Reproduction Study of “Relative Attributes”

Joseph G. Ellis
Columbia University – Spring 2013 – Reproducible Research
Original Work by: Devi Parikh and Kristen Grauman
Devi Parikh and Kristen Grauman, “Relative Attributes”, International Conference on Computer Vision (2011).

1. Introduction

“Relative Attributes” [1] has quickly become a seminal paper within the field of Computer Vision and Visual Search and Retrieval. This work was given the Marr Prize for Best Paper at the International Conference on Computer Vision in 2011, and has since been cited almost 50 times in 2 years. The work presents a novel and intuitive framework for zero-shot learning, which is learning to classify or retrieve an image class that a computer system has never seen before. This is obviously a very difficult task, but the author’s propose a framework that is both novel and intuitive. I will attempt to reproduce the experiment that the authors present in this work, and will explicitly state what I was able to reproduce and what I was not able to accomplish. The ICERM guidelines for reproducibility will also be used describe the relative ease of reproducibility of “Relative Attributes”, and suggestions to make this work more reproducible will be presented. All code and data that was used to create this report can be found at https://github.com/jellis505/reproducingcodes.

2. Relative Attributes Theoretical and Computational Overview

We begin by giving a brief overview of the computational work completed within the “Relative Attributes” paper. Each of the sections below will be discussed as to their relative ease of reproduction later in Section 3.

2.1 Dataset Collection

The paper first details the use of two distinct datasets to test their zero-shot learning experiments on. The two datasets are the Outdoor Scene Recognition (OSR) [2] dataset, and the second is a subset of the Public Figure Dataset (PubFig) [3]. Each dataset was made readily available by the authors' on the Relative Attributes website (http://filebox.ece.vt.edu/~parikh/relative.html). The OSR dataset contains images 2,688 images from 8 different scene classes described as coast, forest, highway, inside-city, mountain, open-country, street, and tall-building. The subset of images used from the PubFig dataset contains pictures of Alex Rodriguez, Clive Owen, Hugh Laurie, Jared Leto, Miley Cyrus, Scarlett Johansson, Viggo Mortensen, and Zac Efron. This subset of the dataset used contains 800 images, 100 from each of the celebrities mentioned above.

2.2 Feature Extraction

To make each of these images useable in the ranking formulation and framework presented below in section 2.3, feature descriptors must be extracted for each of these images. For the OSR dataset the 512-dimensional Gist [2] descriptor was used. This is a very popular feature descriptor in the computer vision field, and is widely used in a variety of applications. The features extracted for the PubFig dataset were the Gist descriptor and a 30 dimensional Lab color histogram concatenated together to create the feature descriptor for the PubFig dataset images.

2.3 Creating Relative Attributes

Each of the image categories then must be ranked against one another as to how much they exhibit a given attribute. For example, an image from the "forest" category is deemed as exhibiting more of the "natural" attribute than an image from the "tall-building" category. The OSR dataset is described by 6 attribute categories: natural, open, perspective, large-objects, diagonal-plane, and close-depth. The PubFig dataset images are then described using 11 attributes which are: masculine-looking, white, young, smiling, chubby, visible-forehead, bushy-eyebrows, narrow-eyes, pointy-nose, big-lips, and round-face. A relative ordering for image classes for each attribute within is then created. The relative ordering for the OSR dataset is provided in Table 1. These are the relative orderings that were used by the authors' in all of their experiments and tests, which I also used.

Table 1. Relative Ordering of Attributes for OSR Dataset

OSR Dataset Relative Ordering								
	tall-building	inside-city	street	highway	coast	open-country	mountain	forest
natural	1	2	2	3	4	4	4	4
open	1	2	2	4	4	4	3	1
perspective	7	5	6	4	2	1	3	3
large-objects	5	3	3	4	4	2	2	1
diagonal-plane	6	4	4	5	3	2	2	1
close-depth	4	4	4	4	1	3	2	4

2.4 Learning Relative Attributes

Given the relative ordering of the attributes in Table 1 the author's propose to create a ranking framework to learn the relative ranks based on the descriptor for a given image. To do this the author's proposed a novel implementation of RankSVM [4] to learn the ranks with imposed similarity constraints. The formulation presented in the paper is as follows. We are given a set of training images $I = \{i\}$, that are each described by a feature vector $\boldsymbol{x_i}$, and a set of attributes $\{a_m\}$. Based on the relative ordering of attributes for each image category we can also deduce some constraints, namely we create a set of ordered pairs of images $O_m = \{(i,j)\}$ and set of unordered pairs of images $S_m = \{(i,j)\}$ such that $\forall(i,j) \in O_m, i > j$, i.e. image i has a higher attribute rank than image j for attribute a_m. Similarly, $\forall(i,j) \in S_m, i \sim j$, i.e. image i and j have the same attribute rank for a given attribute a_m. Given these constraints the goal here is to learn a ranking function,

$$r_m(\boldsymbol{x_i}) = \boldsymbol{w_m^T x_i} \tag{1}$$

such that for a given set of images, we can recreate the relative ordering described above. To do this our ranking function must satisfy the following constraints:

$$\begin{aligned} \forall(i,j) \in O_m &: \boldsymbol{w_m^T} x_{\boldsymbol{i}} > \boldsymbol{w_m^T} x_{\boldsymbol{j}} \\ \forall(i,j) \in S_m &: \boldsymbol{w_m^T} x_{\boldsymbol{i}} = \boldsymbol{w_m^T} x_{\boldsymbol{j}} \end{aligned} \tag{2}$$

This problem is NP-hard and intractable, but we can compute an approximation using non-negative slack variables, which makes this work very similar to SVM classification. The authors use the RankSVM formulation proposed in [4], and to implement this they use the freely available implementation created by Olivier Chappelle, which can be found at http://olivier.chapelle.cc/primal/ [5].

2.5 Zero-Shot Learning Framework from Relative Attributes

Once the relative ranks are learned, the authors then discuss a simple framework for zero-shot learning classification based on these learned attribute ranks. We assume that for the zero-shot learning task we have a list of seen and unseen images from which we want to perform classification. We want to compare the unseen image classes to those image classes that are seen. For each seen image class the author's find the mean learned ranking score (u_m) for each attribute a_m, and with these learned ranking attributes create a mean vector of attribute rank scores for an image $\boldsymbol{u_i}$, where i represents the image class (forest, tall-building, etc.). The covariance matrix for all of these attribute rank scores is also estimated and denoted as $\sum_i$. These parameters are then used to model the seen classes as a Gaussian distribution such that each seen class can then be described as,

$$c_i^{(s)} \sim N(\boldsymbol{u_i}, \boldsymbol{\Sigma_i}). \tag{3}$$

Given the relative ordering of the classes based on their attributes and the seen class distributions the authors attempt to estimate the distribution of the unseen classes. Their framework is presented below, and is reprinted directly from the paper so as to minimize any confusion.

1.) If $c_j^{(u)}$ is described as $c_i^{(s)} > c_j^{(u)} > c_k^{(s)}$, where $c_i^{(s)}$ and $c_k^{(s)}$ are seen categories, then we set the m-th component of the mean $\boldsymbol{u}_{\boldsymbol{jm}}^{(u)}$ to $\frac{1}{2}(\boldsymbol{u}_{\boldsymbol{im}}^{(s)} + \boldsymbol{u}_{\boldsymbol{km}}^{(s)})$.
2.) If $c_j^{(u)}$ is described as $c_i^{(s)} > c_j^{(u)}$, we set $\boldsymbol{u}_{\boldsymbol{jm}}^{(u)}$ to $\boldsymbol{u}_{\boldsymbol{im}}^{(s)} - d_m$, where d_m is the average distance between the sorted mean ranking scores $\boldsymbol{u}_{\boldsymbol{im}}^{(s)}$'s for seen classes for attribute

a_m. It is reasonable to expect the unseen class to be as far from the specified seen class as other seen classes tend to be from each other.

3.) Similarly, if $c_j^{(u)}$ is described as $c_j^{(u)} > c_k^{(s)}$, we set $\boldsymbol{u}_{\boldsymbol{jm}}^{(u)}$ to $\boldsymbol{u}_{\boldsymbol{km}}^{(s)} - d_m$.

4.) If a_m is not used to describe $c_j^{(u)}$, we set $\boldsymbol{u}_{\boldsymbol{jm}}^{(u)}$ to be the mean across all training image ranks for $a_{m,}$ and the m-th diagonal entry of $\boldsymbol{\Sigma}_j^{(u)}$ to be the same.

5.) In the first three cases, we simply set $\boldsymbol{\Sigma}_j^{(u)} = \frac{1}{S}\sum_{i=1}^{S} \boldsymbol{\Sigma}_i^{(s)}$.

Then for a test image i, the ranking scores are computed, and denoted as $\boldsymbol{x_i}$. We classify this image as the class that returns the maximum probability based on the given Gaussian probability density function of each class.

3. Reproduction of Experiments

This section discusses the ability to which I was able to reproduce the experiments performed by the author's in "Relative Attributes". The section also highlights the actions that I performed to attempt to recreate this work. All of this work was completed on an HP Z620 Workstation with 8GB of RAM. Matlab was the software that was used for all computation throughout this experiment.

3.1 Dataset Collection

Collection of the datasets used for this work was accomplished with relative ease. The authors made their datasets freely available, and packaged them with information that was very useful in working with these datasets. The datasets were downloaded from the author's website at http://filebox.ece.vt.edu/~parikh/relative.html. Each dataset also came with a ".mat" data file that stored the class of each image, which images were used for training the ranking algorithm, the learned rank weights, the extracted features from each image, and the learned ranking score for each image. This data was readily available and easily accessed. I was able to successfully complete this portion of the reproduction effort with little problems.

3.2 Feature Extraction

The next step in the processing pipeline after collecting the dataset was to extract features from the images to utilize the ranking formulation proposed in section 2.4. The GIST descriptor was used as the feature descriptor for the scene images, and the PubFig dataset images were described using a concatenation of the Gist descriptor and a 45-bin Lab color histogram according to the paper. However, after collecting the data from the author's website I found that the images in the PubFig dataset were described by a feature vector that was 542 dimensions within the provided "data.mat" file. The GIST descriptor is 512-dimensions long, and therefore I assumed that instead of a 45 bin Lab color histogram as stated in the paper, the authors instead used a 30 dimensional Lab color histogram concatenated onto the GIST descriptor to create the feature vector for the PubFig dataset images.

The most commonly used implementation of the GIST descriptor provided by the original author takes some parameters as input along with the image. No parameters are stated within the relative attributes paper discussing how to extract GIST descriptors from the images. I used the extraction procedures detailed in the GIST author's tutorial which can be found at http://people.csail.mit.edu/torralba/code/spatialenvelope/, to perform my feature extraction. The results were satisfactory, and I was able to extract features that were very similar to those provided by the author's. I compared my extracted features to the author's using the formula below in equation (4), where $\boldsymbol{x}_i$ is the feature descriptor I extracted from image i and $\boldsymbol{y}_i$ is the feature descriptor provided by the authors for image *i*. The percent difference results between my features and the authors' can be seen in Table 2. The features I extracted from the OSR dataset were closer to the authors' provided features than those from the PubFig Dataset.

$$Percent\ Difference = \frac{\sum|\boldsymbol{x}_i - \boldsymbol{y}_i|}{\sum\left|\frac{\boldsymbol{x}_i + \boldsymbol{y}_i}{2}\right|_{L1}} \tag{4}$$

Table 2. Percent Difference of Features Extracted

	OSR Dataset	PubFig Dataset
Percent Difference	8.1046	36.9811

3.3 RankSVM Computation

After feature extraction the next step to conduct the authors' experiments was to implement their ranking formulation provided in section 2.4 of this paper. To do this I took the freely available Matlab implementation of RankSVM created by Olivier Chappelle, and modified the ranking formulation to take similarity constraints as input. This was the same procedure that was outlined in the paper and pursued by the authors. The authors have also made this portion of code freely available, and I have compared my implementation to theirs. Our results compared very favorably and it can be deduced that our ranking algorithms are performing the same operations. Using the similarity measure provided in equation 4 the percent difference between the ranks learned from my implementation of RankSVM with similarity and the authors' implementation is .00063% for the OSR dataset using 4 category pairs for training as discussed in the Section 3.4.

3.4 Training Scheme for Experiments

The author's also developed and detailed the training scheme that was developed for their experiments in Section 4.2 to learn the ranking scores from a given set of training images. The description of their training scheme can be seen in the excerpt below taken directly from the paper.

"Unless specified, we use 2 unseen and 6 seen categories. To train the ranking functions, we use 4 category pairs among seen categories, and unseen categories are described relative to the two closest seen categories for each attribute (one stronger, one weaker). We use 30 training images per class, and the rest for testing"

In my personal opinion this statement is highly ambiguous. There are multiple ways that the above statement could be interpreted and I therefore attempted multiple ways of choosing the O and S set constraints described in section 2.4 of this paper. A complete list can be seen below, including the last member of the list, which is the way that the author has stated they trained this ranking function. I began an email correspondence with the author, and only after this email correspondence was I able to deduce the proper way to select the constraints to train the ranking scores for the images. The proper way to do perform the training is in the 4th bullet provided below.

- Create 4 category pairs from the seen classes, and then compare each of the 30 training images from the first category in each pair to the second category in each pair for every attribute. This gives us a total of 30 constraints per category pair.

- For each seen category, randomly choose 4 other seen classes and compare each image in the seen category to one image from each of the other 4 seen classes. This gives 120 constraints per image category.
- For 4 seen categories, randomly choose 4 other seen classes and compare each image in the seen category to one image from each of the other 4 seen classes. This gives 120 constraints per image category.
- Create 4 unique categories pairs from the seen classes, and then compare each of the 30 training images associated with the first category in the pair to each of the 30 training images from the second category in the pair. This result in 900 constrains per image category.

The author of the paper has also quickly validated my code and work, and stated that it looks fine, and she believes this is also how she trained her ranking function. However, the ranking scores that I have found using this training scheme are slightly different than the ranks that are provided with the data by the author's. The results and differences can be seen in Figure 1 below, and correspond to the OSR dataset. The results presented below were found with no unseen classes, because we assume that the learned ranks provided with the author's dataset were learned with no unseen classes, or presumably this would have been expressly stated. All of these results were performed given the extracted features from the author to minimize the impact of all possible variables that could influence the ranking outcome except for the training scheme. Figure 1 shows the results of their learned ranks in blue, and my learned ranks in red with different numbers of training category pairs.

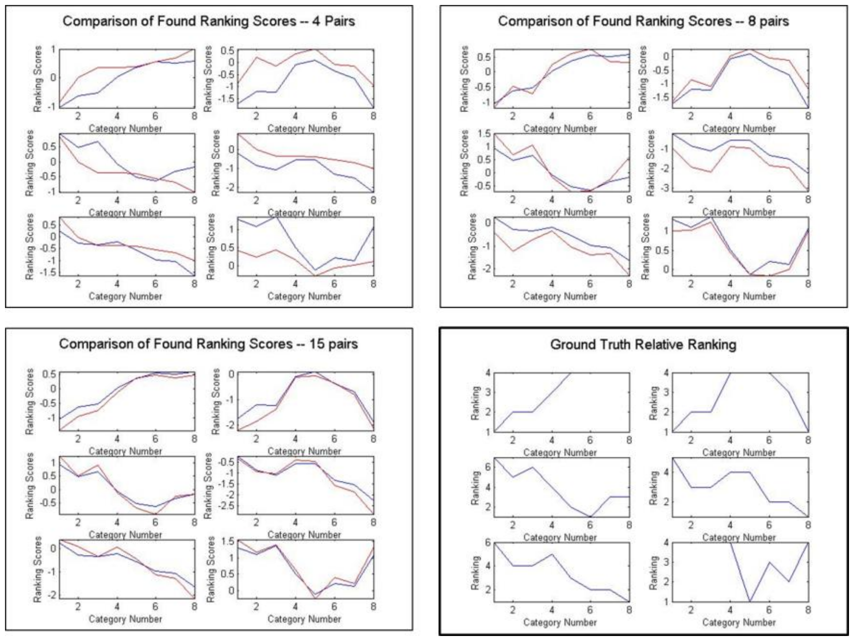


Figure 1. Accuracy of Found Ranking Scores

The shape of the desired curve based on the ground truth ranks of the objects seen in Table 1 can be seen in the plot in the bottom right. Each ranking function then in turn is desired to be similar to this ground truth ranking. We can see that when only using 4 pairs for training of the ranks our values are very different than the learned values given by the author in their data. However, once we use 15 pairs class pairs for learning the image ranks we have very similar plots. This leads me to believe that to have strong classification accuracy it is important to have many pairs of image categories for training. However, the authors state they only used 4 class pairs for training, unless stated otherwise in their experiments. Also, the authors show that more training pairs is not considerably helpful for classification; this contradicts what I have found and presented in this section.

3.5 Generative Probability Framework

After training the ranking functions in the pipeline the final step is to create probability distributions for each class. This is done using the algorithm described in Section 2.5. I believe that I have been able to accurately accomplish this task, and I present results to this effect below. I used the trained ranks given by the authors and automatically generated a random subset of unseen classes, and performed the same tests as the authors described in their training scheme in Section 4 of their paper. The results on the OSR dataset can be seen below in Figure 2. The accuracies that I have found almost exactly match those presented by the authors.

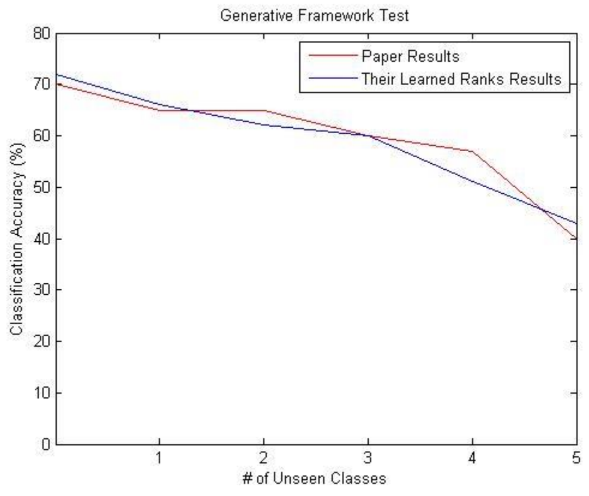


Figure 2. Generative Framework Tests

I have been able to accurately recreate their graph with their given data. Therefore, I believe that my generative framework is operating in the way described by the authors. The only difference between this example, and the final results shown in section 3.6 is in the way that the training data was chosen to train the ranking scores. This leads me to believe that I have not trained the ranking functions used in classification in the same way as the authors did.

3.6 Final Replication Results

In this section the final results from my pipeline are presented and compared to the results that were given within the paper by the authors. Within "Relative Attributes" there were no actual values given for classification accuracies, simply graphical plots of the accuracies for different circumstances. Therefore, I have done my best to estimate what these values were for each plot and the presented results can be seen in the figures below.

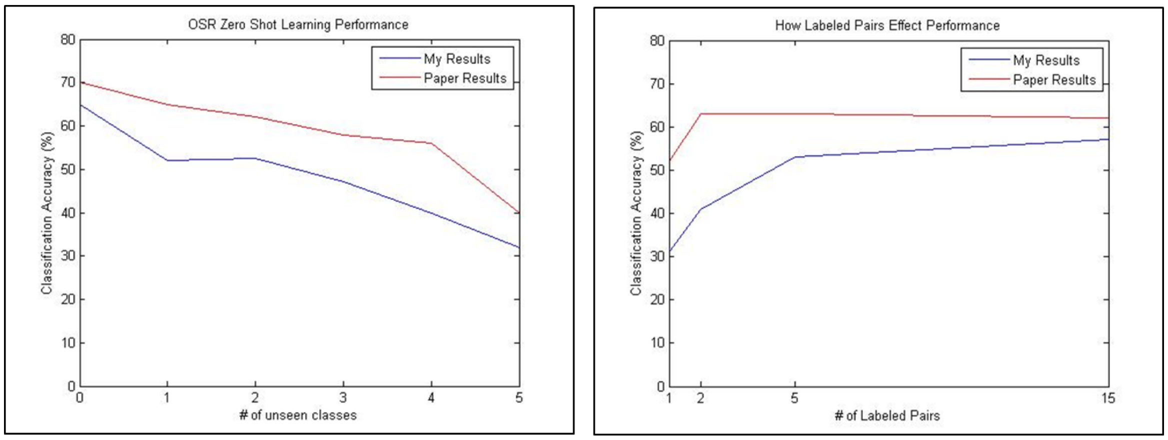


Figure 3. OSR Results – # of Unseen Classes and # of Labeled Pairs

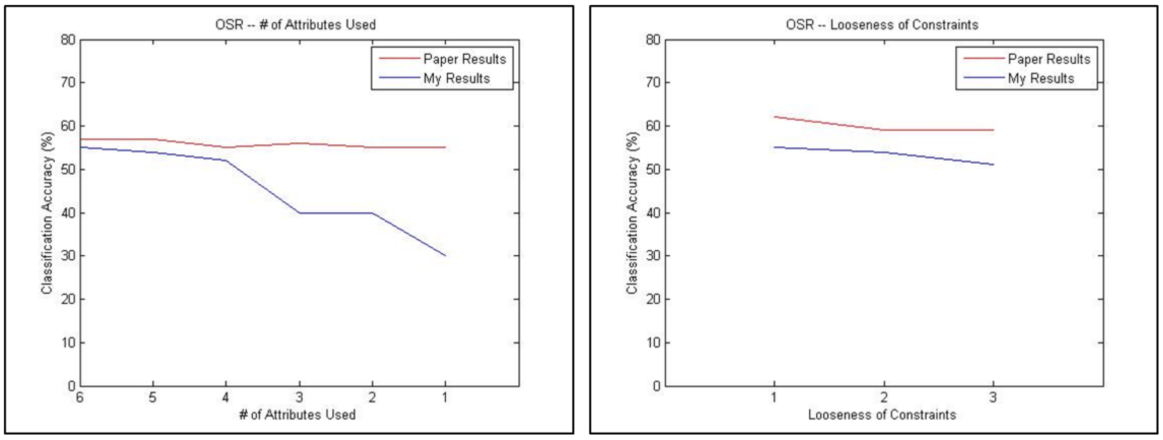


Figure 4. OSR Results -- # of Attributes Used and Looseness of Constraints

Figure 3 shows how closely I was able to replicate the results of the authors on the OSR dataset, and we can see here that the general curvature of the accuracies between their experiments and mine are very similar. However, we can see that my results are slightly lower than their reported performance for both metrics in Figure 3. The most telling plot is the one entitled "How Labeled Pairs Effect Performance", we can see that as the number of pairs used to train the ranking functions increase my classification accuracy increases and is very close to the presented accuracy within the paper. This agrees with my results presented in Figure 1, showing that many training pairs is imperative for learning a ranking function that resembles the ground truth relative ranks of the images. Each of these tests as stated in the paper were done with 6 seen categories, and 2 unseen categories and the ranking functions were trained with 4 category pairs, unless implied by the graph otherwise.

Figure 4 demonstrates the accuracies that my full pipeline has achieved on the other two results presented by the authors. This is the accuracy based on the number of attributes used for the classification task, and based on the looseness of constraints for estimating the unseen class distributions. As we can see above, my results are very similar to those found by the authors when most of the attributes are present in the classification, but then fall off drastically when few attributes are available for classification. I hypothesize that this is because many of the classes have the same rank for category pairs, making it very difficult to separate these pairs when less attributes are available. Finally, we can see that my results have a similar shape to the author's graphs that are presented on the OSR dataset for the looseness constraint. This is a measure of how closely each unseen pair is compared to the seen pairs. When the looseness constraint is 1 the algorithm seen in Section 2.5 is performed on the nearest seen neighbors to each unseen class, when it is 2 it is performed on the second nearest seen neighbors in rank score, etc. Both methods are robust to the looseness constraint that is presented here. However, I believe this is due to the fact that the accuracy on the unseen classes is not presented here, but instead a combined accuracy across seen and unseen classes. The idea of poor classification accuracy on the unseen classes will be discussed again later in Section 3.

My results on the PubFig dataset mirror the results discussed on the OSR dataset. Each plot has a very similar relationship between the authors' results and my results as detailed in my description of the OSR dataset. These results can be seen in Figures 5 and 6 below.

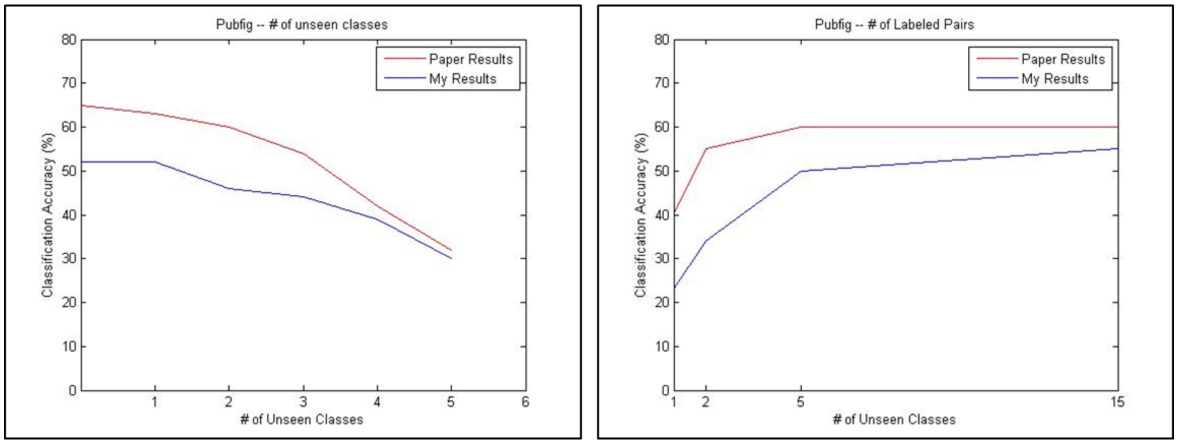
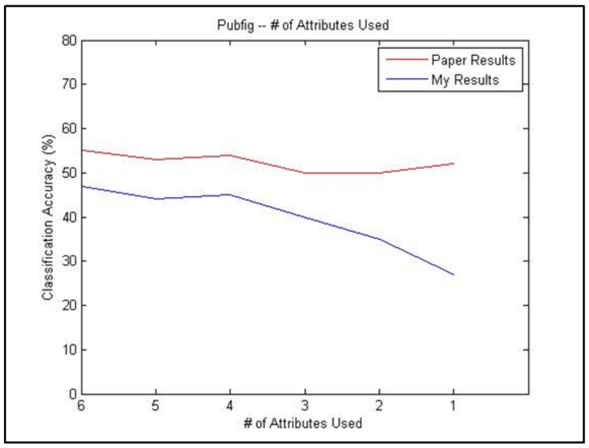


Figure 5. PubFig Results – # of Unseen Classes and # of Labeled Pairs



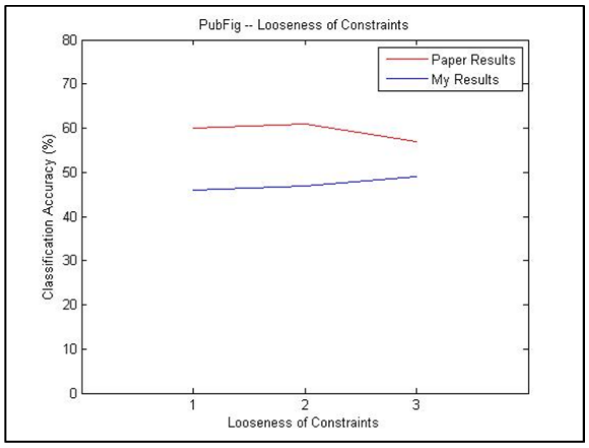


Figure 6. PubFig Results -- # of Attributes Used and Looseness of Constraints

3.7 Discussion of Reproduction Results

As can be seen in the discussion in this section I have been able to replicate many intermediate results that that the author has presented. However, my final results deviate between 10-20% from the stated results in the paper given by the author. This could be due to some outstanding issues, which will be discussed further here.

There is a distinct possibility that I am still not choosing the training data in the same way as the authors did when they created their results. After the email discussion that I had with the authors I changed my training scheme to match what they stated that they used. However, as can be seen in Figure 1, my trained ranking functions only begin to approximate the ground truth relative orderings of each class attribute ranking after 15 labeled category pairs are used for training. However, each of the authors' experiments are performed in the paper using only 4 category pairs. I was not able to achieve ranks that mimicked the ground truth ordering using only 4 category pairs and this I believe is the main issue that as to why my results are consistently lower than those presented by the authors.

Another interesting point that is not shown within the paper is that the authors never actually showcase their accuracy on unseen classes. The main goal of this zero-shot learning framework is to classify unseen categories correctly; however no results actually showcasing this capability are shown. All results show total classification accuracy across all classes, with unseen and seen accuracy calculated together. I contacted one of the authors of the work to inquire into what accuracy they had seen on solely the unseen classes, but was informed that the accuracy was much lower than the accuracy presented on seen categories. This was somewhat upsetting to me given that the ability to classify unseen categories was the reason that I was interested in this project. I was only able to achieve very minimal accuracy with my pipeline for classifying the unseen categories, and very low results using the authors provided ranks. I hypothesize that the authors provided ranks were trained on all categories as well, so it is not a fair comparison between my pipeline's result and attempting to replicate this experiment. I simply present the author's data with my generative framework to show that even if we train the ranks using (presumably) all categories, and then randomly choose some of these categories to be unseen and estimating their probability distributions we still achieve very poor results. These results can be seen below in Figure 7.

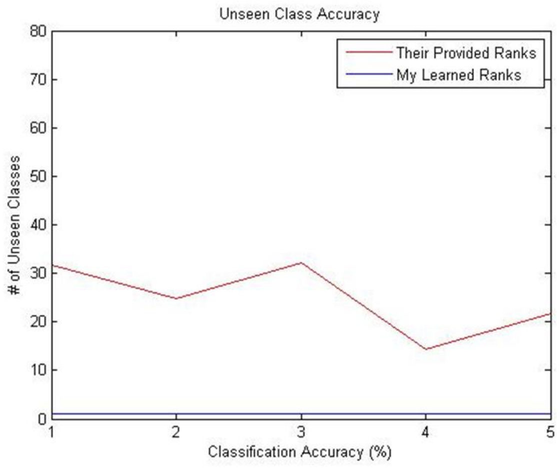


Figure 7. Classification Accuracy on Unseen Classes

I have shown in this section that I have been able to recreate many portions of the authors proposed pipeline for zero-shot learning classification. However, there are two troubling facets to reproducing their results that I have not been able to solve. First, the training scheme that I am currently utilizing provides results across all tests that are 10-20% lower than those presented by the authors. At this point having had correspondences with one of the authors I am sure that I am training the ranking functions in the same way that is described within the experiments section of the paper. Secondly, the performance on unseen categories is much lower than that of the seen categories.

4. Discussion of ICERM Guidelines for "Relative Attributes"

In this section we will discuss the relative merits of this work in relation to the guidelines set forth by the Institute for Computational Research in Mathematics (ICERM) Workshop on Reproducibility in Computational and Experimental Mathematics, which was held on December 10-14, 2012. The relative difficulty of reproducing "Relative Attributes" for each particular guideline set in the workshop will be discussed below in both quantitative and qualitative detail.

4.1 Defining the extent and scope of computational work to be performed

This paper is a highly theoretical work, and the focus of the paper is actually describing the novel algorithms and zero-shot learning framework. The experiments are thus only one section, and they are not discussed in detail. They are used to rationalize why their learning framework performs better than the current previous state of the art for a difficult computer vision problem. The author's state how they perform their experiments and how they designed their algorithm. I was able to implement a first version of their experiment pipeline in 4-5 days. This is slightly longer than the time that I thought it would take, but the algorithms are very well defined in the paper, and therefore I was able to easily implement their basic pipeline. However, getting the same results as the author's proved to be an extremely difficult task and something that I ultimately was not able to accomplish.

4.2 Defining the platforms and software to be utilized

The authors' do not state the platforms and software that they used to perform their computations and experiments. However, they do provide data for the project in the form of .mat files, and also provide a link to their modified Matlab function to perform ranking of the images. This made attempting to recreate their results in Matlab the most straightforward way forward, so this is the path that I took. It would have been nice for the author's to state that they used Matlab and what type of platform their experiments were performed on. I also suggest that in the future the author's discuss the execution time for their experiments.

4.3 Reasonable standards for dataset and software documentation

The author's made accessing their dataset very simple and easy. It was presented on their website and either the actual images or links to the actual images were given. They also provided their extracted features, learned weights, ranks, and the category of each image provided. This was very nice and made understanding their data very simple. They also included a nice README file that detailed what each variable in the .mat file represented. Overall their data was very nicely packaged and I was able to get access to and understand the data sufficiently in a matter of hours. The only complaint that I would have about the author's process of making their data available is that they do not explicitly state the training scheme that they used for learning the provided image rank scores and weights. If this was provided it would have been much easier for me to gauge whether my built ranking and training pipeline was producing identical results to the authors.

4.4 Reasonable standards for persistence of resulting software and dataset preservation and archiving

The authors chose to use Matlab as the format for which to transfer their data, and did not make their code readily available. Matlab is produced by MathWorks, and they work very hard to make sure that each new version of Matlab is backward compatible with older versions of Matlab. Therefore, the author’s will have little problem making sure that there data is usable in the future, and .mat files are widely used within the Electrical Engineering community.

4.5 Reasonable standards for sharing resulting software among reviewers and other researchers

The authors have only provided their augmented version of Chappelle’s Matlab RankSVM implementation publicly, and no other code has been provided. Therefore, their experiments must be rebuilt from scratch, which is a non-trivial issue. As discussed earlier it took me 4-5 days to completely recreate their pipeline, but I have at this point not been able to recreate the results that they presented. It would have been very nice to have their pipeline to test against the work that I have created. At this point, I have not been able to verify their tested results and my thoughts and speculations on why this is can be seen in Section 3.6.

5. Discussion of ICERM Best Practices for "Relative Attributes"

In this section I will discuss how well the authors fulfilled the ICERM best practices for reproducible research as stated in the ICERM Workshop in Reproducibility in Computational and Experimental Mathematics. Each of these best practices was taken into account during the writing of this paper, and I have addressed each of the topics throughout the writing of this paper and the availability of my code and data.

5.1 A precise statement of assertions to be made in the paper

This is completed in the abstract of the paper, and they accurately describe the assertions that are made later in the paper.

5.2 A statement of the computational approach and why it constitutes a rigorous test of the hypothesized assertions.

The author's state how their experimental set up is done in section 4. They discuss why they chose the baselines that they used for comparison. They also discuss their experimental set-up, their training scheme, and the results that they found. I believe that there explanation is not complete. After discussing with the author there was material that I learned as to how they trained their RankSVM implementation with image category pairs that would have been impossible to discover from the paper only. This is a large portion of how the experiments were completed and in my opinion was not described to a necessary level. The authors could have simply included 2 or 3 more sentences discussing their training scheme that would have been very helpful in my reproduction effort. The additional information that I propose should be included can be seen in Section 3.4, and is in the 4th bullet of that section.

5.3 Complete statements of every algorithm employed

The authors do a very good job of discussing their algorithms and the framework that they proposed in this work. There are two main portions of the algorithm that is discussed which is the ranking function, and the generative framework. I was able to accurately recreate both of these functions within my pipeline. A discussion of my ability to recreate the algorithms that they described can be seen in Sections 3.3 and 3.5.

5.4 Salient details of auxiliary software used in the computation

The authors do not discuss what software that they used to perform their experiments, however they provide their data and a portion of their code in Matlab compatible formats. The authors should have included a sentence within Section 4 detailing what software and hardware platform they used to complete these experiments. I assume that all of the work was completed in Matlab, and therefore completed my work using Matlab.

5.5 Salient details of the test environment, including hardware, system software, and the number of processors utilized

No details about the test environment or system hardware are provided by the authors. The authors should have included details of their operating system, computer model, number of processors, and amount of RAM available on the machine.

5.6 Salient details of data reduction and statistical analysis methods

There was no data reduction or statistical analysis methods presented in the paper. The only results presented within this work were the accuracy of classification tasks, and this does not need to be explained.

5.7 Discussion of the adequacy of parameters such as precision level and grid resolution

The authors do not discuss portions of their experimental framework that did have parameters that could be tuned or changed. The author's extracted GIST features from each of the images as stated in the paper, but GIST has some possible parameters that are tuned, and these were not discussed by the authors. It would have been nice to know the exact GIST implementation that the author's used with parameters provided. The author's do state the cost weights that they applied to each training example in the RankSVM implementation which was necessary and very helpful. Therefore, the authors did not include parameters that were needed for replication only once.

5.8 Full Statement of Experimental Results

The author's devote an entire 2 pages within their paper to discuss the results that they have presented. They go through each result in detail and provide rational for what we see in each of the results figures. The only thing that made understanding these results slightly difficult is the absence of the actual numerical accuracies for each test. The author's present graphs that show the accuracies, but the scale is such that we can only estimate the accuracy of each test to around 2% point accuracy. It would be nice to be presented with a table of the actual numerical accuracies that the author's found during their tests.

5.9 Verification and Validation tests performed by the authors

The authors do not discuss any verification and validation tests that they performed. Their results were compared to previous baselines, and I believe that by completing these baselines they were able to validate their test pipeline.

5.10 Availability of computer code, input data and output data, with some reasonable level of documentation

The authors have made their input data readily available with an adequate level of documentation. The raw images are provided, with their extracted features, learned ranking weights, rank scores, which images were used for training, and ground truth image classes.   This data is very well documented, and I was able to easily make use of it. The only suggestion to the authors is to include the training scheme that was used to create this data.

The only portion of code that the authors have made readily available is their modification of Olivier Chappelle's RankSVM implementation. This code is very well commented and easy to use.

However, there is a major issue with the amount of code that they make freely available. Their RankSVM modification is only about six added lines from the original code, and therefore upwards of 95% of their code has not been shared publicly. It would have made my reproduction efforts much easier to have their implementation to reference. I am able to get results that are similar to their results, but only when I use many more training category pairs than they stated should be used in the paper, as can be seen in Section 3.4. Therefore, it would have been very useful to have their implementation of the experiments to check my work against. This would have made replication possible.

5.11 Curation: Where are code and data available? With what expected persistence and longevity? Is there a site for future updates?

The code and data that is described in section 5.10 and throughout this paper can be found at (http://filebox.ece.vt.edu/~parikh/relative.html). This website is well organized and it is easy to find the data that is used in the experiments. It seems to me that this website will remain updated with information well into the future. If the author were to upload their code to the website it would be very nice to have a version controlled implementation. If the authors were to include their code in the website I would suggest using GitHub for versioning control, and for allowing Computer Vision researchers to make enhancements with ease.

5.12 Instructions for completing computational experiments within the paper

The instructions given by the author's and ability to which this paper was reproducible can be seen in Section 3 of this paper. Suggestions for making this work more easily reproduced are also present within this section.

5.13 Terms of use and licensing

Due to the fact that the author's do not provide code except for slightly modified work of another author they have very little licensing attached to their work. The licenses from the original release of the datasets still hold on their dataset, and the original license of the

RankSVM implementation still holds. Therefore, the author's did not need to put a new licensing structure on what they released, and did not do so.

5.14 Avenues of exploration examined throughout the development, including information about negative findings

The authors discuss their findings in detail throughout the results section. They offer some avenues for future work within their conclusion section. They have no negative findings throughout the paper, although I believe that the classification accuracy on unseen classes should have been revealed within the paper.

5.15 Proper Citation of all code and data used, including that generated by the authors.

The authors cited all of the code and data properly, which made it easy to access the data that they used, and the RankSVM implementation that they modified. The authors also did a good job of citing other relevant works within the field.

6. Conclusion

I have presented a complete reproduction of the work "Relative Attributes" presented by Devi Parikh and Kristen Grauman. I was able to reproduce much of their proposed zero-shot learning pipeline, but was ultimately unable to completely reproduce the results that they presented in the experiments section of their paper. This was due to the fact that I was not able to achieve the same quality of ranking functions using 4 labeled category pairs as stated in the training set up of the authors. I believe that in creating the results that were presented in the paper the author’s used a slightly different training scheme than what was presented, due to the fact that I was not able to accurately reproduce the final results that they presented, but was able to accurately recreate each of the intermediate results before and after the ranking function training. I also assessed the reproducibility of the work according the ICERM Guidelines and Best Practices for reproducibility.

References

[1] D. Parikh and K. Grauman, "Relative Attributes," in *International Conference on Computer Vision (ICCV)*, 2011.

[2] A. Olivia and A. Torralba, "a Holistic Representation of the Spatial Envelope," *International Journal of Computer Vision (IJCV)*, 2001.

[3] N. Kumar, A. Berg, P. N. Belhumeur and S. K. Nayar, "Attribute and Smile Classifiers for Face Verification," in *Iternational Conference on Computer Vision (ICCV)*, 2009.

[4] T. Joachims, "Optimizing Search Engines using Clickthrough Data," in *ACM SIGKDD Conference on Knowledge, Discovery, and Data Mining*, 2002.

[5] O. Chappelle, "Training a Support Vector Machine in the Primal," in *Neural Computation*, 2007.