Why do These Match? Explaining the Behavior of Image Similarity Models Supplementary

Bryan A. Plummer[1], Mariya I. Vasileva[2], Vitali Petsiuk[1], Kate Saenko[1,3], and David Forsyth[2]

[1] Boston University, Boston MA 02215, USA
[2] University of Illinois at Urbana-Champaign, Urbana IL 61801, USA
[3] MIT-IBM Watson AI Lab, Cambridge MA 02142, USA
{bplum,vpetsiuk,saenko}@bu.edu {mvasile2,daf}@illinois.edu

1 Candidate Salience Map Generator Descriptions

In this section we provide additional details about each of the candidate saliency map generation methods used in our paper. We split these approaches into two groups: methods which analyze behavior solely through input manipulation (described in Section 1.1) and those which use an optimization procedure to learn some parameters in combination with input manipulation (described in Section 1.2). Please see Section 3.2 of our paper for a description of how these methods are adapted to our task. We also provide a runtime comparison of each approach in Table 1. A qualitative comparison between saliency map generators on the Polyvore Outfits and AwA datasets is provided in Figures 1 and 2.

1.1 Saliency Maps by Input Manipulation

A straightforward approach to producing a saliency map is to manipulate the input image by removing image regions and measuring the effect this has on the similarity score. If a large drop in similarity is measured, then the region must be important to this decision. If almost no change was measured, then the model considers the image region irrelevant. The saliency map is generated from this approach by averaging the similarity scores for each pixel location over all instances where it was removed from the input. The challenge then is to determine how to manipulate the input image to discover these important regions.

Sliding Window [7]. The first approach to removing regions of an image we shall discuss is a sliding window, where regions are sampled regularly across an image. There is a direct tradeoff, however, with how densely frames are sampled and the computational time it takes to do a forward pass through the network for each manipulated image. If frames are not densely sampled to enable an efficient solution, then it wouldn't be able to localize important regions accurately. If regions are too densely sampled then removing them might not make enough of a difference in the similarity score to take measurements accurately. When manipulating the inputs of the reference image, we apply 625 occlusion windows

each covering a square region of about 12% of image area. When manipulating both images we apply 36 occlusion windows to the reference image.

RISE [5]. This method uses Monte Carlo approach to generate saliency maps. A set of N random binary masks of size $h \times w$ is sampled where each element is independently set to 1 with probably p, and all other elements are set to 0. Typically these masks are much smaller than the input image, so they are upsampled using bilinear interpolation. This produces small continuous regions within the upsampled mask that can be used to manipulate the input image. To remove the fixed grid structure the masks are upsampled to larger than image size and then cropped randomly. For both datasets we randomly sample 2,000 random masks upsampled from 8×8 mask with the probability of preserving a region of 0.5. When manipulating the inputs of the reference image, we generate 30 random masks. Although this approach does require a significant number of random masks, we found this approach significantly outperforms using a sliding window that samples a similar number of masks on our task.

1.2 Learned Saliency Maps

We shall now discuss methods which combine input manipulation with an optimization procedure used to directly learn a saliency map. As in Section 1.1, we compare generating saliency maps for a single query image at a time using a fixed reference image as well as generating a saliency map by manipulating both the query and reference images.

LIME [6]. Rather than masking regions without any concern over the continuity of a region, this approach to generating saliency maps operates over a superpixel segmentation of an image. Images are manipulated by randomly deleting superpixels in the image. After sampling $N = 1000$ manipulated inputs, the importance of each superpixel is estimated using Lasso. Finally, important regions are selected using submodular optimization.

Mask [2]. In this approach a low resolution saliency map is directly learned using stochastic gradient decent and upsampled to the image size. Instead of manipulating an image by just deleting regions as in other methods, two additional perturbation operators are defined: adding Gaussian noise and image blurring. To help avoid artifacts when learning the mask a total-variation norm is used in addition to an $L1$ regularization to promote sparsity. This approach removes the reliance on superpixels and tends to converge in fewer iterations than LIME, although it is considerably slower in practice than other approaches (see Table 1). That said - one advantage it does have over other approaches is the ability to learn the salience map for both the query and reference image jointly (which we take advantage of when we are not using a fixed reference image). We learn a 14×14 perturbation mask for both datasets. We train the mask for 500 iterations using Adam [4] with a learning rate of 0.1.

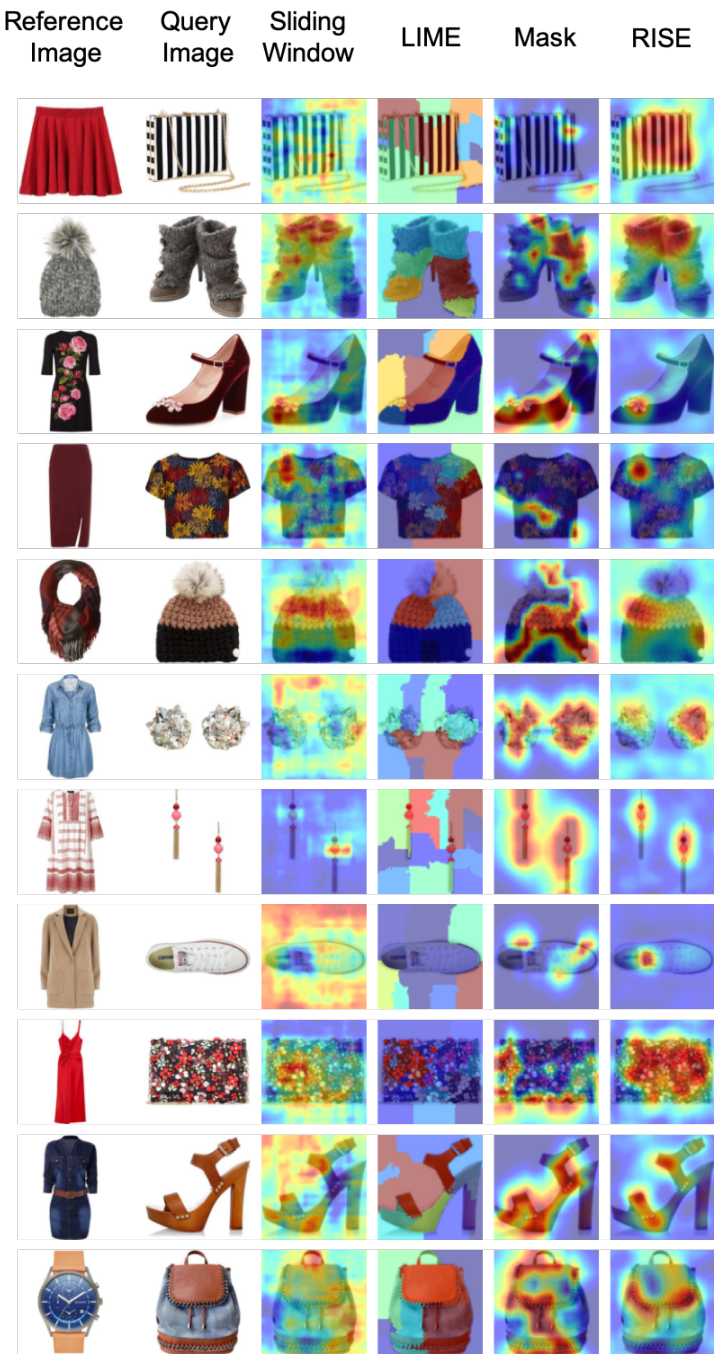

Fig. 1: Qualitative examples comparing the saliency map generator candidates on the Polyvore Outfits dataset.

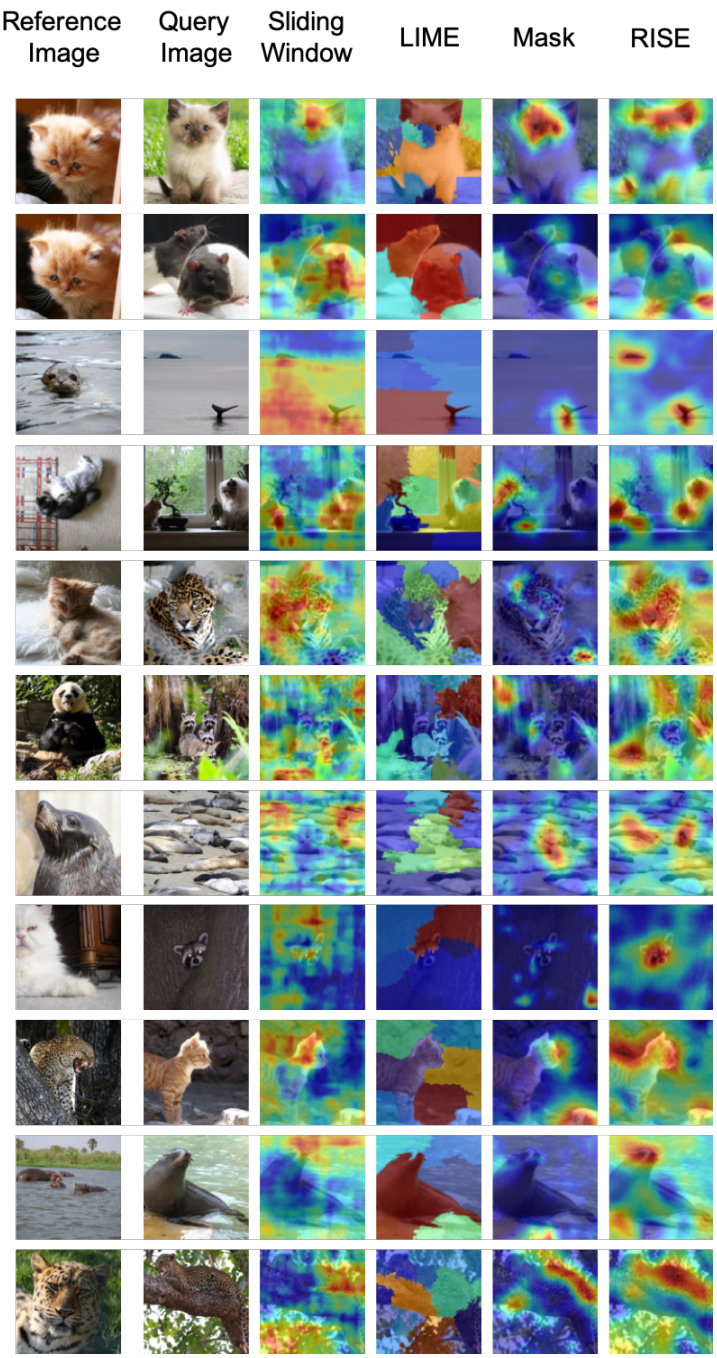

Fig. 2: Qualitative examples comparing the saliency map generator candidates on the Animals with Attributes dataset.

Table 1: Runtime comparison of the compared saliency generation methods and how using a fixed reference image, or manipulating both the query and reference images affects performance.

Method	Fixed Reference?	Time(s)
Sliding Window	Y	0.2
LIME	Y	1.2
Mask	Y	4.1
RISE	Y	0.3
Sliding Window	N	2.5
Mask	N	7.2
RISE	N	5.8

2 Additional Experimental and Implementation Details

2.1 Compared Attribute Methods

In this section we describe the attribute prediction methods we use as baselines for our attribute experiments in Section 4.2. More details on our method are provided in Section 2.2. For our Attribute Classifier, FashionSearchNet, and SANE predictor we use the same 50-layer ResNet base image encoder [3] and the last convoluntional layer of the network the same number of channels as the number of classes. The output of the last convolutional layer we refer to as our Attribute Activation Maps in our paper, and we use a global average pooling (GAP) layer to obtain attribute scores from these Attribute Activation Maps.

- **Random baseline.** For insertion we randomly select an attribute not in the image to insert into the image. Analogously, for deletion we randomly select an attribute that is in the image to be removed.
- **Attribute Classifier.** This model uses a our image encoder as described earlier as a simple baseline. Although classic methods typically use a fully connected layer rather than a convolutional layer followed by a GAP layer, we found the latter (*i.e.*, our approach) to improve performance 2-3 mAP in our experiments. This model has the same architecture as our SANE attribute predictor, but unlike the SANE model, it doesn't get any kind of supervision of its Attribute Activation Map.
- **FashionSearchNet [1].** This network uses an Attribute Activation Map to identify and extract a region of interest for each attribute. These extracted regions are fed into two branches consisting of three fully connected layers which is trained for both attribute classification and image retrieval. We remove the image retrieval components in our experiments. This model provides a weakly-supervised baseline of an Attribute Activation Map to compare to SANE's saliency-supervised activation map.

2.2 SANE Details

Due to its efficient (see Table 1) and overall good performance (see Table 1) we selected the fixed-reference RISE as our saliency map generator. For each

training image, we sample up to five similar images using the ground truth annotations of each dataset and generate saliency maps using each sampled image as the reference image. We train our attribute model for 300 epochs using Adam [4] with a learning rate of $5e^{-4}$ and set $\lambda = 5e^{-3}$ in Eq. 3 from the paper. After each epoch, we computed mAP on the validation set and kept the best performing model according to this metric. Additional qualitative examples for explanations produced by our SANE model on the Polyvore Outfits and AwA datasets are provided in Figures 3 and 4.

We provide an example of the attribute deletion process in Figure 5. After identifying an attribute to remove in an image, we search for the K most similar image to the input from a database that doesn't contain the input attribute. Image similarity is computed over the attribute space, *i.e.*, we want to keep the predictions of each attribute the same, and only vary the target attribute. For Polyvore Outfits, we only consider images of the same type (*i.e.*, so tops can only be replaced with other tops). To ensure we don't bias towards a single attribute model, average the predictions made by each attribute model in our experiments (Attribute Classifier, FashionSearchNet, and SANE).

We see on the left side of Figure 5 that some attributes like colors are largely retained when the attribute has to do with a non-color based attribute. On the returned AwA images on the right side of Figure 5 we see how some attributes can lead to significant changes in the images or almost none at all depending on the attribute selected to remove. For some items we can see that multiple attributes may have changed in the retrieved images in these figures. To try to account for this, we compute our attribute insertion/deletion metrics over the $K = 5$ most similar images to the input returned by our image selection procedure and then average the change in similarity using these 5 images. While this does provide a noisy estimate of the change in similarity, our human evaluation in Section 4.4 shows that there is a correlation between gains in our automatic metrics and improvements in a human user's understanding of predictions made by an image similarity model.

In Section 3.3 we discuss how we estimate how likely each attribute is a "good" explanation in held-out data. This is used as a prior to bias our attribute selections towards attributes that are known to be good attribute explanations. In Figure 6 we show the ground truth bias for the attribute detection task according to our metrics for the AwA dataset. Note, however, that this prior would change for a different image similarity model. For example, if the image similarity model was more biased towards colors, then we would expect to see the likelihood for "black," "brown," and "gray" to increase.

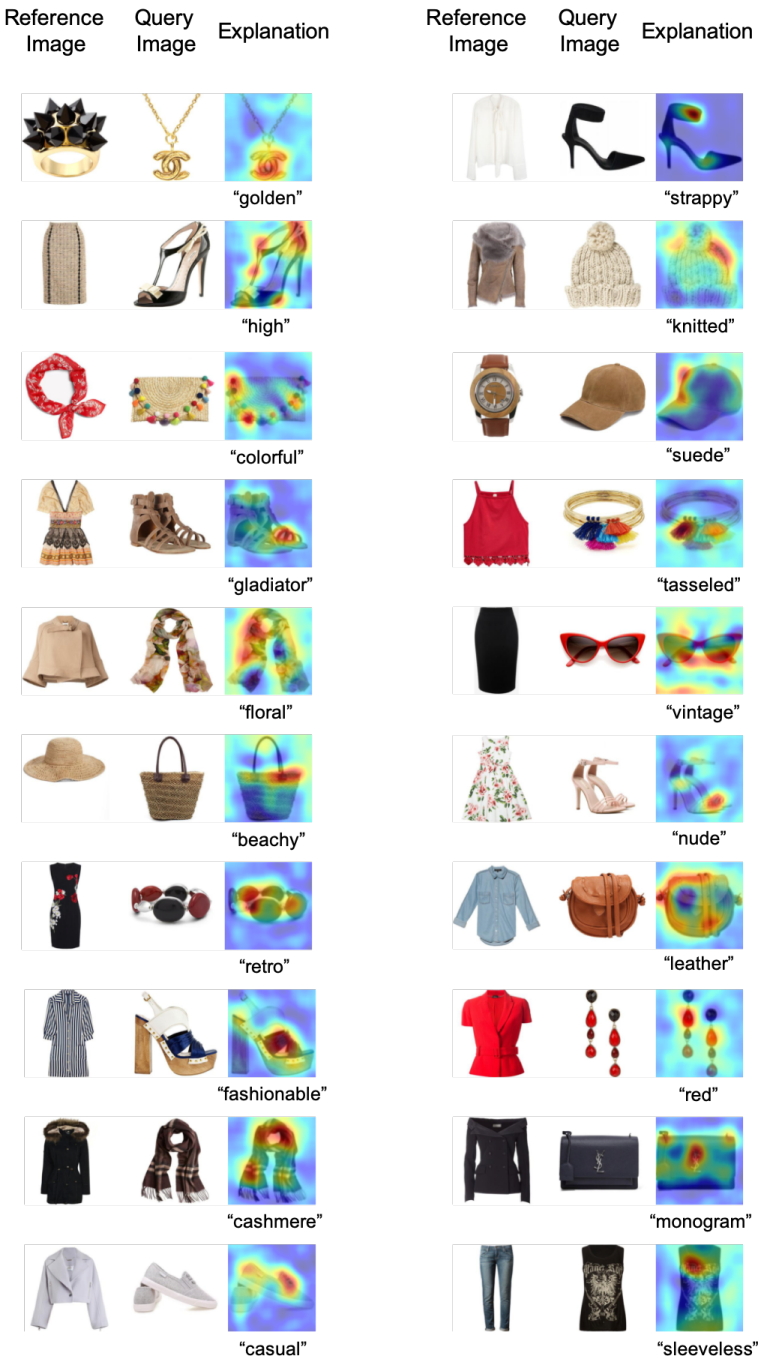

Fig. 3: Additional qualitative examples of our SANE explanations on the Polyvore Outfits dataset.

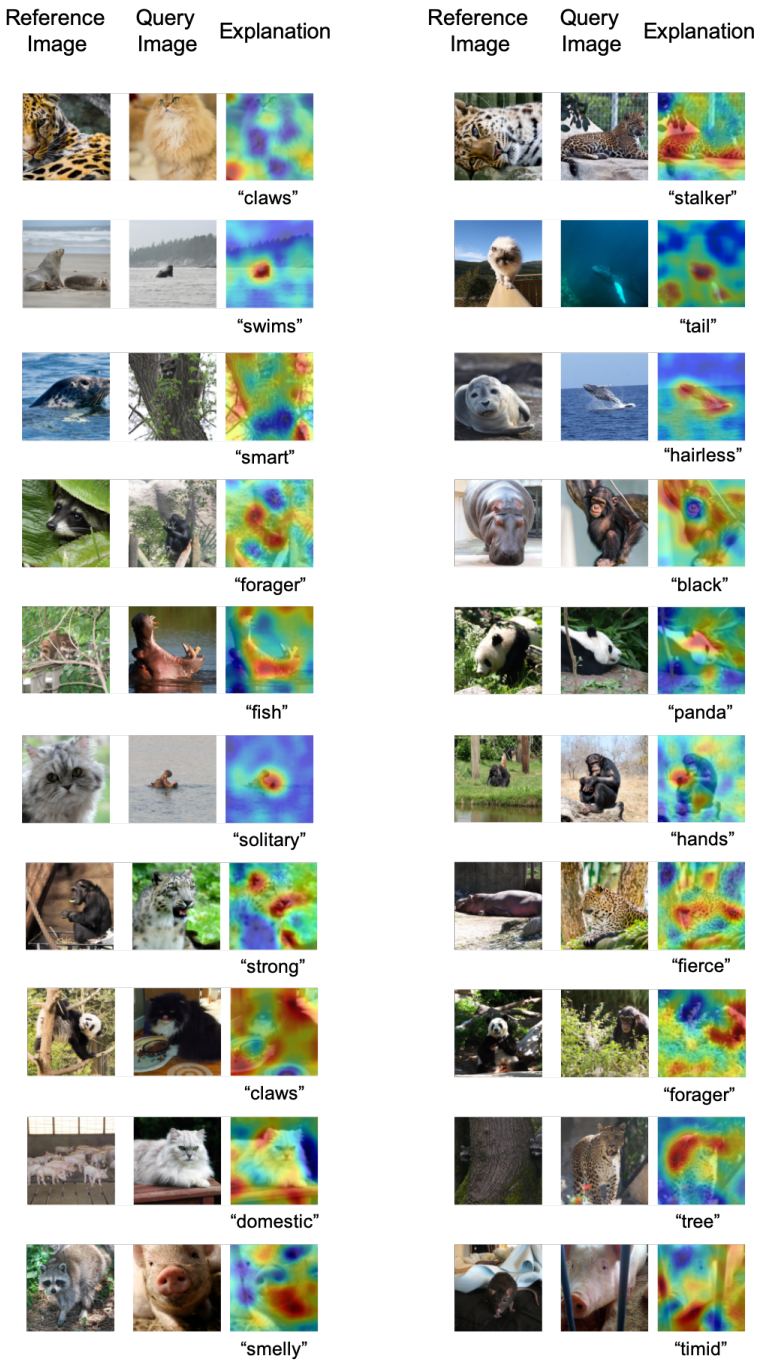

Fig. 4: Additional qualitative examples of our SANE explanations on the AwA dataset.

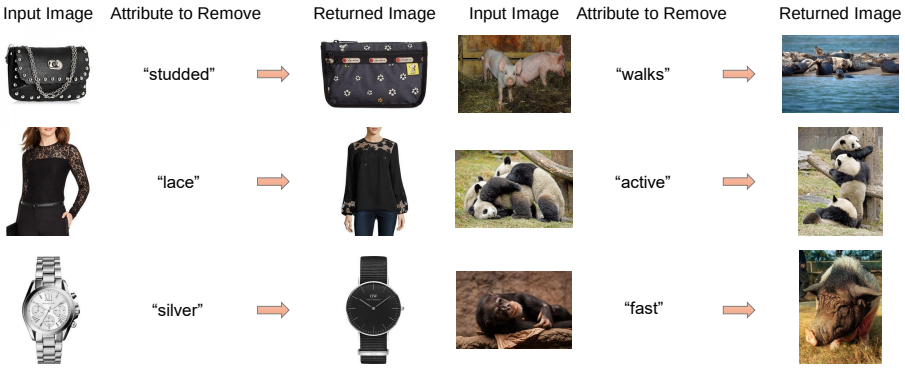

Fig. 5: Examples of the attribute deletion process used to evaluate how good an attribute is as an explanation. We measure the similarity of the input image and some reference image as well as between the returned image and the reference image. If a large change in similarity is measured then the attribute is considered a “good” explanation. If similarity stays about the same, the attribute is considered a “poor” explanation, *e.g.*, trying to remove “active” from the pandas on the right.

3 User Study

An example of our user study variants for a given image triplet is shown in Figures 7 and 8. The question asks users to select, given the image triplet (A, B, C) presented, along with any additional information in each case, whether the image similarity model predicted B or C as a better match for A, as outlined in Section 4.3.

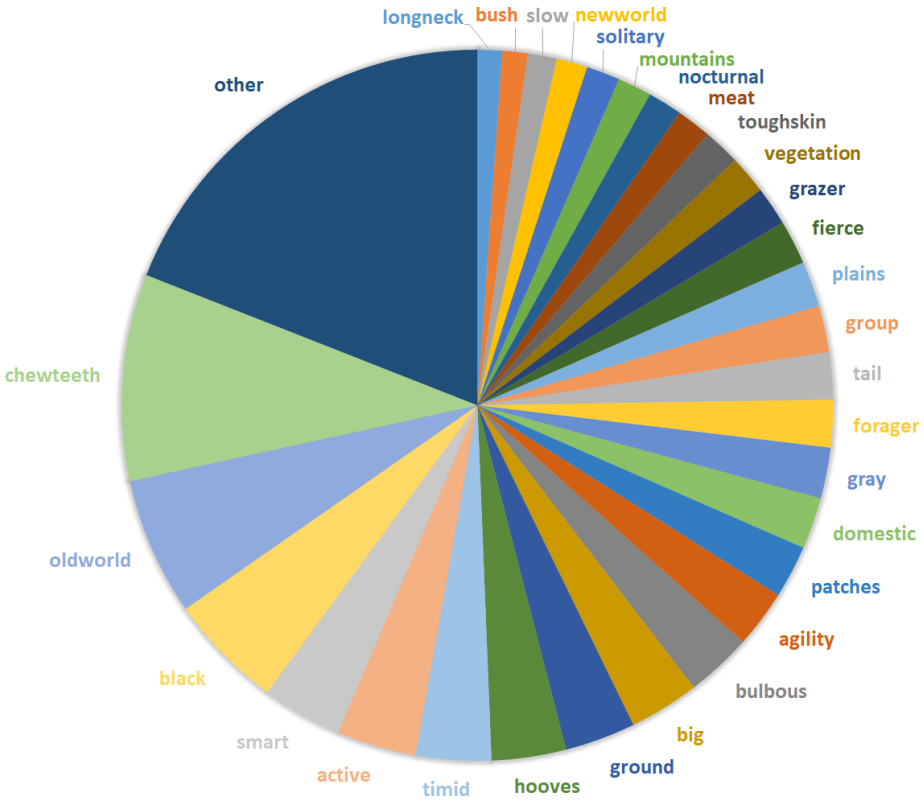

Fig. 6: The likelihood each attribute in the AwA dataset was identified as the best attribute for an image pair on held-out data.

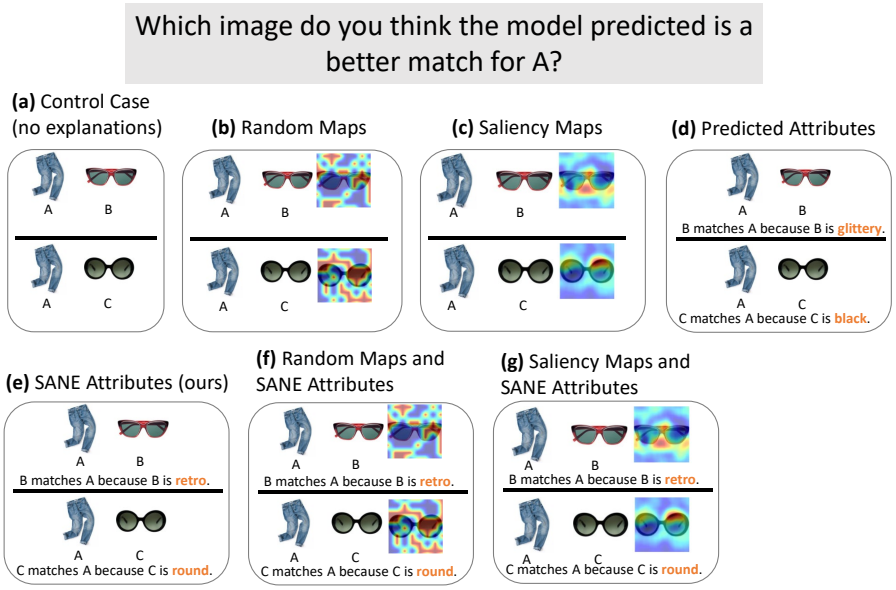

Fig. 7: Variants of our user study for a given image triplet on the Polyvore Outfits dataset.

4 Discovering Useful Attributes

For datasets without attribute annotations, or those where the annotated attributes doesn't cover the extent of the visual attributes present in the dataset (*i.e.* there are many unannotated attributes) we propose a method of discovering attributes that are useful for providing model explanations. An attribute that is useful for explanations would commonly appear in the high importance regions of saliency maps. When generating saliency maps for a query image, if many reference images attend to the same region of the query image then it is likely they are all matching to it for similar reasons (*i.e.* there may be some attribute that they share which matches the query). Given this observation, we discover attributes using the following saliency-based procedure:

1. Obtain K similar images for query image q using k-NN.
2. Generate a saliency map over q for each of the similar (reference) images.
3. Keep only those reference images which have their saliency peaks in the most common location (such as a unit square in a 7×7 grid) and pick top N of them that have the highest similarity.
4. For each reference image, generate its saliency map with q and crop a 30×30 patch around the peak saliency region in the reference image.
5. Upsample all the generated patches to full image resolution and get their embeddings.

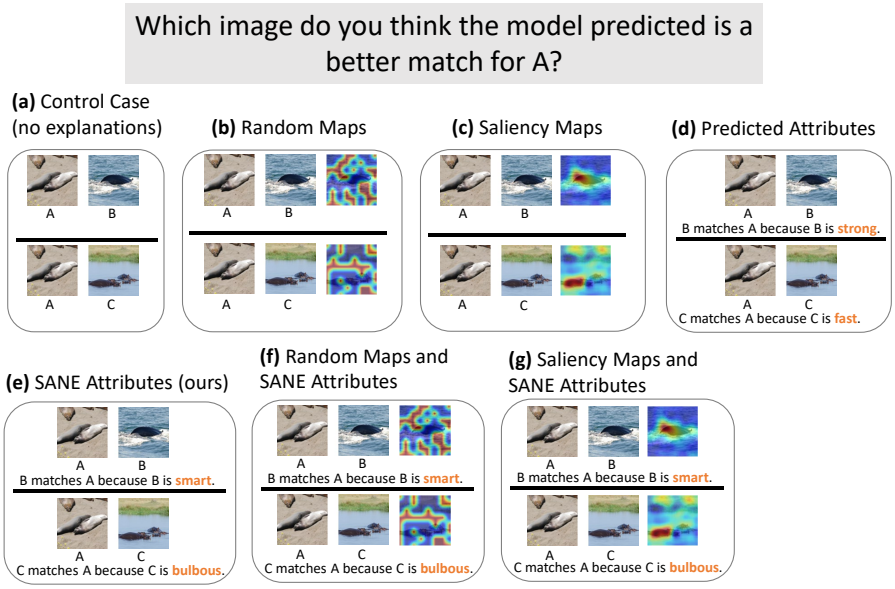

Fig. 8: Variants of our user study for a given image triplet on the AwA dataset.

6. Cluster the patches produced for multiple queries q. Each cluster represents an attribute. If multiple patches were extracted from an image and they got assigned to different clusters, this image would be labeled with multiple attributes.

Figure 9a illustrates the clustering produced by this procedure for a set of queries from Polyvore Outfits dataset.

To evaluate this approach we compare it to randomly assigning images to clusters and to clustering based on their own embeddings, disregarding the saliency of image regions (Figure 9b). Saliency-based attribute discovery works best among the three unsupervised methods for Polyvore Outfits data, but full-frame clustering outperforms it for the AwA dataset (Table 2). We suspect the full frame clustering works better for AwA since it considers the background more than the patch-based method (Polyvore Outfits image's typically have white backgrounds). In addition, our discovered attributes would likely be noisier due to the similarity model focusing on the background patches in some images as well. Although our initial results are promising, attempting to discover attributes useful for explanations warrants additional investigation.

References

1. Ak, K.E., Kassim, A.A., Lim, J.H., Tham, J.Y.: Learning attribute representations with localization for flexible fashion search. In: The IEEE Conference on Computer

Table 2: Discovered attribute explanation performance comparison using the full SANE model.

	Polyvore Outfits		Animals with Attributes 2	
Attribute Types	Insertion (↑)	Deletion (↓)	Insertion (↑)	Deletion (↓)
Random	25.3	-6.3	2.1	-8.5
Full Frame Discovery	29.4	-9.7	4.8	-22.6
Patch Discovery	30.2	-10.3	5.4	-22.9
Supervised Attributes	**31.5**	**-11.8**	**6.2**	**-24.1**

Vision and Pattern Recognition (CVPR) (2018)
2. Fong, R.C., Vedaldi, A.: Interpretable explanations of black boxes by meaningful perturbation. In: The IEEE International Conference on Computer Vision (ICCV) (2017)
3. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016)
4. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: The International Conference on Learning Representations (2015)
5. Petsiuk, V., Das, A., Saenko, K.: Rise: Randomized input sampling for explanation of black-box models. In: British Machine Vision Conference (BMVC) (2018)
6. Ribeiro, M.T., Singh, S., Guestrin, C.: "why should i trust you?": Explaining the predictions of any classifier. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2016)
7. Zeiler, M.D., Fergus, R.: Visualizing and understanding convolutional networks. In: The European Conference on Computer Vision (ECCV) (2014)

(a) Patch-based clustering

(b) Full-frame clustering

Fig. 9: Six clusters defining the attributes for two approaches to attribute discovery. (a) Each image is assigned a list of clusters that have patches from this image. Clustering is performed on salient patches. (b) Each image is assigned one of the clusters as an attribute. Clustering is performed on full-frame images.