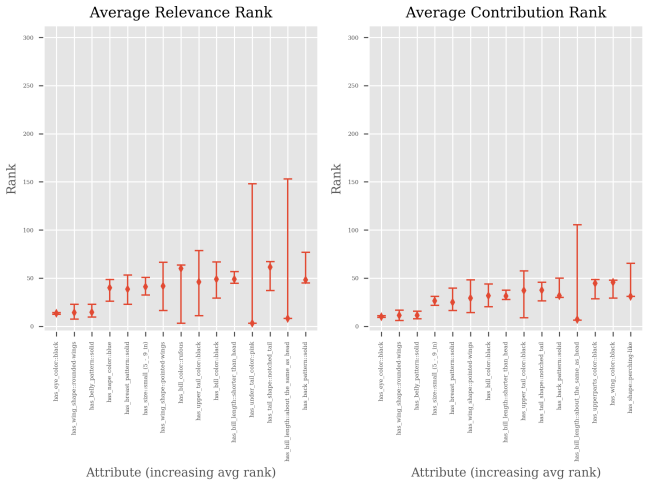
As a form of sanity-check, we also experimented with randomizing the attribute labels to ensure that supervision indeed helps (even if it is sparse), rather than there being some form of a regularizing effect from random labels. On the resampled split of PO, the FITB accuracy of a model that was trained with such random labels was 58.0, compared to 69.7 for a supervised model with OR attribute combination, verifying the role of attribute supervision.

Is it possible to train a model with both supervised and unsupervised similarity conditions? An excellent ques-

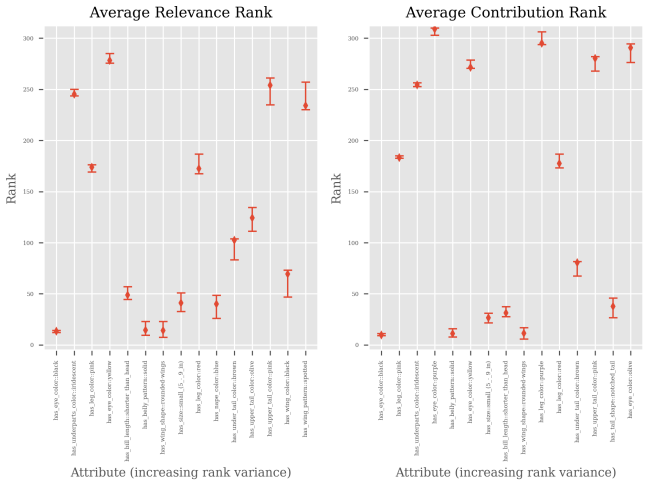
tion and certainly a possibility. However, in our experiments, we found that such a model could (ab)use its additional capacity to overfit to training data. On the CUB dataset, a "hybrid" model with 312 each of supervised and unsupervised similarity conditions could achieve a 5-way 5-shot accuracy of 91.38 ± 0.24. On the resampled split of PO, a model with 150 each supervised and unsupervised conditions had a FITB accuracy of 62.90. Note that in this experiment, we had to restrict to fewer than 206 attributes because of GPU memory constraints, but even so, this model had more (a total of 300) similarity conditions than both the PAN-Supervised and PAN-Unsupervised models reported in Table 1 of the main paper. We leave to future work, a more in-depth analysis of this model and a possible way of effectively utilizing both unsupervised and supervised similarity conditions.

Do the attribute importance scores always reflect human intuition? On the CUB dataset, we found some variance across the attribute importance scores for the same examples in different training runs starting from the same initializations. Fig 4 shows the variance in average ranks of different attributes across 3 different training runs of PAN-supervised. Ranks are computed based on both the relevance score (left) and the contribution to the similarity score (right) with the highest score being rank 1. The average ranks are computed by first computing for each run, the average rank of a given attribute in terms of its score from the PAN model across all pairs of images in the novel (or test) split of the dataset. This results in 3 average ranks (one per run). The average of these and the standard deviation (using error bars) are plotted in the figure.

We see that after the first three attributes sorted by rank (Fig. 4 (top)), the variance in the average rank increases by a lot. There are other attributes with smaller variance (Fig. 4 (bottom)) but at a much higher rank (meaning they have low relevance or contribution). Thus, in the case of CUB, PAN has learnt a relevance predictor, that has fairly high variance dependent on initialization for most attributes. This behavior is likely a consequence of two factors: first, attribute labels were sparse and noisy and so PAN learned to treat some similarity conditions as latent or unsupervised. For some attributes, this variance in relevance score is also possibly a consequence of that attribute not being useful for determining similarity. As a consequence, the model appeared to override these less important attributes to instead learn some general similarity metric. This would manifest itself as that attribute having low recognition performance coupled with high contribution to the overall similarity score in some runs, but not others, resulting in high variance in an attribute's overall rank across initializations. Thus, we found PAN's attribute relevance predictions to be too noisy to be reliably consistent with human intuition, especially for the CUB dataset.



(a) Attributes sorted according to increasing average rank



(b) Attributes sorted according to increasing variance

Figure 4: Attributes' mean ranks and standard deviations by relevance score and contribution to the similarity score over multiple training runs. See Sec. G for discussion. (Best viewed under zoom)

References

[1] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. Signature verification using a" siamese" time delay neural network. In *Advances in neural information processing systems*, pages 737–744, 1994. 2

[2] Wei-Yu Chen, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Frank Wang, and Jia-Bin Huang. A closer look at few-shot classification. *arXiv preprint arXiv:1904.04232*, 2019. 2

[3] Guillem Cucurull, Perouz Taslakian, and David Vazquez. Context-aware visual compatibility prediction. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12617–12626, 2019. 1

[4] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database.

In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 1

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1

[6] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015. 2

[7] Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop*, volume 2, 2015. 2

[8] Devi Parikh and Kristen Grauman. Relative attributes. In *2011 International Conference on Computer Vision*, pages 503–510. IEEE, 2011. 4