Visualizing Deep Similarity Networks

Anonymous WACV submission

Paper ID 96

Abstract

For convolutional neural network models that optimize an image embedding, we propose a method to highlight the regions of images that contribute most to pairwise similarity. This work is a corollary to the visualization tools developed for classification networks, but applicable to the problem domains better suited to similarity learning. The visualization shows how similarity networks that are fine-tuned learn to focus on different features. We also generalize our approach to embedding networks that use different pooling strategies and provide a simple mechanism to support image similarity searches on objects or sub-regions in the query image.

1. Introduction

While convolutional neural networks have become a transformative tool for many image analysis tasks, it is still common in the literature to describe these deep learning approaches as ”black boxes”. To address these concerns, there have been substantial efforts to understand and visualize the features of classification networks [2, 3, 6, 11, 14, 18, 19, 24, 26, 27]. However, much less work has focused on visualizing and understanding similarity networks, which learn an embedding that maps similar examples to nearby vectors in feature space and dissimilar examples to be far apart [15, 23].

Our approach highlights the image regions that contributed the most to the overall similarity between two images. Figure 1 shows example visualizations for the top image retrieval results from three different application domains (Google Landmarks [25], VGG-Faces [10], and Traffickcam Hotel Rooms [16]). Each row of the figure shows a query image and the three most similar database images returned from a network trained for the respective task. The heatmap overlay shows the relative spatial contribution of each image to the similarity score with the query.

Our approach aligns with the recent trend toward explainability for learning-based tasks and extends recent work in visualizing classification networks to the case of similarity networks. Our specific contributions include:

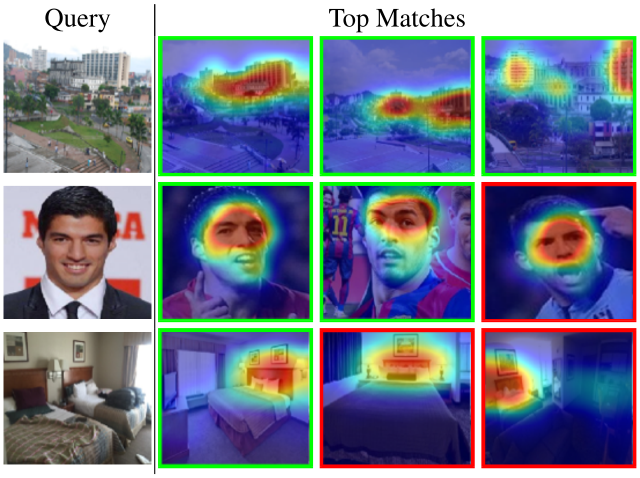


Figure 1: Our approach to visualizing the embeddings generated by deep similarity networks calculates the contribution of each pixel location to the overall similarity between two images. We evaluate our approach on a variety of problem domains and network architectures.

- a novel visualization approach for similarity networks;
- an analysis of the effect of training and “late-stage” pooling strategies for similarity networks; and,
- an approach to using similarity visualizations to support object- and region-based image retrieval.

2. Background

Visualizations provide a way to better understand the learning process underlying deep neural networks. Much of the work in this area focuses on visualizations for classification networks and not similarity networks. While networks used for each type of problem share many similarities, the differences in the output (i.e., sparse vs. dense feature vectors) is significant, requiring new methods for visualizing similarity networks.

CNN Visualization Previous work on CNN visualizations can be broadly categorized by the depth of the portion of the networked being visualized. Some methods provide visualizations that highlight the inner layer activations [3, 6, 27]. A majority of the work targets the output layer to produce visualizations which seek to explain why classification networks output a particular label for an image. These include approaches that mask off parts of the input images and provide a visual quantification of the impact on the output classification [24]. Another approach generates saliency maps, which represent which pixels in an image contributed to a particular output node [14]. There has been work that generates class activation maps, which map an output back to the last convolutional layer in the network by weighting the filters in that layer by the weights between the final pooling layer and the output feature [26]. Inception [18], which hallucinates images that activate a particular class from random noise, can also serve as visualization tool to provide insight into the learning process.

Similarity Learning Much of the work in similarity learning with deep neural networks focuses on learning better similarity functions using, for example, pairwise losses [17, 22, 23], triplet losses [8, 13, 15, 20], and direct embedding [9]. Compared to the efforts toward understanding classification networks, there has been much less work in visualizing and analyzing similarity networks. One method visualizes the similarity of single filters from the different convolutional layers of an embedding network [1]. Another method computes image similarity as the inner product between the normalized elements of a final max pooling layer and produces a visualization with bounding boxes around highly active regions for the ten features that contribute most to the similarity of a pair of images [11, 19].

Visualizing a few features is effective for networks that tend to be sparse, but in Section 4.1 we show that in embedding networks the similarity tends to be explained by a large number of features. This motivates our approach to visualize how all features affect the similarity score.

3. Visualization Approach

Networks used in similarity learning broadly consist of: (1) a convolutional portion, (2) a ”flattening” operation (usually max or global average pooling), and (3) a fully-connected portion. A recent study covering a number of image retrieval tasks, however, suggests that the best generalization performance is obtained using the output from the layer immediately after the pooling operation [21]. Our approach is applicable to networks of this structure, including popular models such as the Resnet [7] and VGG [10] network architectures.

Given an input image, I, and a trained similarity network, our approach relies on the activations of the layers before and after the pooling operation. Let $\boldsymbol{\alpha}$ represent the $K \times K \times C$ tensor of the last convolutional layer, where K represents the length and width (usually equal) and C represents the number of filters. Let $\boldsymbol{\beta}$ represent the C-dimensional vector after the pooling operation for an image, as shown in Figure 2. In similarity learning, the dot product of these normalized feature vectors is a widely-used similarity function [4, 9, 13, 17, 22, 23], so the similarity of two images $I^{(i)}, I^{(j)}$ can be written as:

$$s(\boldsymbol{\beta}^{(i)}, \boldsymbol{\beta}^{(j)}) = \frac{\boldsymbol{\beta}^{(i)} \cdot \boldsymbol{\beta}^{(j)}}{\left\|\boldsymbol{\beta}^{(i)}\right\| \left\|\boldsymbol{\beta}^{(j)}\right\|} \tag{1}$$

Our visualization approach results in spatial similarity maps, where the overall similarity between two image feature vectors is spatially decomposed to highlight the contribution of image regions to the overall pairwise similarity, as shown in Figure 2. Computing the similarity maps depends on the flattening operation between the convolutional portion of the network and the output feature. Max pooling and global average pooling are the most commonly applied operations at this stage in modern networks. We show how our similarity maps are computed for each case.

3.1. Average Pooling

For networks which employ average pooling as the flattening operation, the output feature, $\boldsymbol{\beta}$, is:

$$\boldsymbol{\beta} = \frac{1}{K^2} \sum_{x,y} \boldsymbol{\alpha}_{(x,y)} \tag{2}$$

where $\boldsymbol{\alpha}_{(x,y)}$ represents the C-dimensional slice of $\boldsymbol{\alpha}$ at spatial location (x, y). The similarity of images $I^{(i)}$ and $I^{(j)}$ can be directly decomposed spatially, by substituting $\boldsymbol{\beta}^{(i)}$ in Equation 1 with Equation 2:

$$\begin{aligned} s(\boldsymbol{\beta}^{(i)}, \boldsymbol{\beta}^{(j)}) &= \frac{\boldsymbol{\beta}^{(i)} \cdot \boldsymbol{\beta}^{(j)}}{\left\|\boldsymbol{\beta}^{(i)}\right\| \left\|\boldsymbol{\beta}^{(j)}\right\|} \\ &= \frac{\frac{1}{K^2}\left(\boldsymbol{\alpha}^{(i)}_{(1,1)} + \ldots + \boldsymbol{\alpha}^{(i)}_{(K,K)}\right) \cdot \boldsymbol{\beta}^{(\boldsymbol{j})}}{\left\|\boldsymbol{\beta}^{(i)}\right\| \left\|\boldsymbol{\beta}^{(j)}\right\|} \\ &= \frac{\boldsymbol{\alpha}^{(i)}_{(1,1)} \cdot \boldsymbol{\beta}^{(j)} + \ldots + \boldsymbol{\alpha}^{(i)}_{(K,K)} \cdot \boldsymbol{\beta}^{(j)}}{Z} \end{aligned} \tag{3}$$

where Z is the normalizing factor $K^2 \left\|\boldsymbol{\beta}^{(i)}\right\| \left\|\boldsymbol{\beta}^{(j)}\right\|$.

These terms can be rearranged spatially and visualized as a heat-map to show the relative contribution of each part of the image to the overall similarity. Symmetrically, the similarity can be decomposed to highlight the contribution of the other image in the pair to the overall similarity, as shown on the right side of Figure 2.

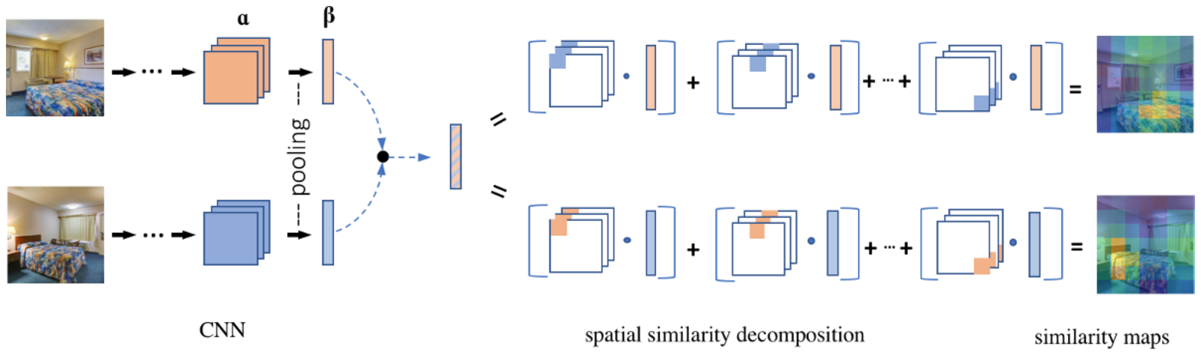


Figure 2: Our approach considers similarity networks with a final convolutional layer, $\boldsymbol{\alpha}$, followed by a pooling operation which produces output features, $\boldsymbol{\beta}$. Similarity between two images is measured as the dot product of these output features after normalization. Factoring this value produces visualizations that highlight how much each region of the image contributes to the similarity.

3.2. Max Pooling

With a modification, the approach can also accommodate networks that use max pooling as the flattening operation. In max pooling, each element of an output vector $\boldsymbol{\beta}$ is equal to the max value of the activation of its corresponding filter in the last convolutional layer:

$$\boldsymbol{\beta} = \max_{x,y} \boldsymbol{\alpha}_{(x,y)} \tag{4}$$

Unlike average pooling, where each of the composite components contribute equally to the output feature, decomposing max pooled features requires an additional step. For a max pooled feature, $\boldsymbol{\beta}$, we construct a surrogate tensor, $\hat{\boldsymbol{\alpha}}$, for the convolutional portion as follows:

$$\hat{\alpha}_{(x,y,c)} = \begin{cases} 0 & \text{if } \alpha_{(x,y,c)} \neq \beta_{(c)} \\ \frac{\alpha_{(x,y,c)}}{N_{(c)}} & \text{if } \alpha_{(x,y,c)} = \beta_{(c)} \end{cases} \tag{5}$$

where $N_{(c)}$ represents the number of spatial locations equal to the maximum value for filter c. That is, for each filter, we assign the maximum value to the location that generated it (divided evenly in cases of ties), and zero otherwise. This gives the following formulation for the spatial similarity decomposition in the case of max pooling:

$$s(\boldsymbol{\beta}^{(i)}, \boldsymbol{\beta}^{(j)}) = \frac{\hat{\boldsymbol{\alpha}}^{(i)}_{(1,1)} \cdot \hat{\boldsymbol{\beta}}^{(j)} + \ldots + \hat{\boldsymbol{\alpha}}^{(i)}_{(K,K)} \cdot \hat{\boldsymbol{\beta}}^{(j)}}{\left\|\boldsymbol{\beta}^{(i)}\right\| \left\|\boldsymbol{\beta}^{(j)}\right\|} \tag{6}$$

Similar to the case for average pooling, similarity maps can be computed in either direction for a pair of images.

We scale the heatmaps using bilinear interpolation and blend them with the original image to show which parts of the images contribute to the similarity scores.

4. Results

Similarity networks trained for three different problem domains are used to test the approach. Except where noted, we use the following network architectures and output features. For the Google Landmarks [25] and TraffickCam Hotel Rooms [16] datasets, we fine-tune a Resnet-50 [7] network from pre-trained ILSVRC weights [12] using the combinatorial variant of triplet loss described in [8]. For the VGG-Faces dataset, we use the VGG-Faces network trained on the VGG-Faces2 dataset [10, 5]. For each of the networks, we use the layer immediately after the pooling operation as our output features (2048-D for Resnet-50, and 512-D for VGG-Faces).

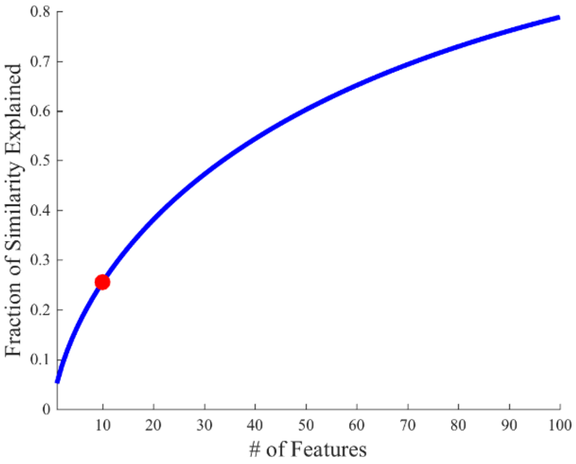


Figure 3: The plot shows the average contribution of the top–K components of the feature vectors to the similarity score between pairs of images from the same class using the pre-trained VGG-Faces dataset. The top 10 features (the number of features visualized in prior work, and identified in this plot by a red dot), account for less than 30% of the similarity score between two images, motivating our attempt to visualize all features.

4.1. Feature Importance

Prior work in understanding similarity networks focuses on either a few filters or few regions that contribute most to the similarity between a pair of images [1, 11, 19]. Our visualization approach, by comparison, summarizes the contribution of every feature to the similarity between a pair of images.

In the following experiment, we demonstrate that for similarity networks, the top few most important components only represent a small fraction of the overall image similarity. Figure 3 shows the average contribution of the first k components for 1000 randomly sampled pairs of images from the same class using the pre-trained VGG-Faces network. The top 10 features (the number of features visualized in prior work, and identified in this plot by a red dot) contribute less than 30% of the overall similarity score. This suggests that, unlike classification networks which output sparse feature vectors, understanding the output of similarity networks requires a visualization approach that explains more than only a few features at once. Our approach to visualizing similarity networks incorporates all of the feature vector components and calculates the contribution of each pixel location to the overall similarity between two images.

4.2. Visualizing Pairwise Similarity

Figure 4 shows pairs of images that produced high similarity scores. In the top pair of images from the Google Landmarks dataset, the viewpoints are quite different, but the visualization approach highlights the specific building that the network identified as being similar. This building is in the foreground of one of the images, but hidden in the background of the other. The middle pair of images are of the same gentleman in the VGG-Faces dataset. the visualization highlights his lower facial features. The final pair of images is from different hotels in the TraffickCam Hotel Rooms dataset. the visualization highlights that both rooms have similar light fixtures mounted to the headboard. These examples demonstrate the ability of the visualization approach in explaining why a network produces similar embeddings for a pair of images, even in cases where that may not be readily apparent to a human observer looking at the images.

4.3. Similarity Learning During Training

Figure 5 shows the visualization for a query image and its top 3 most similar images during the training process. For the Google Landmarks dataset, we see that even by 5,000 iterations, the network has largely learned that it is the skyline that makes this scene recognizable. In subsequent iterations, the network refines the similarity metric and focuses on more specific regions, such as the buildings in the scene. On the TraffickCam Hotel Rooms dataset, on

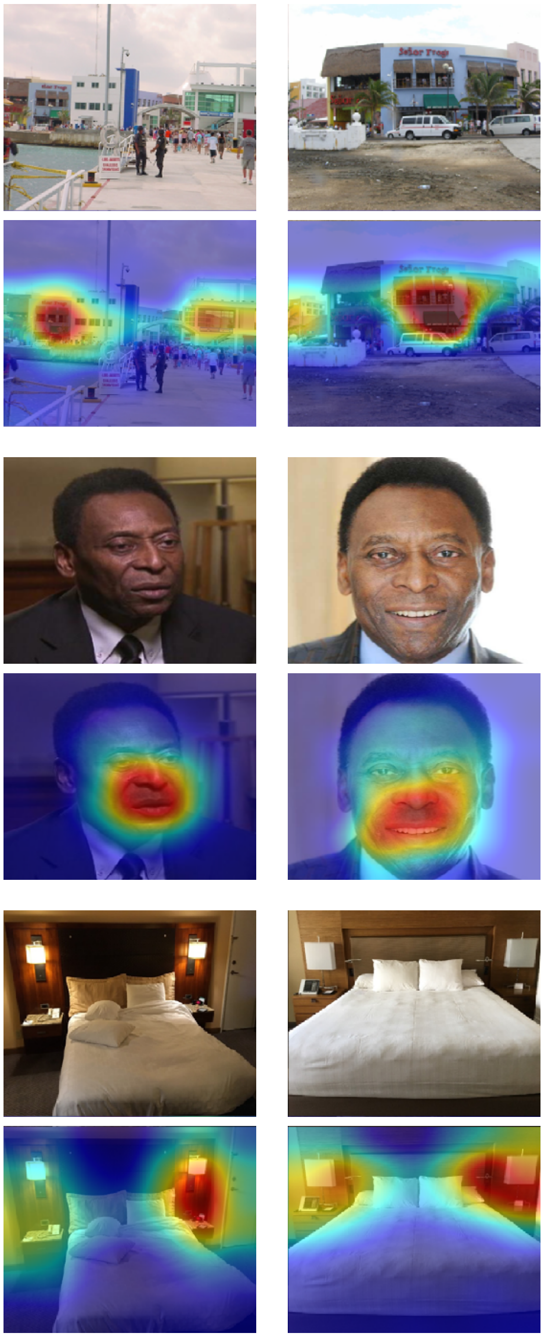


Figure 4: Visualizations to understand image similarity. (Top) For two images of the same landmark, the visualization highlights the building in the background in left image, but the foreground in the right. (Middle) For two images of the same person, the nose and mouth region are highlighted. (Bottom) For two images of rooms from different hotels, the visualization highlights the similar light fixtures mounted to the headboard.

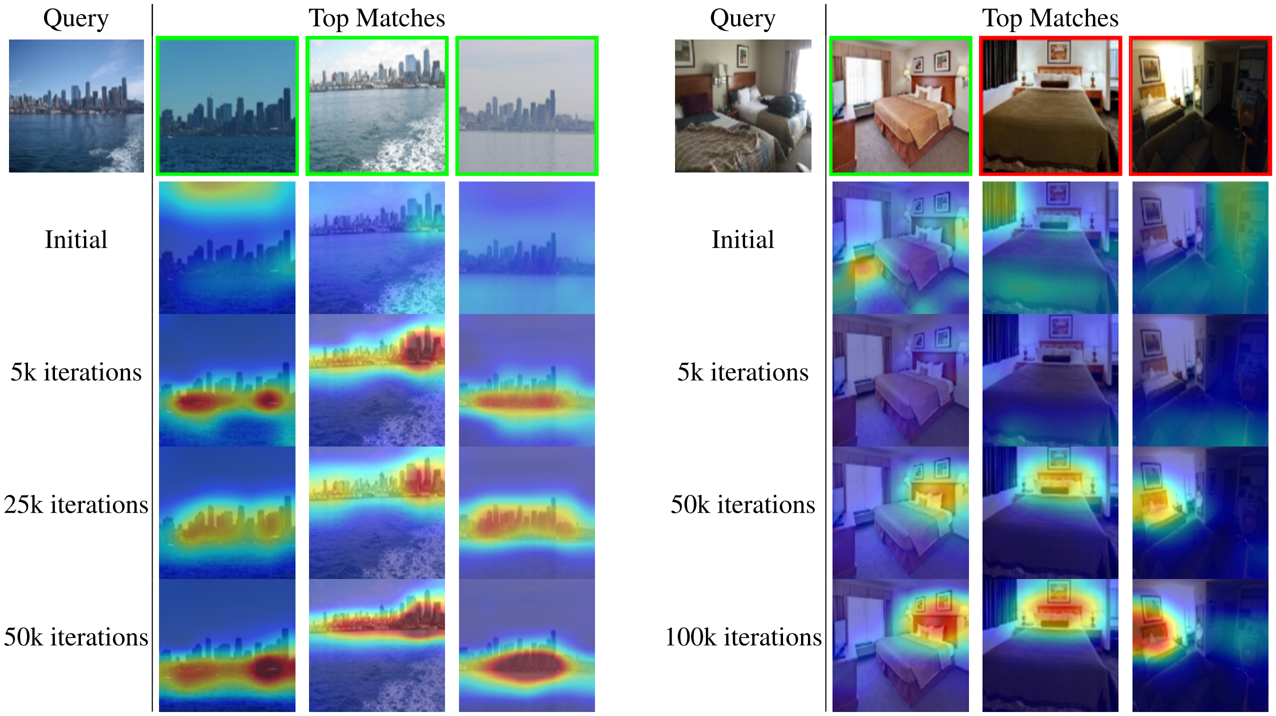


Figure 5: Each subfigure shows visualizations from networks pre-trained on ImageNet and fine-tuned on Google Landmarks (left) and TraffickCam Hotel Rooms (right) during the training process.

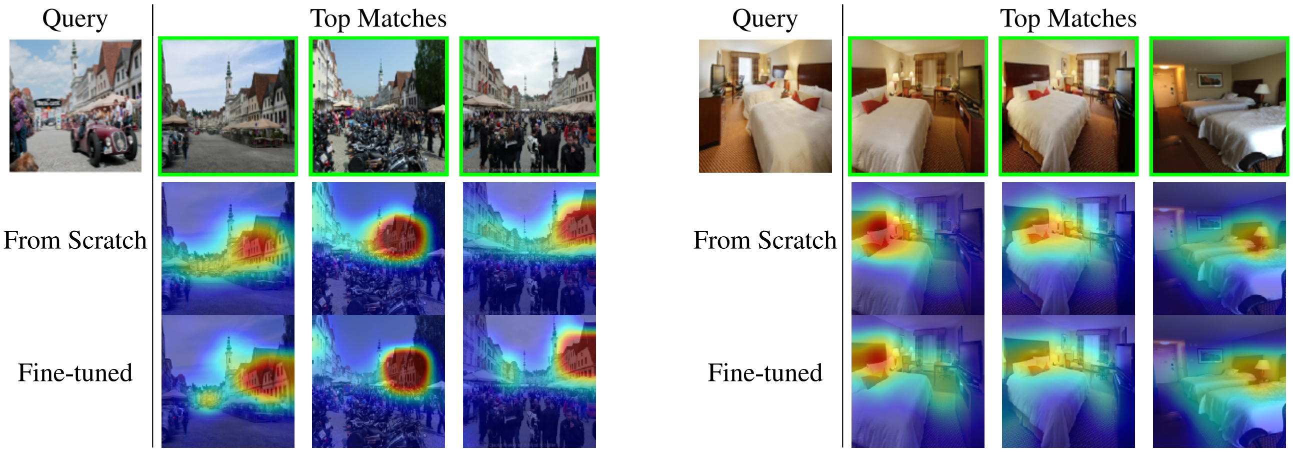
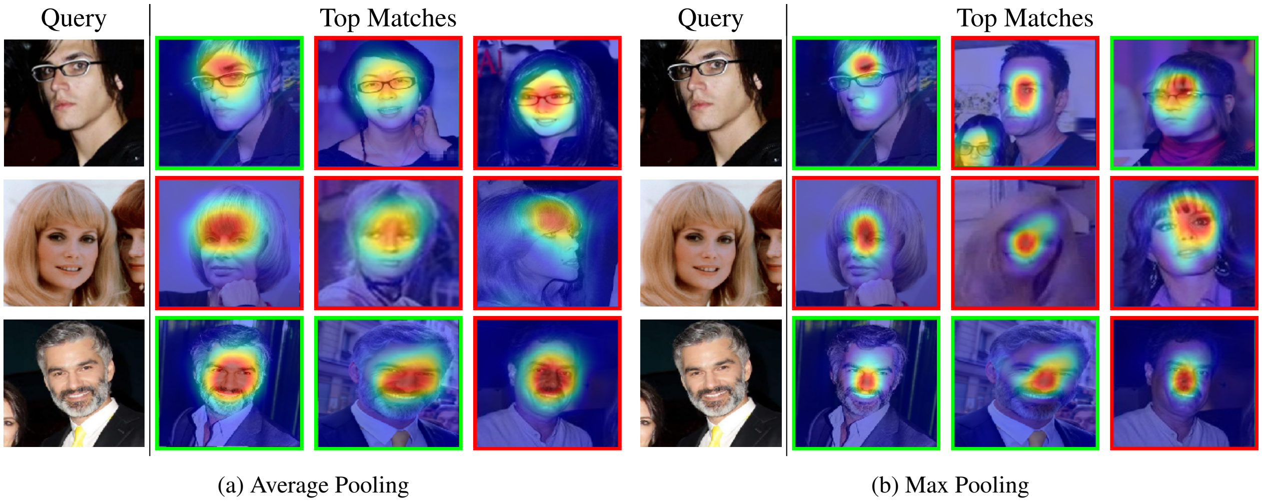


Figure 6: Fine-tuning vs. Training from Scratch. The visualization highlights that, regardless of the initialization, the networks converge to similar representations.

the other hand, the network takes longer to learn a similarity embedding. At 5,000 iterations, the network has not yet focused on specific elements of the hotel rooms. By 50,000 iterations, it becomes clear that the headboard is the relevant part of this particular set of images, and by 100,000 iterations, the network appears to be refining that focus. These examples demonstrate the utility of the visualization in understanding ***when*** a network has learned a useful similarity metric, in addition to understanding what components of a scene the network has learned to focus on.

Another consideration when training similarity networks is whether to train from scratch or fine-tune from pre-trained weights. Figure 6 shows the top three results for a query image, the similarity visualizations when trained from scratch, and when fine-tuned from pre-trained weights. In the examples from the Google Landmarks and TraffickCam Hotel Rooms dataset, we see that both the fine-tuned network and the network trained from scratch converged to similar



(a) Average Pooling

(b) Max Pooling

Figure 7: Average vs. Max Pooling. For the same VGG-Faces network architecture, these visualizations show the pairwise similarity for models trained with average pooling and max pooling.

encodings of similarity (e.g., both the fine-tuned network and network trained from scratch highlight the building facade in the Google Landmarks scene and the headboard in the TraffickCam hotel). These results suggest that both approaches converge to features that encode the same important elements of the scenes and that it is reasonable to fine-tune from pre-trained weights (even from a fairly dissimilar task, such as a classification task trained on ILSVRC).

4.4. Average vs. Max Pooling

As described in Section 3, the visualization approach is applicable to networks with either average and max pooling at the end of the convolutional portion of the network. Figure 7 shows the comparison between two VGG-Faces networks, one trained with average pooling and one with max pooling. For the same image pairs, the embeddings highlight different regions. For example, the average pooling network focuses on glasses in the first query image, while the max pooling network focuses more on eyebrow shape. Additionally, the regions of similarity are larger in the average pooling network compared to the max pooling network. This is reasonable as all of the regions contribute to output embedding in average pooling, but not max pooling.

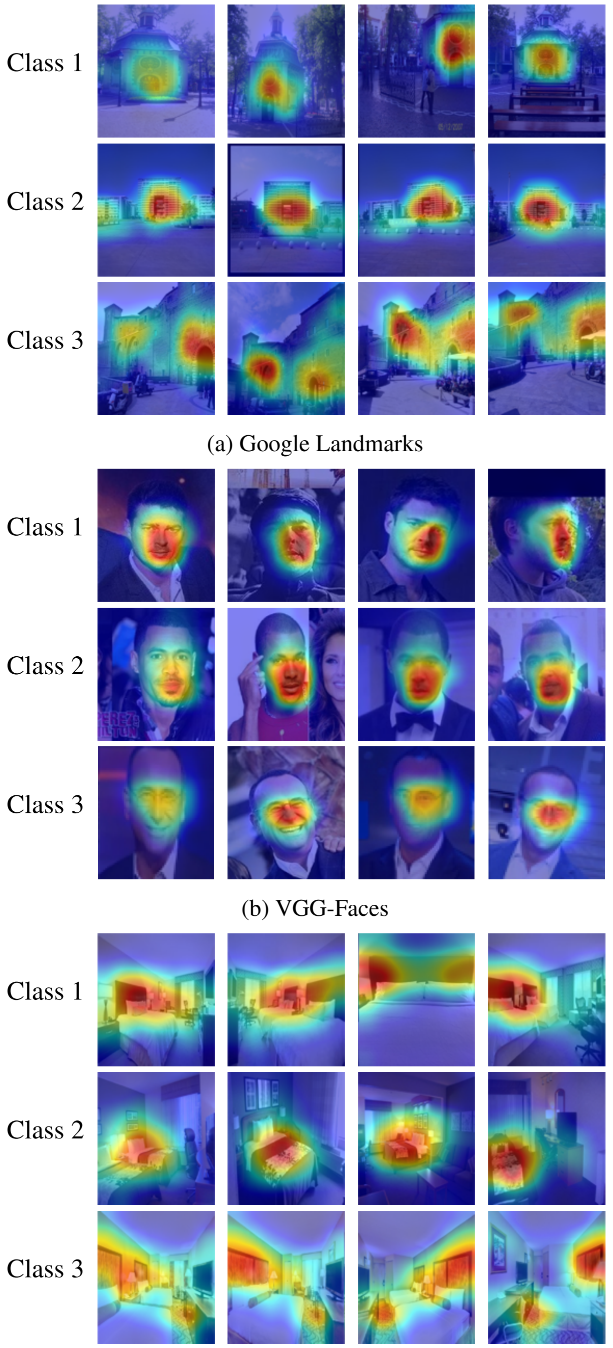
4.5. Class Similarity

Using our method, we can discover the most representative components of a class of images. This is a natural extension of class activation maps for classification networks, which visualize the components of an image contribute the most to a particular output label. We generate class activation maps for a given image by summing the pairwise similarity maps with the other images in the same class. Figure 8 shows class similarity visualization for a selection of images from each of our datasets. The visualizations highlight the portions of the image that most contribute to the similarity of the output feature to those of the images in the same class. For example, in Class 1 of the Google Landmarks dataset, the clock on the building's facade is the most important part in each example image; in Class 2 of the VGG-Faces dataset, the nose and lips are most important; and in Class 3 of the TraffickCam Hotel Rooms dataset, the headboard is most associated with the hotel identity.

4.6. Object- and Region-Specific Retrieval

The previous experiments highlight the utility of the visualization method on image retrieval tasks, which consider the entire image as input. The same visualization approach can also support object- or region-specific image retrieval.

We first compute the similarity maps between a query and the database images. Recall that the similarity map for a pair of images sums to the overall similarity score. Therefore, the sum of the values contained within subregions of the similarity map reflect the how much the subregion contributes to the match with the corresponding images. This subselection allows for database images to be sorted based on the contribution to the query image's similarity score at the region of interest. Figure 9 shows the most similar image from the three most similar classes for both whole image similarity and also using the highlighted sub-regions of the image. This modification allows for object- or region-specific image retrieval. For example, in the Google Landmarks dataset, searches can be constrained to find landmarks with similar archways or buildings with orange roofs.



(a) Google Landmarks

(b) VGG-Faces

(c) TraffickCam Hotel Rooms

Figure 8: Each row shows the regions of the images most representative of the class membership.

5. Conclusions

We present an approach to visualize the image regions responsible for pairwise similarity in an embedding network. While some previous work visualizes a few of these components, we find that the top few components do not explain most of the similarity. Our approach explicitly decomposes the entire similarity score between two images and assigns it to the relevant image regions.

We illustrate a number of possible ways to use this visualization tool, exploring differences in networks trained with max pooling and average pooling, illustrating how the focus of a network changes during training, and offering an approach that uses this spatial similarity decomposition to search for matches to objects or sub-regions in an image.

The research area of similarity networks is quite active, exploring variations in the pooling strategies, learned or explicitly pre-defined linear transforms of the pooled feature, and boosting and ensemble strategies. We will share our code with the goal that the visualizations will provide additional insight about how embeddings are affected by these algorithmic choices.

References

[1] E. Ahmed, M. Jones, and T. K. Marks. An improved deep learning architecture for person re-identification. In *IEEE Conference on Computer Vision and Pattern Recognition*, June 2015.

[2] D. Bau, B. Zhou, A. Khosla, A. Oliva, and A. Torralba. Network dissection: Quantifying interpretability of deep visual representations. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3319–3327. IEEE, 2017.

[3] D. Bau, B. Zhou, A. Khosla, A. Oliva, and A. Torralba. Network dissection: Quantifying interpretability of deep visual representations. In *Computer Vision and Pattern Recognition*, 2017.

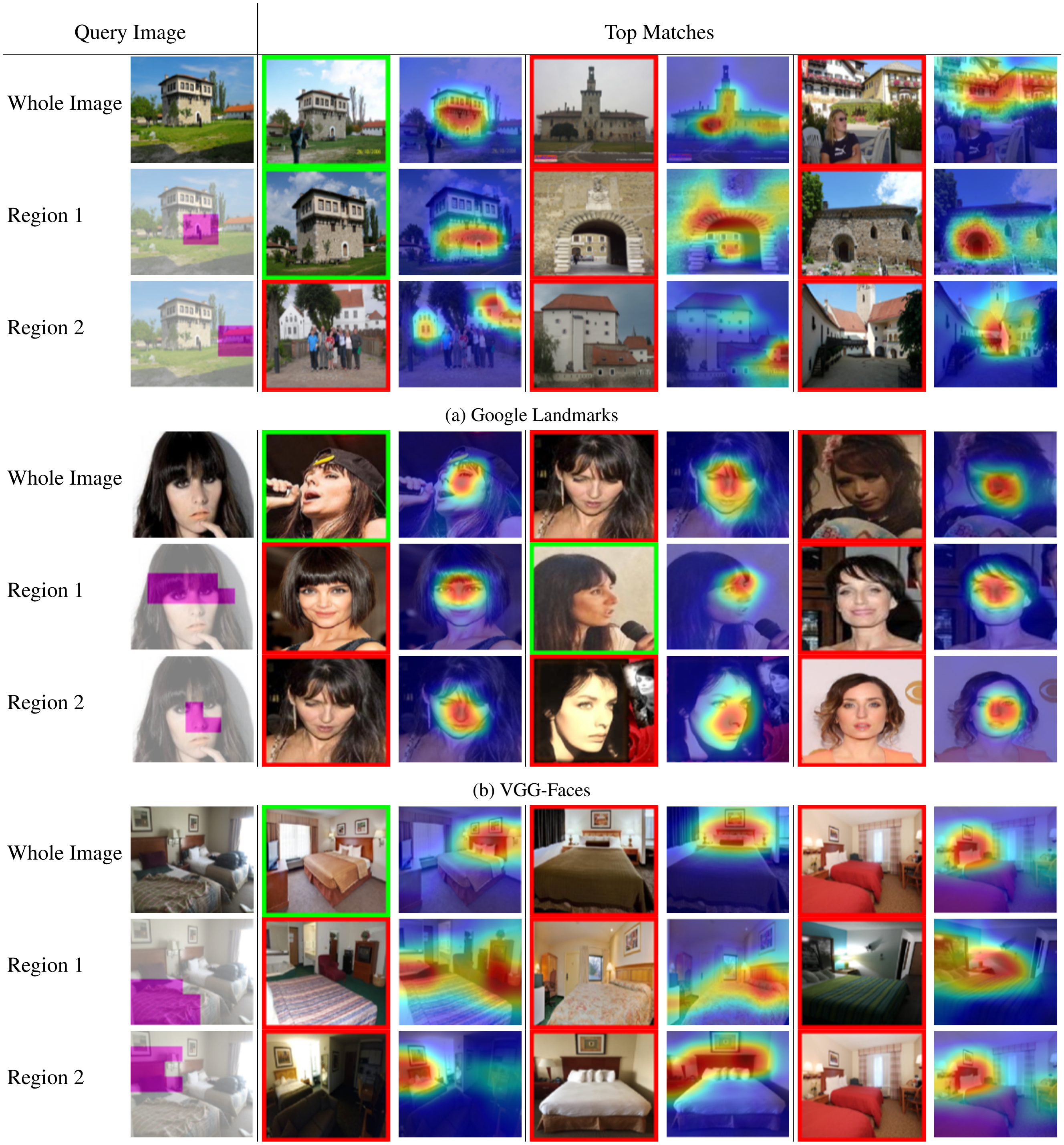
[4] S. Bell and K. Bala. Learning visual similarity for product design with convolutional neural networks. *ACM Transactions on Graphics (TOG)*, 34(4):98, 2015.

[5] Q. Cao, L. Shen, W. Xie, O. M. Parkhi, and A. Zisserman. VGGFace2: A dataset for recognising faces across pose and age. In *International Conference on Automatic Face and Gesture Recognition*, 2018.

[6] D. Erhan, Y. Bengio, A. Courville, and P. Vincent. Visualizing higher-layer features of a deep network. Technical Report 1341, University of Montreal, June 2009. Also presented at the ICML 2009 Workshop on Learning Feature Hierarchies, Montréal, Canada.

[7] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[8] A. Hermans*, L. Beyer*, and B. Leibe. In Defense of the Triplet Loss for Person Re-Identification. *arXiv preprint arXiv:1703.07737*, 2017.



(a) Google Landmarks

(b) VGG-Faces

(c) TraffickCam

Figure 9: Object- and Region-Specific Retrieval. We show the most similar image from the three most similar classes when using either the whole image as the query input, or selected sub-regions of the image. This allows for object- or region-specific image retrieval; for example, “find landmarks with similar archways”, “find faces with brunette bangs” or “find hotels with similar looking bedspreads.”

[9] Y. Movshovitz-Attias, A. Toshev, T. K. Leung, S. Ioffe, and S. Singh. No fuss distance metric learning using proxies. In *IEEE International Conference on Computer Vision*, Oct 2017.

[10] O. M. Parkhi, A. Vedaldi, and A. Zisserman. Deep face recognition. In *British Machine Vision Conference*, 2015.

[11] F. Radenović, G. Tolias, and O. Chum. CNN image retrieval learns from BoW: Unsupervised fine-tuning with hard examples. In *European Conference on Computer Vision*, 2016.

[12] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[13] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 815–823, 2015.

[14] K. Simonyan, A. Vedaldi, and A. Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *CoRR*, abs/1312.6034, 2013.

[15] H. O. Song, Y. Xiang, S. Jegelka, and S. Savarese. Deep metric learning via lifted structured feature embedding. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4004–4012. IEEE, 2016.

[16] A. Stylianou, A. Norling-Ruggles, R. Souvenir, R. Pless, undefined, undefined, undefined, and undefined. Indexing open imagery to create tools to fight sex trafficking. *2015 IEEE Applied Imagery Pattern Recognition Workshop (AIPR)*, 00:1–6, 2015.

[17] Y. Sun, Y. Chen, X. Wang, and X. Tang. Deep learning face representation by joint identification-verification. In *Advances in neural information processing systems*, pages 1988–1996, 2014.

[18] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[19] G. Tolias, R. Sicre, and H. Jégou. Particular object retrieval with integral max-pooling of cnn activations. In *International Conference on Learning Representations*, 2016.

[20] E. Ustinova and V. Lempitsky. Learning deep embeddings with histogram loss. In *Advances in Neural Information Processing Systems*, pages 4170–4178, 2016.

[21] N. Vo and J. Hays. Generalization in metric learning: Should the embedding layer be the embedding layer? *arXiv preprint arXiv:1803.03310*, 2018.

[22] J. Wang, Y. Song, T. Leung, C. Rosenberg, J. Wang, J. Philbin, B. Chen, and Y. Wu. Learning fine-grained image similarity with deep ranking. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1386–1393, 2014.

[23] D. Yi, Z. Lei, S. Liao, and S. Z. Li. Deep metric learning for person re-identification. In *International Conference on Pattern Recognition*, pages 34–39. IEEE, 2014.

[24] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. *CoRR*, abs/1311.2901, 2013.

[25] Y.-T. Zheng, M. Zhao, Y. Song, H. Adam, U. Buddemeier, A. Bissacco, F. Brucher, T.-S. Chua, and H. Neven. Tour the world: building a web-scale landmark recognition engine. In *IEEE Conference on Computer Vision and Pattern Recognition*, Miami, Florida, U.S.A, June, 2009.

[26] B. Zhou, A. Khosla, À. Lapedriza, A. Oliva, and A. Torralba. Learning deep features for discriminative localization. *CoRR*, abs/1512.04150, 2015.

[27] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Object detectors emerge in deep scene cnns. In *International Conference on Learning Representations (ICLR)*, 2015.