Image Analogies for Visual Domain Adaptation

Jeff Donahue
UC Berkeley EECS
jdonahue@eecs.berkeley.edu

Abstract

In usual approaches to visual domain adaptation, algorithms are used to infer a common "domain transform" between the feature space of a source domain and that of a target domain. We propose a new approach in which we use "image analogies" to infer a domain transform that can be applied to images *rather than features. This approach is applicable for domain pairs in which the domain transform is known to be approximately a filter or texture change. By comparing the use of analogous images for training examples to the use of the original images for training examples in a discriminative classifier for the target domain, we show that image analogies can effectively capture certain types of domain transformations.*

1. Introduction

Domain adaptation is an important challenge in modern computer vision. Torralba and Efros [13] showed that when standard object recognition algorithms are trained on one benchmark dataset (e.g., Caltech, PASCAL, ImageNet) and tested on another, performance suffers drastically (versus testing on the dataset on which the classifier was trained). They further demonstrate dataset bias by showing that researchers who work with these datasets regularly can easily discriminate between them given just a few examples of each.

Domain adaptation is also essential in real world applications of object recognition. For example, if an autonomous vehicle is trained to detect signs, obstacles, etc. in a laboratory environment, its accuracy will diminish significantly when reapplied to the real world. In such applications where recognition accuracy could make the difference between life and death, strong domain adaptation is required.

In general, it may be impossible to train any vision system on a "true random sample" of the objects of interest "in the wild," as it seems that all existing datasets have some type of bias due to the methodology with which they were constructed, so training on a biased sample of the objects we want to recognize and then rely on domain adaptation to generalize what we learned in training. For these reasons, domain adaptation is a problem that cannot simply be brushed aside if visual object recognition research is to be practical and useful in the real world.

In this paper, we attempt to address a subset (or a restricted version) of the domain adaptation problem. The method we propose here will not solve the "general" domain adaptation problem, as it requires that the *domain transform*, or the theoretical operation that turns an example of the source domain into an example of the target domain, can be approximated by an image filter or a change in the image's texture. More explicitly, the domain transform must be possible to approximate using the *image analogies* algorithm described by Hertzmann *et al.* in [7]. While prior work in domain adaptation has attempted to find a domain transformation in the *feature space* of the image [10, 11], our work in this paper tries to find a domain transformation in the *image space*. As such, our approach may be compatible with existing domain adaptation techniques as it occurs at a different point in the pipeline.

The image analogies algorithm takes as input three images A, A', and B. Images A and A' are, respectively, the *unfiltered* and *filtered* source images. Image B is an unfiltered target image, and the output is a new image B' that is related to B in the same way as A' is related to A. In other words, $A : A' :: B : B'$ [7]. See Figure 1.

Our work in this paper shows how this algorithm can be used to create *synthetic examples* of a target domain to be used as training examples for that domain. If we have a single pair of corresponding images in the source and target domain and any number of examples of the source domain, we can "learn" the image space transformation based on our single example pair between the source and target domain, and then apply this transformation to all of our existing examples of the source domain, giving us corresponding synthetic examples of the target domain.

As mentioned above, the two domains must differ by an image filter that the image analogies algorithm can recognize. This fact limits the applicability of this paper - it cannot solve the general domain adaptation problem, in which our domains might be (for example) different datasets, as

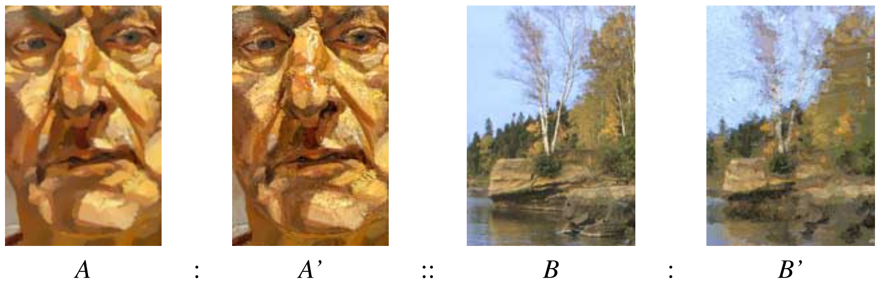

Figure 1. **Image analogy example:** The image analogies algorithm proposed by Hertzmann *et al*. [7] takes as input an unfiltered image *A*, a filtered image *A'*, and another unfiltered image *B*, outputting an "analogous" image *B'* that relates to *B* in the same way as *A'* relates to *A*. Figure taken from [7].

discussed in the Torralba *et al*. paper [13]. However, this approach is nonetheless useful in the real world, as there are indeed useful domain pairs that differ only by a filter.

One example of a pair of domains in which our approach might be applicable, and the one that inspires the experiment in this paper, is two different types of cameras, such as (domain A) photos taken by a high-resolution DSLR camera, and (domain B) photos taken by a low-resolution webcam. In this case, the difference between the two domains could be approximated by a Gaussian blur and some Gaussian noise, a uniform filter that the image analogies algorithm can potentially learn. The only caveat is that we must have a single pair of images that correspond *exactly* (i.e., pixel-for-pixel) taken by the two cameras.

Another potential application might be in recognizing objects in paintings based on examples from photographs, assuming that the painting attempts to represent the inspiration behind it in a way that generally preserves its proportions (as in a painting by Monet or da Vinci), rather than a more abstract representation (as in a painting by Picasso). If we have a photograph of the scene or object that the artist was painting and the painting itself, we can use image analogies to learn the domain transformation between photographs and paintings done by the artist and in the style of which we have an example.

In this paper, we take the *DSLR* domain from the office dataset [11] and creating a simulated *"webcam"* domain by decreasing the resolution of the *DSLR* images and adding to them a Gaussian blur and Gaussian noise. We test the effectiveness of a classifier that is trained simply on the original *DSLR* images, and compare the accuracy of such a classifier to the accuracy of our approach, in which we train a classifier on *image analogies* generated using a single pair of images: an original *DSLR* image and the corresponding synthetic *"webcam"* image generated from it. We demonstrate the usefulness of these image analogies in building a classifier by showing that the classifier accuracy is much higher when training on the *image analogies* than when training on the corresponding original *DSLR* images.

2. Related Work

Much recent work in computer vision and machine learning has considered the problem of learning transformations (see [2, 4, 6, 10–12] for some examples in computer vision research). However, the key distinction in our work is that these papers all look for transformations in the feature space rather than the image space.

There is also a large body of work in the graphics community that addresses the problem of texture synthesis and mapping. In [5], the researchers describe an automated method of synthesizing images with a texture matching a given input image. [1] improves on previous approaches to texture synthesis by better preserving local structures with a parameter that controls randomness. The work in [7] particularly inspires our work. It takes an input of an untextured image and a textured image to "learn" the texture, and then applies the learned texture to a different untextured image, outputting the "analogous" image that relates the second untextured image in the same way that the given textured image relates to the first untextured image. All of this work looks at the resulting images with synthesized textures as ends unto themselves (a valid viewpoint, as many of the results are quite impressive), and do not propose the possibility of reapplying the resulting images to aid in visual recognition tasks. To our knowledge, this work is the first instance of looking at texture-based transformations in the image space and applying them to the problem of domain adaptation.

3. Approach

The main idea is to use leverage the idea of *image analogies* [7] to create a stronger classifier in a target domain, given many training examples of a source domain, which is differentiated from the target domain by a filter or a change in texture. For an overview of the algorithm we use, see Figure 3.

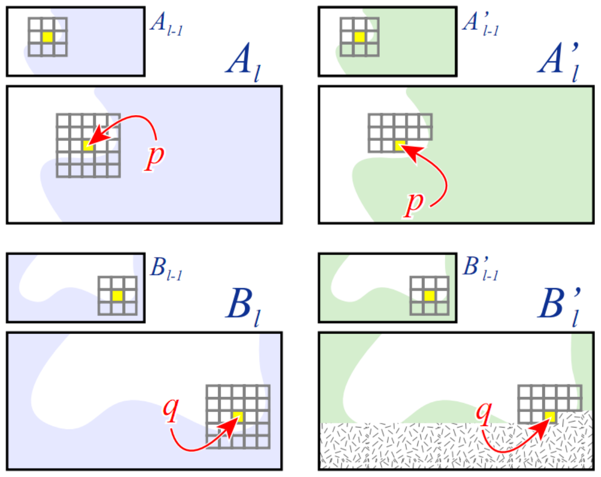

Figure 2. **Image analogies algorithm:** The image analogies algorithm proposed by Hertzmann *et al*. [7] works by looking at the pixel neighborhood *q* in images *B* and partially synthesized *B'* and finding the nearest neighbor *p* of images *A* and *A'*. The algorithm is run at multiple resolutions or "levels"; fine-grained level *l* and coarse-grained level $l - 1$. Figure taken from [7].

3.1. Image Analogies

Central to our approach is the technique proposed by Hertzmann *et al*. [7] for synthesizing *image analogies*. We briefly review it here; see [7] for details.

Suppose we have a pair of images *A* and *A'*, where *A'* corresponds to *A* but with some filter or texture applied. Then, we have another image *B*, and we want to create an image *B'* which relates to *B* in the same way as *A'* relates to *A* (see Figure 1). The image analogies algorithm of [7] gives us a method of synthesizing such an image *B'* given inputs *A*, *A'*, and *B*.

In essence, the algorithm of [7] works by going through each pixel in image *B* and the partially synthesized image *B'*, finding the closest match in images *A* and *A'* in a pixel neighborhood of a given size *n* x *n*. The nearest-neighbor search is done in feature space, where the features for each pixel include the RGB values at that pixel, along with other information like the luminance and responses to certain filters. (In our approach, we convert the image to HSV color space and use the "V" (value) channel as our only feature.) Once the pixel index of the nearest neighbor is found, the pixel at that index in *A'* is used as the output pixel in synthesized image *B'*. See Figure 2.

The authors additionally detail several ways to improve the algorithm, including running the algorithm at multiple resolutions to more accurately capture filters of varying size, speedup methods like using an approximate nearest-neighbor search, and using a "coherence search" to preserve coherence with the neighboring pixels in the synthesized image. We will not go into detail about these improvements here; please see [7] for more information.

3.2. Analogies for Domain Adaptation

We leverage the idea of image analogies [7], described in Section 3.1, to improve classification accuracy in object recognition across domains. The problem we attempt to solve is as follows: *Given examples of objects* $x_1, x_2, ..., x_n$ *and their labels* $y_1, y_2, ..., y_n$ *in a source domain, and a single example of an image* I' *in a target domain which corresponds pixel-for-pixel to some image* I *in the source domain, how can we build a classifier of the target domain?* In this context, the source and target domain must differ by a filter or texture; something that can be detected by the image analogies algorithm.

For example, the source domain might be a set of photographs taken using a high-resolution, high-quality DSLR camera, and the target domain might be a different set of photographs taken using a low-resolution, low-quality webcam. In this case, our problem statement would require a number of labeled examples of the object categories of interest taken by the DSLR camera, along with a single pair of photographs, one taken by the DSLR camera and the other taken by the webcam, that are of the same scene, corresponding pixel-for-pixel.

Given the problem statement and information above, the basic approach should now be relatively obvious. We will use the image pair I and I' to "learn" the domain transformation between the source and target domains using the image analogies algorithm, generating an analogous image for each of the examples we have of the source domain. This gives us a corresponding analogous image for each example we have of the source domain. In terms of the image analogies language, we generate x'_i using $I : I' :: x_i :: x'_i$, where x_i is an example from the source domain. If our intuition is correct, these analogous images x'_i that we have generated should correspond closely to how each of the corresponding original images x_i from the source domain would appear if they were in the target domain.

We proceed by taking these analogous images x'_i to be the training examples for our classifier of the target domain, each having the same label y_i as the original image from the source domain x_i from which it was generated. Based on our intuition that these analogous images x'_i are a better representation of what the image would look like in the target domain than the original images x_i, a classifier for images in the target domain trained on examples x'_i should be more accurate than one trained on examples x_i. Note that since our approach operates in image space (rather than, for example, feature space), it can be used with any choices of image features, representation, and classifier.

3.3. Image Representation and Classification

In this section, we detail our choices of image representation and classification algorithm. As we noted in Section 3.2, this approach to domain adaptation operates in image space, and as such, it can be used with choice of any image representation and classification algorithm. Since our goal is only to convince the reader that image analogies are useful for domain adaptation, our choices for representation and classification algorithm are ones that we believe to be popular and well-understood, sacrificing state-of-the-art performance (to which we could not compare any other algorithm, as our particular problem has not been addressed) for simplicity and reproducibility.

For our image features, we use *SIFT descriptors* [9] computed at keypoints using the VL_SIFT function from the VLFeat library [14] with default settings. A single SIFT descriptor [9] describes a square image patch of a given size and orientation as a 128-dimensional vector, where the square is divided into a 4 x 4 grid and the gradient response in each of 8 directions inside one square of the grid are the elements of the 128-dimensional descriptor vector.

Once we have SIFT descriptors for all the images in our database, we cluster them using the k-means algorithm, with $k = 800$, giving us 800 cluster centers $\vec{c_1}, \vec{c_2}, ..., \vec{c_{800}}$, each of which is a 128-dimensional vector. We then generate a *bag-of-words histogram* for each image. A bag-of-words histogram is a k-dimensional vector $\vec{v}$, where each v_i corresponds to the number of SIFT descriptors in the image whose nearest cluster center generated by k-means was cluster center c_i. Finally, our image representation is simply the vector normalized version of this bag-of-words histogram, $\vec{v'} = \frac{\vec{v}}{||\vec{v}||}$.

Given a vector representation for each image, we can use standard discriminative classification algorithms to build a model of each category we are interested in classifying. In particular, we use a standard one-vs.-all linear support vector machine (*SVM*), making use of the LIBSVM [3] SVM library with all default parameters (except the C parameter) to perform the actual training and classification. A linear SVM takes as input any number of training examples and their class labels, and finds a margin that maximizes the distance between positive and negative examples (allowing for classification error, the importance of which is set by a parameter C, which we set to $C = 1000$). A standard linear SVM only classifies examples into two classes ("positive" and "negative"), but this can be turned into a multi-class classifier by training a single SVM for each class, where the positive class is all examples of the class of interest, and the negative class is all examples of all other classes. Then, a novel example is classified as whichever class's SVM gives the *highest* (most positive) response.

```
Algorithm CreateTargetDomainClassifier
Input: Source domain example images $x$ and corresponding labels $y$ $[(x_1, y_1), ..., (x_n, y_n)]$, image in source domain $I$, corresponding image in target domain $I'$
Output: $M$, a one-vs.-all SVM model of the target domain
1.  for ($i \leftarrow 1, ..., n$)
2.      $x'_i \leftarrow$ SynthesizeAnalogousImage($I$, $I'$, $x_i$)
3.      $d_i \leftarrow$ ComputeSIFTDescriptors($x_i$)
4.      $d'_i \leftarrow$ ComputeSIFTDescriptors($x'_i$)
5.  $centers \leftarrow$ KMeans($[d_1, d'_1, ..., d_n, d'_n]$, $k = 800$)
6.  for ($i \leftarrow 1, ..., n$)
7.      $v'_i \leftarrow$ ComputeBoWHistogram($d'_i$, $centers$)
8.  $M \leftarrow$ TrainLinearSVM($[(v'_1, y_1), ..., (v'_n, y_n)]$)
```

Figure 3. **Algorithm overview:** The above pseudocode represents the "algorithm" we use for the problem statement given in Section 3.2. See Sections 3.1, 3.2, and 3.3 for details on each step. We include the above algorithm summary mainly to aid in understanding, not as an implementation suggestion. In practice, as we will do multiple runs with different sets of training examples, we will want to precompute the analogous images for all examples we have. We also precompute the cluster centers produced by k-means and the bag-of-words histograms for all analogous images. Therefore, in each trial run of our algorithm, we only select a subset of the examples available to us for training (using the rest for testing), and then train the SVM on the selected examples.

4. Data

We explore the utility of our approach on the Office dataset from [11]. This dataset is intended as a domain adaptation benchmark, with 31 categories (each category being a type of object one might find in an office, e.g., *backpack*, *keyboard*, *headphones*), and 3 *domains*: *Amazon*, *DSLR*, and *webcam*. While we have repeatedly cited as an example application of our approach the possibility of classifying images taken from a webcam using labeled example images taken from a DSLR, we unfortunately cannot directly use the *DSLR* and *webcam* domains from this dataset to test our approach, as it lacks a critical element: a pair of photos that correspond pixel-for-pixel between the two domains. Some images between the *DSLR* and *webcam* domains of the dataset are very similar (indeed, some are of the same object in the same position), but have a slight change in perspective or orientation that would cause the image analogies algorithm to fail.

Because it is somewhat difficult to create the setup needed to run a real-world experiment on our approach (positioning two different cameras in the exact same way to get two images with a pixel-for-pixel correspondence), we have chosen, for now, to test our approach on synthetic data. Specifically, we use the actual *DSLR* domain of the Office dataset as our source domain (see the left column of Figure 5 for sample images), and then create a syn-

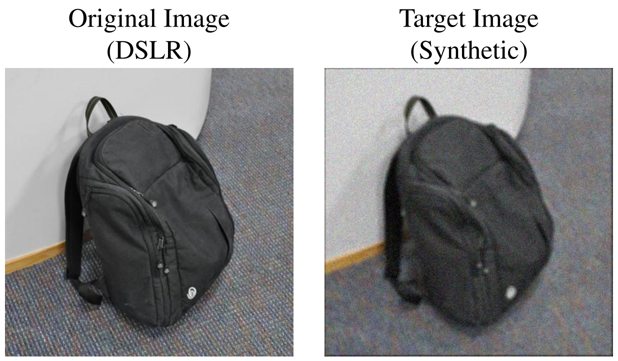

Figure 4. **Image pair used to generate all analogies:** The above single pair of images was used to generate all analogous images for the experiments run in this paper. On the left is the first image in the *backpack* category of the DSLR domain of the Office dataset [11]. On the right is the synthetic image generated from the original (left) by adding Gaussian blur and Gaussian noise. See the right column of Figure 5 for sample images generated by the image analogies algorithm [7] using this image pair (along with the original image in the left column).

thetic "webcam-like" target domain by taking each image and reducing its resolution, adding a Gaussian blur, and then adding Gaussian noise. As the description suggests, this procedure produces low-resolution, blurry, noisy images that are seemingly similar in appearance to those that a low-quality commercial webcam might produce. See the middle column of Figure 5 for samples of the synthetic target domain images produced.

To compute the analogous images, we chose a single image from the source domain (the *DSLR* domain of the Office dataset [11]) and the target image generated from that source image. In particular, we used the first image in the *backpack* category of the dataset (for no reason other than the fact that it happens to be the first image in the first alphabetical category; any other choice of image would presumably have produced similar results). See Figure 4 for the image used. Then, we use the image analogies algorithm [7] to generate an analagous image for each example in the *DSLR* from our one example of the source domain and its corresponding target domain image. Note that the resolution of the source domain image must be reduced to match the target domain image, as this is necessary for the image analogies algorithm. Due to time constraints (the image analogies MATLAB implementation used is quite slow), we generated analogies for only the first 13 of the 31 categories in the Office dataset, so our results will be on only these 13 categories.

Interestingly, many of the images produced for this dataset using the image analogies algorithm are quite "bad" in the sense of being realistic, with noticeable artifacts like large grey blotches in certain parts of many of the images. See the right column of Figure 5 for samples of the analogous images produced.

5. Results

We now present results to demonstrate that image analogies can be valuable in building classifiers for an unknown target domain based on examples from a source domain. In Section 4, we described the data that we will use in our experiments, consisting of the first 13 categories of the DSLR domain from the Office dataset [11].

The classification task that we are interested in is to classify the target domain images - the synthetic images that were generated by applying a Gaussian blur and Gaussian noise to the original DSLR images (the middle column in Figure 5). In our approach, we train classifiers on the analogous images generated from the single pair of source and target images available to us (Figure 4), as described in Section 3. See the right column of Figure 5 for samples of the images we use to train SVM classifiers in our method.

5.1. Baselines

We compare our method against three baselines. Each of the baselines uses the exact same methodology as we use in our method described in Section 3.3 (same features, same classification algorithm, etc.), but with a different set of training examples.

The first baseline, *DSLR*, is perhaps the most obvious: rather than training on the images generated by analogy (our approach), simply train on the original DSLR images (the left column in Figure 5). If our approach cannot outperform this baseline, there is no reason to generate training data using image analogies rather than simply using the original images. Hence, this baseline is a useful way to determine whether the classifier is getting any useful information from the image analogies algorithm.

The second baseline, *DSLRSmall*, is a modified version of the first: train on the original DSLR images, resized to the same resolution as the images in the target domain. Due to the way that SIFT keypoints are selected, if we use the original (large) DSLR image results, we will get SIFT descriptors spanning a much smaller image patch (relative to the image's content) than the SIFT descriptors computed on the target data. Hence, to compare our algorithm fairly, removing the possibility that any improvement is only due to our examples being of the same resolution as the target domain images, we also compare against the *DSLRSmall* baseline.

The third baseline, *Ideal*, will give the "ideal" performance in the sense that it is trained on images from the actual target domain. Hence, there is no domain adaptation in this baseline - we are training on images from the same domain that we are testing on, essentially making it a traditional image classification task. We do not need to outperform this baseline in order to show that image analogies

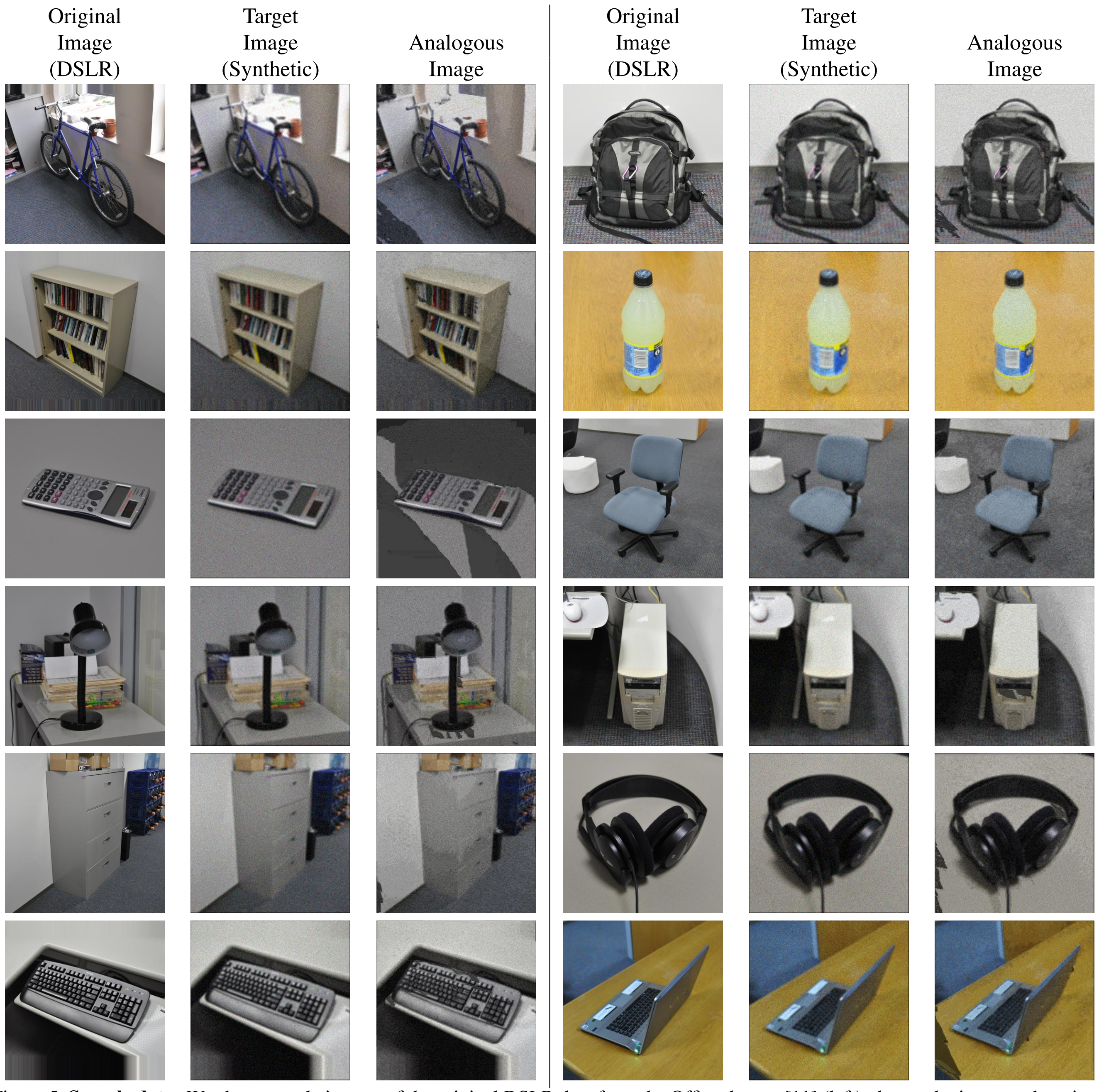

Figure 5. **Sample data:** We show sample images of the original DSLR data from the Office dataset [11] (left), the synthetic target domain version of the image (middle) (generated by adding Gaussian blur and Gaussian noise to the original image), and the image generated using an image analogy based on the single corresponding image pair shown in Figure 4 and the original DSLR image. Note that many of the analogous images could be described as failure cases for the image analogies algorithm, clearly showing undesirable artifacts, like the large grey blotches in the analogous images of the bicycle and the calculator.

are useful, we only include it for the purpose of analysis. If the performance of our method came anywhere near this baseline, it would mean that, for this data, training on image analogies is nearly as good as training on the target domain itself.

5.2. Experiment Setup

To test our approach, we run 100 trials, where in a given trial we randomly choose 5 training examples from each class, and use the remaining images to test classification accuracy. We use the same images for our approach and all of the baselines in each trial, so that results on a given trial can be compared directly.

To reiterate, in our approach, we train on images generated by analogy (right column of Figure 5). In the *DSLR* and *DSLRSmall* baselines we train on the original DSLR images (left column of Figure 5) at the original resolution and the resolution of the target class (respectively). In the *Ideal* baseline we train on the synthetic (target) domain on which we will test (middle column of Figure 5). We test our approach and all three examples on the synthetic (target) domain (middle column of Figure 5) - all images that were not used for training.

5.3. Experimental Results

We find from the experiment described above that our approach greatly outperforms both of the *DSLR* baselines, and not far off from *Ideal* performance. See Figure 6.

While both *DSLR* baselines do better than chance (7.69%), the *DSLRSmall* baseline greatly outperforms the *DSLR* baseline, more than doubling its performance. This is likely due to the effect that the resolution difference has on the computed SIFT descriptors, detailed in .

Even in comparison to the stronger baseline, *DSLRSmall*, our method is clearly superior. Our performance is 21% higher than the performance of *DSLRSmall*.

In comparison with the *Ideal* baseline's performance, our method, of course, cannot compete. However, considering that the *Ideal* baseline is trained on the same domain on which it is tested, we expect that its performance should be higher. Taking this into account, the fact that our method results in 90% of the classification accuracy of the *Ideal* baseline seems to show that our approach is fairly strong.

These results show that, at least for this synthetic data, image analogies are a highly useful tool in classifying images of an unknown domain. The *DSLR* baselines were given the same information as our algorithm (images from the DSLR dataset along with their labels), with the exception of the single pair of corresponding images from the source and target domain that our algorithm uses to generate the image analogies. From this single extra image pair and the image analogies algorithm of [7], we get a new approach to image classification in a target domain that boosts performance by over 20%, nearing the performance of training on the target data itself.

Method	**Classification Accuracy**
Ours - Image Analogies	63.79%
Baseline 1 - DSLR	23.56%
Baseline 2 - DSLRSmall	52.65%
Baseline 3 - Ideal	70.75%
Chance - 1/(13 classes)	07.69%

Figure 6. **Experimental results:** A table of results from our image classification experiment described in Section 5. Our approach significantly outperforms both *DSLR* baselines (improving on the better of the two, *DSLRSmall*, by 21%), and is not far from the performance of the *Ideal* baseline (90% of its accuracy), for which there is no domain adaptation problem (in that it is trained and tested on the same domain). All methods had statistically significantly different performance, based on a paired t-test with $\alpha = 0.0001$, and all had better than chance performance.

6. Conclusions

We have presented a new approach to domain adaptation which operates in the image space, rather than in a feature space. This approach is applicable to domain pairs where the domain transformation between them can be approximated by a filter or change in texture. Based on a corresponding pair of images in the source and target domains, we can use the image analogies algorithm described by Hertzmann *et al*. in [7] to create additional examples of the target domain from each example of the source domain.

Our experimental results in Section 5.3 show that these image analogies can be highly useful in training a classifier in the unknown domain. The approach of training a classifier on image analogies beats a strong baseline by over 20%, demonstrating that the analogous images synthesized from the original images give more information to the classifier than the original images alone. The lower performance of the baseline further shows that filtering an image in the way that we did to generate the target domain (adding Gaussian blur and Gaussian noise), can significantly confuse a classifier, a result that might not be obvious given the faith we place in features like SIFT descriptors to recognize semantic objects without being distracted by artifacts like blurring and noise. Luckily, our results clearly show that image analogies can address some of the weaknesses in our image representation.

While this experiment was not performed on a "real-world" domain transformation, in that the target domain was generated from the source domain by applying a filter, it isn't unreasonable to speculate that the results of the experiment would likely be similar if using actual webcam images as the target domain. The images generated by the procedure explained above seems to produce images that appear very similar to the type of photographs one expects a webcam to produce.

7. Future Work

This project has a number of possible future directions that merit further investigation.

While we believe our results constitute a legitimate "proof of concept" for the idea of using image analogies for domain adaptation, it would of course be more convincing if we could replicate the experiment on real-world data, where the source domain is again images taken by a DSLR camera and the target domain is images taken by a webcam (or some other type of low-resolution, low-quality camera). This has the drawback of being more difficult to setup, potentially requiring two cameras to be set up in the exact same position and pose, to take a snapshot of the exact same scene, such that each pixel index in the two images corresponds to the same point in the scene. This is further complicated by the fact that different cameras output different resolutions and aspect ratios. Fortunately, our system requires just a single pair of input images that correspond in this way, giving the potentially painful task of setting up two cameras perfectly a reasonably high payoff (assuming the results found in this paper carry over to the real-world task).

It is also possible that the cameras might not need to be set up *perfectly*. The authors of [7] describe (but do not show, for copyright reasons) image analogies results in which the source image pair is a photograph and a realistic painting of the scene shown in the photograph, and the output is reasonable. Due to human error, there is no way that the artists' paintings were pixel-for-pixel reproductions of the scene, so the authors apply a warping method to the painting in order to bring it into pixel-for-pixel correspondence with the photograph of the scene. Potentially, such a warping approach could be applied to correct a less-than-perfect pair of corresponding input images in the DSLR/webcam problem we're interested in as well.

Other areas where image analogies might prove useful in domain adaptation might be in classifying images filtered using "Instagram" type filters that are becoming increasingly commonplace in recent years (e.g., sepia filters). Such filters are often intended to produce a pleasing "retro" effect that usually reduces the quality of the final image, similar to the synthetic domain we tested here (which added a blur and some noise). See Figure 7 for examples generated using an online image editing tool, Picnik [8]. The attractive thing about this idea, of course, is that it is a useful real-world application while still requiring no "hardware" camera setup, as an example pair can be produced simply by taking a "normal" photograph and filtering it using the filter for which we are interested in doing object recognition.

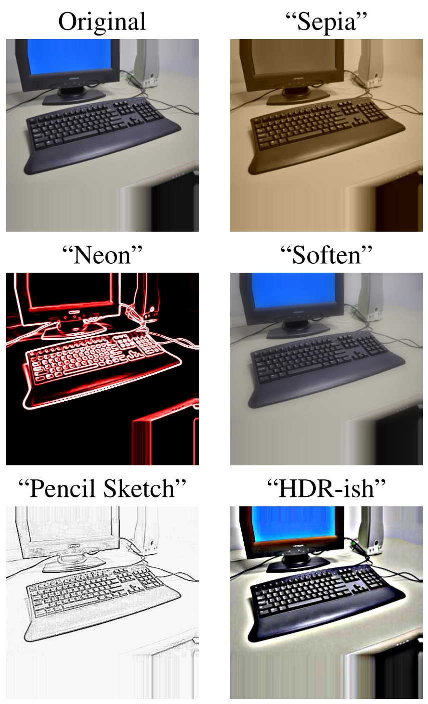

Figure 7. **Future directions for image analogies for domain adaptation:** In recent years, "Instagram" type filters are becoming increasingly popular, often intended to produce a pleasing "retro" effect that generally reduces the quality of the image. Our approach could be used to do object recognition on domains like those above, given a single "normal" photograph and its corresponding filtered version, plus normal photographic examples of the objects we are interested in classifying in the filtered domain. The original image is from the Office dataset [11], and the other images were generated using filters from an online image editing tool, Picnik [8].

References

[1] A. Efros and T. Leung. Texture Synthesis by Non-parametric Sampling. In *ICCV*, 1999.

[2] B. Kulis and P. Jain and K. Grauman. Fast Similarity Search for Learned Metrics. In *CVPR*, 2004.

[3] C. Chang and C. Lin. *LIBSVM: a library for SVMs*, 2001.

[4] S. Chopra, R. Hadsell, and Y. LeCun. Learning Similarity Metric Discriminatively, with Application to Face Verification. In *CVPR*, 2005.

[5] D. J. Heeger and J. R. Bergeny. Pyramid-Based Texture Analysis/Synthesis. In *SIGGRAPH*, 1995.

[6] G. Chechik and V. Sharma and U. Shalit and S. Bengio. Large Scale Online Learning of Image Similarity through Ranking. *Pattern Recognition and Image Analysis*, 2009.

[7] A. Hertzmann, C. E. Jacobs, N. Oliver, B. Curless, and D. H. Salesin. Image Analogies. In *SIGGRAPH*, 2001.

[8] G. Inc. Picnik. http://www.picnik.com/, 2005.

[9] D. G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *IJCV*, 2004.

[10] K. Saenko, B. Kulis, and T. Darrell. What You Saw is Not What You Get: Domain Adaptation Using Asymmetric Kernel Transforms. In *CVPR*, 2011.

[11] K. Saenko, B. Kulis, M. Fritz, and T. Darrell. Adapting Visual Category Models to New Domain. In *ECCV*, 2010.

[12] T. Hertz and A. Bar-Hillel and D. Weinshall. Learning Distance Functions for Image Retrieval. In *CVPR*, 2004.

[13] A. Torralba and A. Efros. Unbiased Look at Dataset Bias. In *CVPR*, 2011.

[14] A. Vedaldi and B. Fulkerson. VLFeat, 2008.