Semantic Symmetry Evaluation

Asha Anoosheh
ashaa@student.ethz.ch

Introduction

Symmetries, in general, can be thought of a pattern - or a subset of one - that can be mapped onto another subset of itself. This can be translational, requiring only translation transformations, rotational, requiring only rotation transformations, or reflectional, requiring only mirroring along an axis. Of course, something can have multiple of these symmetric properties, as well.

Symmetries can be important to many computer-vision related tasks as recognizing patterns in images usually implies something of interest, such as recognizing building facades via translational symmetry of the windows, or identifying disorders in medical imaging if anomalous asymmetries are found.

A special case to observe is within the area of reflective symmetries, particularly when the symmetries are not literal. Current methods for detecting or evaluating reflection symmetries in images compare traditional visual descriptors from the RGB pixel-space of the image, or, more often, just compare the raw pixel values directly. When the symmetry in an image is semantic - found in the human-interpretable meaning of the picture content - rather than direct, these algorithms cannot capture the semantics within these symmetries.

The aim of this project is to try and observe the same, higher-order symmetry that the human visual system can perceive, rather than simply aligning RGB information. In particular the focus is mainly on vertical reflections, though it can be seen later how to use this for rotational symmetries as well. This is useful for cases where aesthetic appeal may want to be evaluated for images, such as in marketing, as humans are known to visually prefer symmetries over non-symmetries. It is also useful for progressing toward a better understanding of how humans understand symmetry so one can more closely replicate it.

The method used here involves Convolutional Neural Networks (CNNs), which possess the complexity required to automatically extract meaningful features in high-dimensional data (i.e. images) and perform tasks such as classification on them - all in a single unit. One of these networks consists of initial convolutional layers, which are trained to find kernels that return high values for certain features they encounter, followed by fully-connected layers to perform a task-specific function such as classification or decoding. If a CNN is trained to classify object(s) in an image, its initial convolutional layers will be trained to detect key features that discriminate specific classes of objects/figures/animals/etc found in an input image. The output of this can be used as features for comparing the symmetry within an image; therefore, if similar classes are found in both halves of an image, regardless of their orientation or color, they can be matched as the same or similar type of content. The output of each convolutional layer kernel is a map slightly smaller than its input (due to convolutional strides that may be present) that for each pixel indicates how strongly it elicits a response from the kernel; additionally, each layer has dozens of kernels, for different types of activations, most corresponding to a different "meaning."

In this work, various types of images are passed through convolutional layers of a network already trained to perform classification, then the activations are used in a number of ways (to be detailed) to calculate a score for the amount of vertical symmetry in each given image.

Related Work

A simple, existing approach to evaluating reflective symmetry is the Symmetry-Distance function found in Kazhdan *et. al* [1]. It uses a simple L2 norm of an image minus its flipped version; their function is one of the ones used in this project (compared against even more.) Another method, from Amirshahi [2], is not meant for evaluating symmetry persay, but rather self-similarity, which is equivalent in our case for a vertical axis. These are detailed further on.

Papers that are very closely related to this include Lettry and Vanhoey [3], which aims to detect repeating lattices for translational symmetry in images using the activations in neural networks, and Brachmann and Redies [4], which was published toward the end of this project, and aims to quantify vertical symmetry in images just like our goal, except using a different measure on the network activations produced. Ideas from both will be incorporated into this paper for evaluation against each other..

Goal

The goal is to find a method of quantifying symmetry in images, in a semantic sense rather than a descriptor-based one. The human perception of semantics is a nuanced one that varies from individual to individual. Humans can describe the ideas present within images and find symmetry in their presentation, whereas current algorithms can only find features to corners to mathematically match and quantify. The high-level knowledge of what the scene contains is crucial to identifying this potentially different form of symmetry.

Methodology

The process starts with feeding an image into a Convolutional Neural Network, pre-trained on image classification. We can extract the feature maps (activations) of the images from the convolutional layers. A convenience of convolutional layers is that they operate spatially and preserve the scale, orientation, and spatial structure of the input image, meaning the output is not uninterpretable as with traditional fully-connected layers. These features that are produced are spikes of activations in the kernels they excite; so a neuron sensitive to dog noses will fire on the pixel region of a dog nose in a photo. And as one goes down the layers toward the later convolutional ones, we see that their kernels operate on smaller versions of the image (as it is downsampled on its way through the network) meaning it is receptive to a much larger field and can capture scene elements, large whole objects, animals, etc.

With these feature maps, we essentially have many images of the same size, on which we are free to experiment with many symmetry-detection/scoring algorithms. Additionally, these can be done at each convolutional layer to observe the different "kinds" of symmetries computed at each level. One would expect the initial layers to capture more color and texture-based symmetry, while the later ones capture the more semantic ones.

The project, as a whole, uses four different algorithms to calculate the symmetry of a given image or activation, as a single scalar value. As they (almost) all only work on 2D matrices, the RGB images are always turned to grayscale while the activations are flattened using one of three different techniques.

The first is the function found in [1], referred to here onwards as Reflection. It does not produce normalized numbers on its own, and therefore only relative ranking of images can be analyzed.

$$score(I) = \left\| \frac{I - flip(I)}{2} \right\|$$

The second is a simple normalized differencing term, referred to as Difference:

$$score(I) = \frac{sum(|I - flip(I)|)}{sum(I)}$$

The next is the self-similarity measure adapted from the PHOG in [2] which will be referred to as Histogram. This is also normalized, producing values in the range [0,1], and it recursively computes subscores until a maximum depth, which are averaged at the end. The original algorithm uses oriented gradients, but for activation maps, this is not meaningful and is replaced by total energy instead.

for $d = 2...maxDepth$:

I_d = *split I into* 2^i *equal sections and sum values within each section*

left, *right* = $leftHalf(I_d)$, $rightHalf(I_d)$

subScore = *histIntersection*(*left* / *sum*(*left*), *right* / *sum*(*right*))

totalScore += *subScore* / *d*

The last one comes from [4], referred here as Dual. It is slightly different from the others in that it uses activations two sets of activations instead of just one. It runs both the unflipped and flipped (vertically) versions of the image through the network independently, and compares the corresponding results. Additionally, this method is a bit odd as convolutional layers are mostly invariant to translational and mirrored inputs, so the resulting activations from the flipped input should, in theory, not differ significantly from simply the flipped version of the unflipped activation.

$$score(I) = \frac{sum(\left|I - flip(I)\right|)}{sum(elemwiseMax(I, flip(I)))}$$

As mentioned before, the activations must be flattened from 3D to 2D somehow, and for this I have experimented with three approaches. The first is to stack the activations on top of each other, essentially losing no data and making it 2D. The second is to sum the activations in the channel dimension, keeping all raw value information, but losing third-dimension position of each. The third is inspired by [3], which instead takes the pixelwise maximum in the channel dimension, losing even more information than the former, but - ideally - reducing the dimension down to the most relevant, unnoisy activations. Though in the original, the maximum is taken across all layers, whereas here the layers are to be kept independent, so maxima are taken only intra-layer.

We have many possible tasks to experiment with here; the following were chosen for analysis and visualization in this report:

1. Activation maps from HybridNet are visually compared next to ones from the VGG-16

2. A single image is scored at each layer (including raw image) to compare scores with activations
3. The relative-score difference between image and activation spaces are compared per algorithm and per flattening method
4. Chosen algorithms from the prior trials are re-compared across HybridNet and VGG-16
5. Images from an evaluation set are ranked in the image space and activation space
6. Selected screenshots from movies are ranked at every layer, including from raw image
7. Randomly-selected album covers ranked in the image and activation spaces
8. Reflective symmetry is evaluated about many rotational axes of the image and the best one is chosen. The largest differences between those gathered from the images are compared with those calculated from activations
9. The evaluation set is once again ranked, but according to a rotational symmetry measure - the sum of reflective symmetries for each discretized axis through its center.

Data

The images to test on were hand-picked, obtained from various, license-free sources on the web. There are three image sets used for experimentation. The main one consists of carefully-selected images containing different types of symmetries, as well as in both the semantic and literal senses. Another consists of stills from during notable movies, and the last is a collection of randomly-gathered album covers, due to their inherently-square shape.

For the pre-trained CNN, I use two for comparison. The ideal network for this task is one that is trained on both objects and scenes from the ImageNet and MIT Places datasets, respectively. Fortunately there happens to be one for this, known as the MIT HybridNet [5]. Another more recent and advanced model is known as VGG [6], which theoretically should produce better, more meaningful activation maps, but takes much longer to run due to very high gpu-memory requirements and it only trained on ImageNet. I use the VGG-16 network for comparing against the HybridNet in some trials. These are used with the Caffe framework.

Changes and challenges along the way

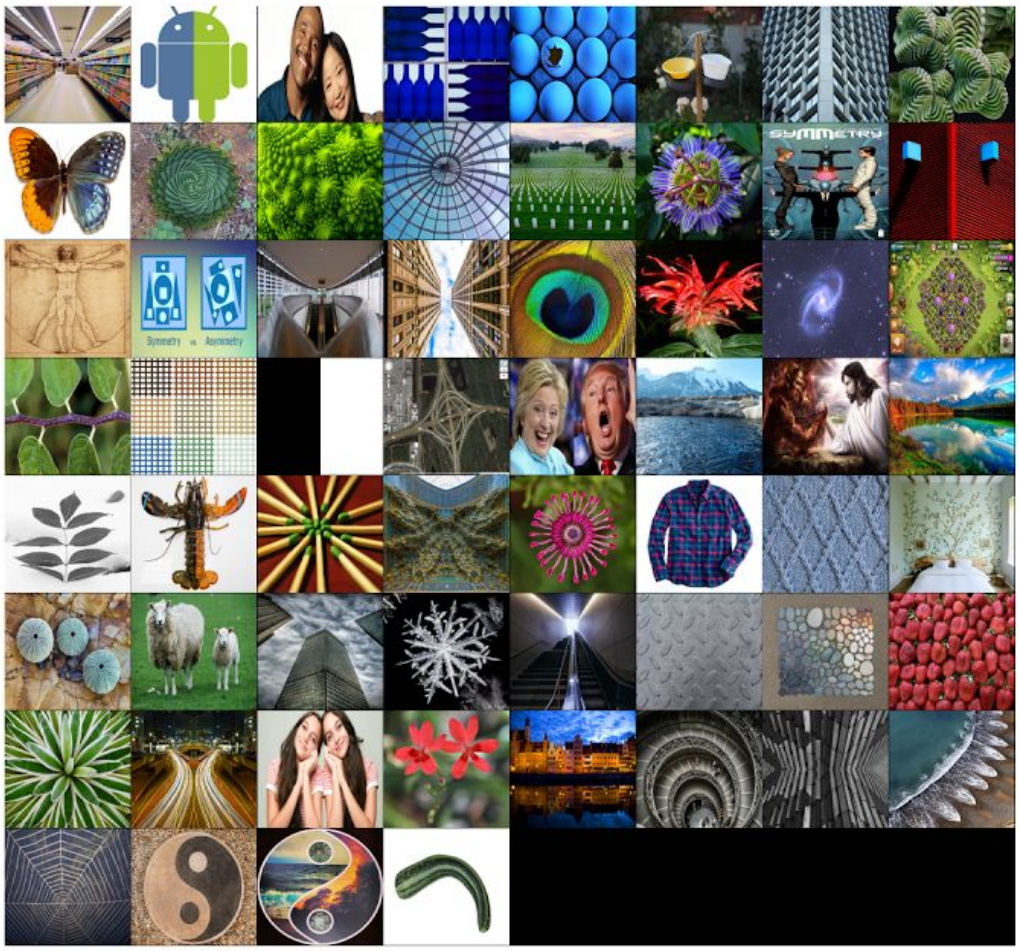
The main challenges included finding a suitable algorithm for quantifying symmetry such that the score would not be (too heavily) influenced by the raw amount of pixel values. This included the dilemma of using original size images versus resizing them to the size the network was trained on. On one hand, since no size-dependent layers are used in the networks, the input images could be of any size and not be distorted by resizing. On the other hand, having a

consistent number of pixels among images ensures no issues related to the calculated score being influenced by it. Additionally, since the network was originally trained on images of 227x227, the filters are accustomed to features of its own scale; therefore, I eventually stuck with the latter.

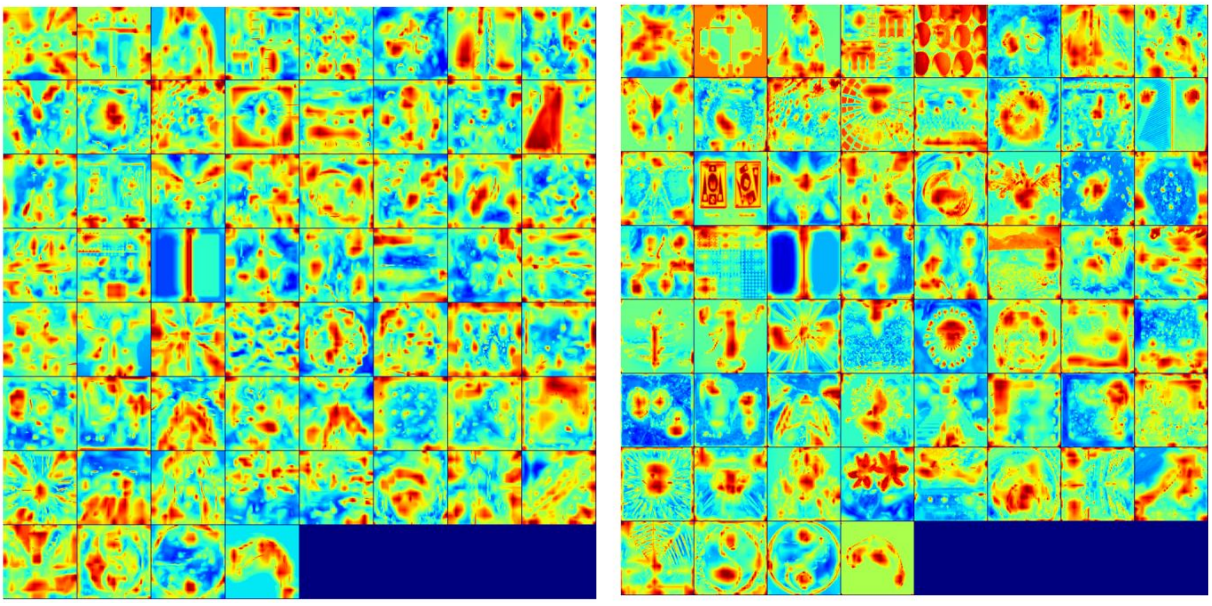
In addition to multiple architectural redesigns in the entire pipeline to alleviate limited memory constraints, additions were also made along the way to the symmetry-quantification algorithms. One problem with a yet-unclear answer is how to combine the many activation maps from each layer, whether they should be combined across layers even though their sizes do not match, or if they should be combined at all. In Louis *et. al*, there is one activation per image regardless of layer count, as all activation maps are resized to the same size, then the maximum is taken across the pixels. Since I want to keep layers separate in order to evaluate their scores, I used this idea but only taking the maximum of activations per layer, in addition to my initial ideas of using the pixelwise sum of the activations, and comparing both to the raw activations (stacked).

Another problem was how to choose a "final" score among each layers' scores to use for evaluation. Even though each layer still has its own score for analysis purposes, it's not convenient to have to visualize and analyze 5 to 15 score lists each time, so it's useful to combine them in a meaningful way to have a single value per image. This also incorporates the tying together of different meaning of each layer's responses (from image to semantic), rather than having a purely semantic or purely image-based score. Initially, the median of the score values was taken, yet it almost always ended up being the middle-most layer's values for algorithms that did not produce an inherently-normalized score, as the amount of pixels often determined the magnitude range of the scores. Instead I opted for a geometric mean over all layers, which handles the potential case of large variations in magnitude among values.

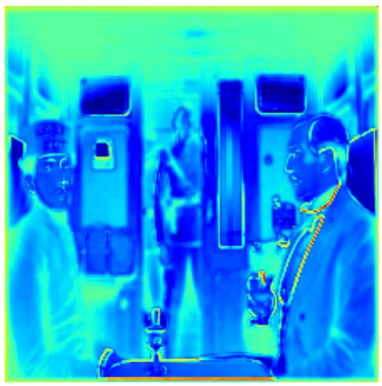
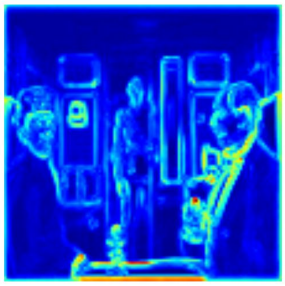
Results

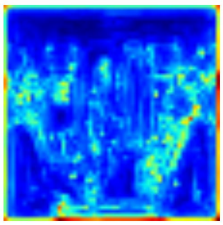
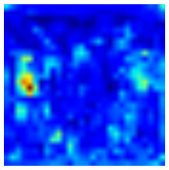
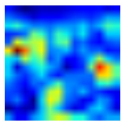
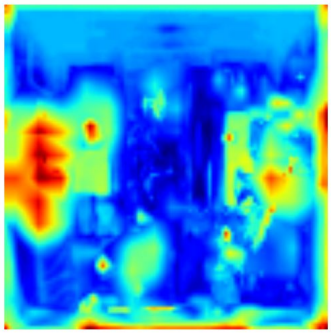


Activations of above test images from HybridNet (left) and VGG-16 (right)



Comparison of symmetry scores at each layer compared to raw image (using histogram)

	Image	**Score**
Data		0.400
C1		0.878
C2		0.920

C3		0.871
C4		0.642
C5		0.442
C-max		0.807

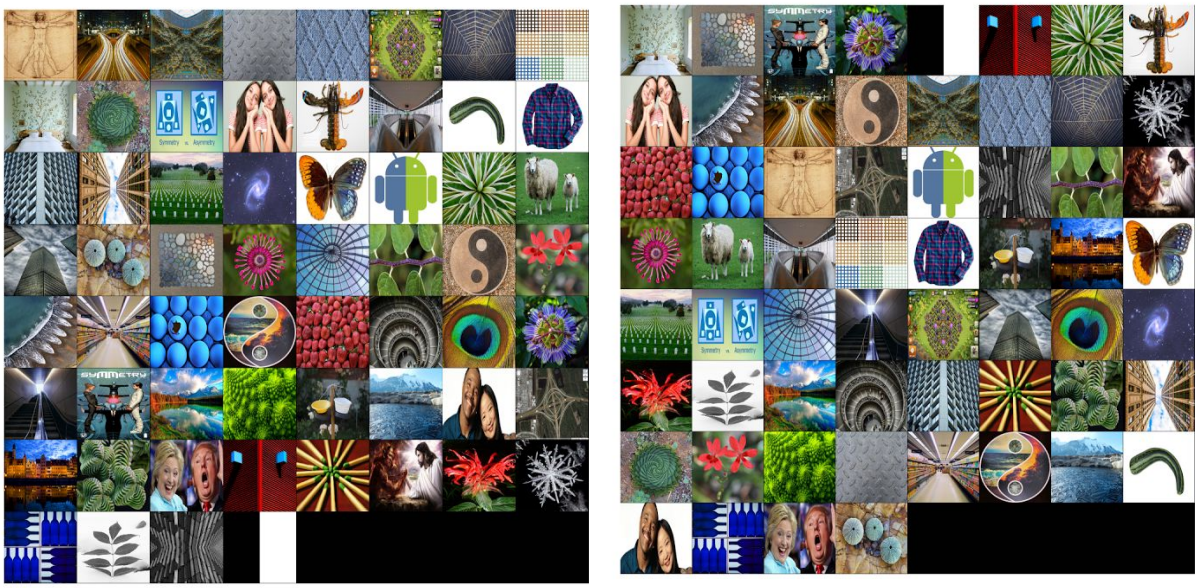
Evaluation set images: Sum of relative-score differences between image scores and layer scores

	Difference	Reflection	Histogram
Stack	8.63	3.31	5.24
Sum	3.94	3.01	5.67
Max	3.83	2.89	11.56

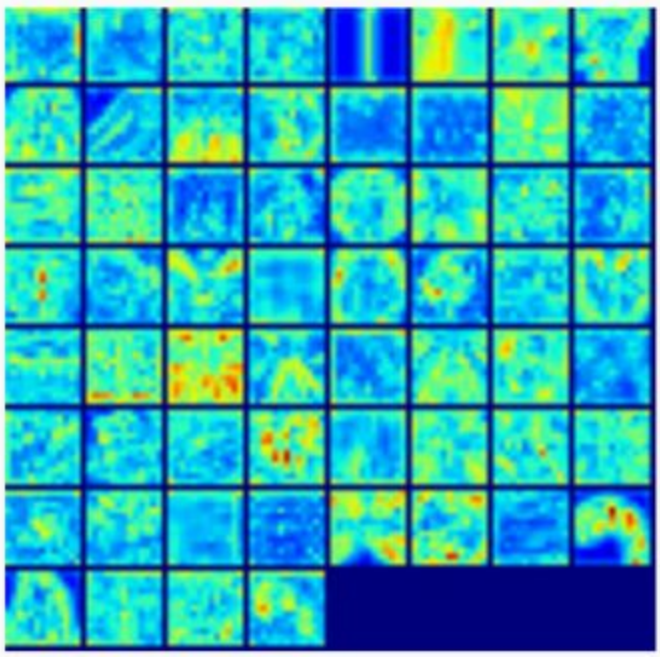
HybridNet vs VGG: Sum of relative-score differences between image scores and layer scores

	Hybrid-Net	VGG-16
Diff-Max	3.83	2.99
Refl-Max	2.89	2.11
Hist-Stack	5.24	8.22
Dual(-Stack)	31.11	*<lack of memory>*

Ranking of test images from RGB images [left] and using conv-layers [right] (using histogram)

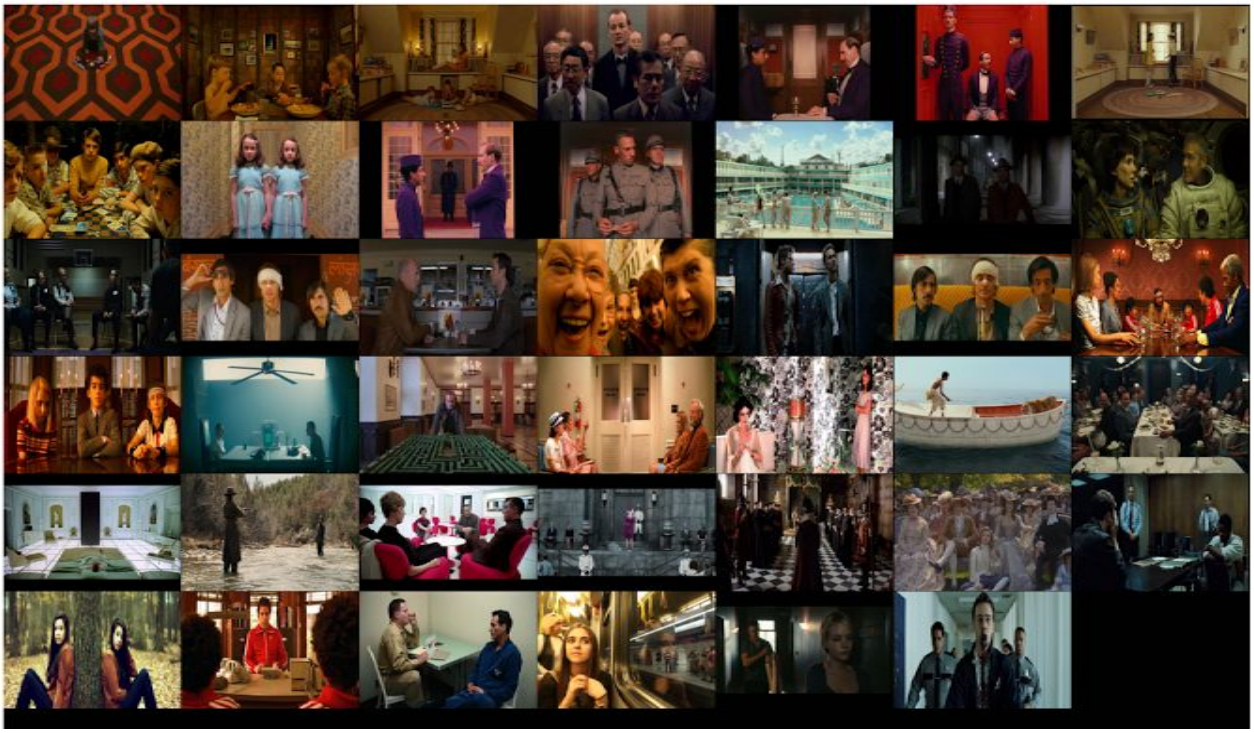


Corresponding maximal activations to the above conv-layer order



Ranking of Movie stills from RGB images (using differencing)

Ranking of Movie stills of Conv-1 activations (using differencing)

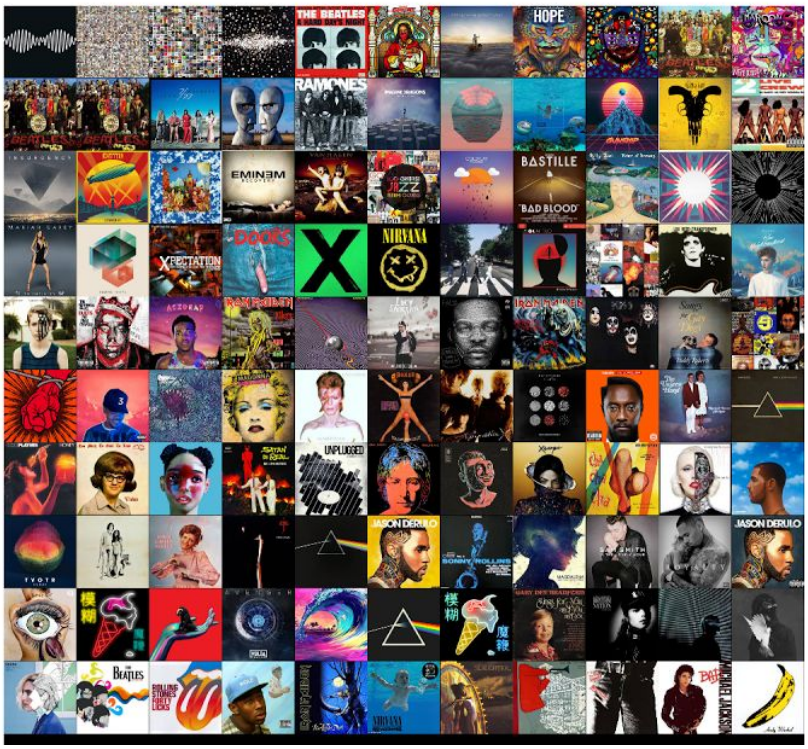


Ranking of Movie stills of Conv-3 activations (using differencing)

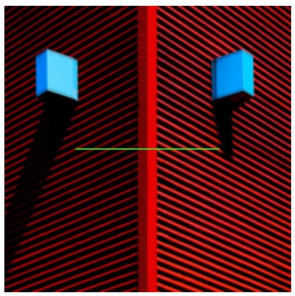
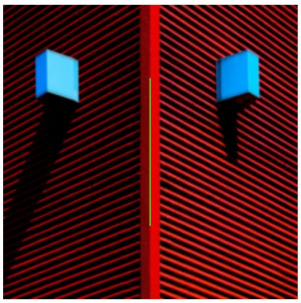
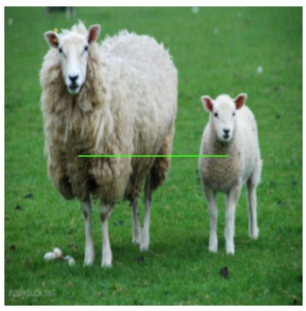
Ranking of Movie stills of Conv-5 activations (using differencing)

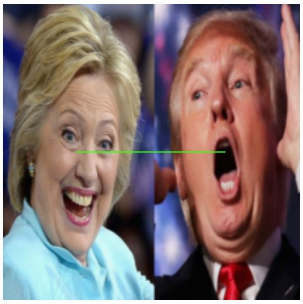
Album Covers ranked from RGB images (using Histogram)

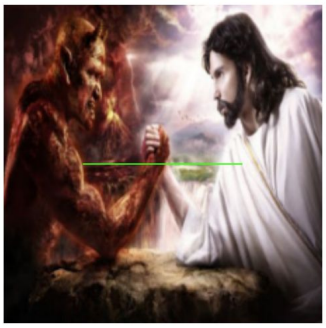
Album Covers ranked from Activations (using Histogram)



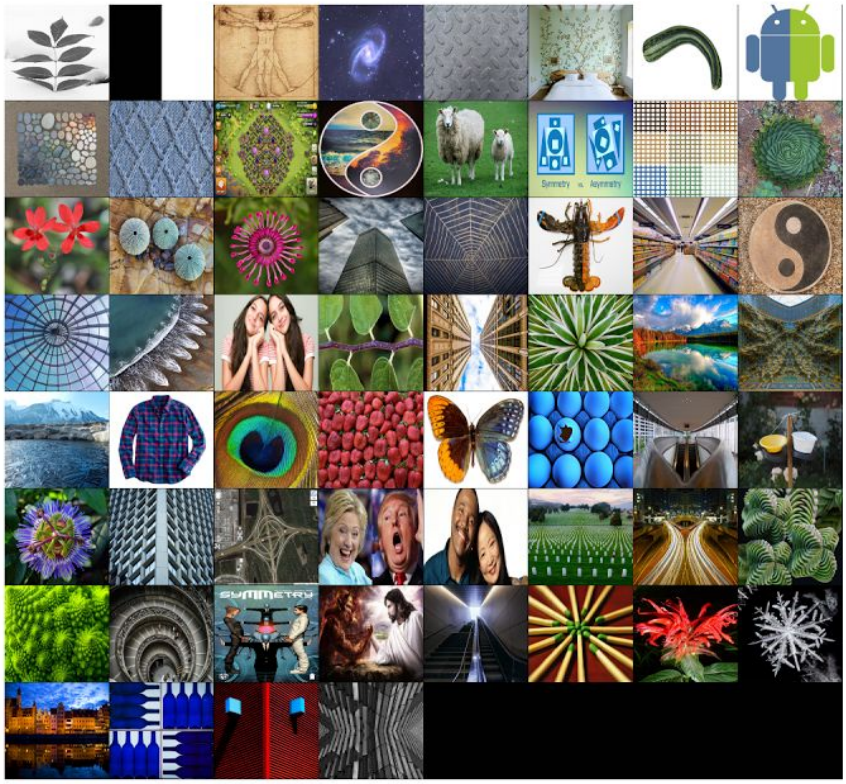
Rotation Axes - best found (using Reflection)

Using RGB image	Using Activations (VGG)
	
	
	
	

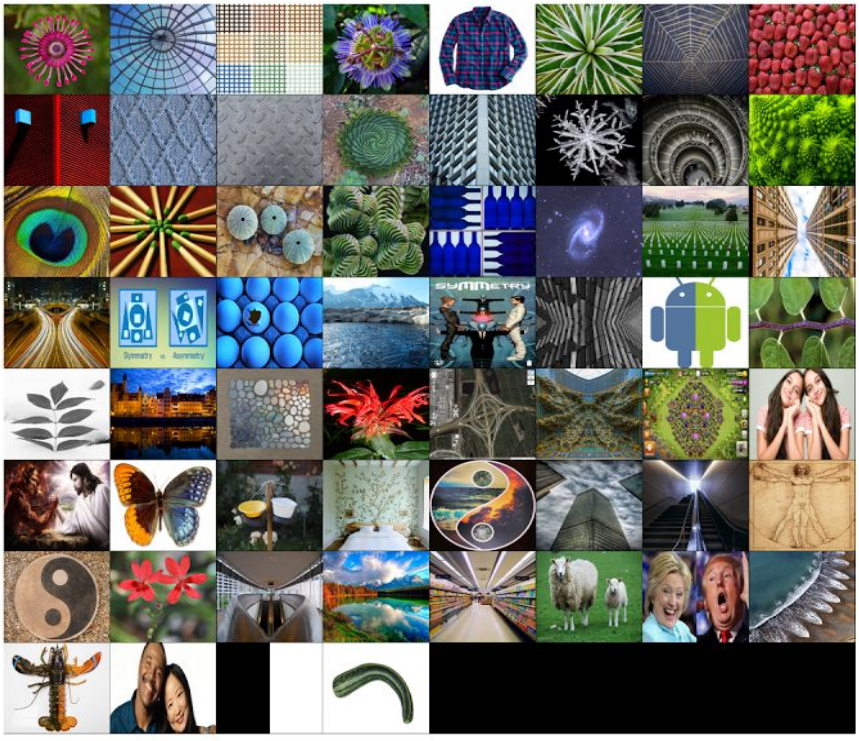




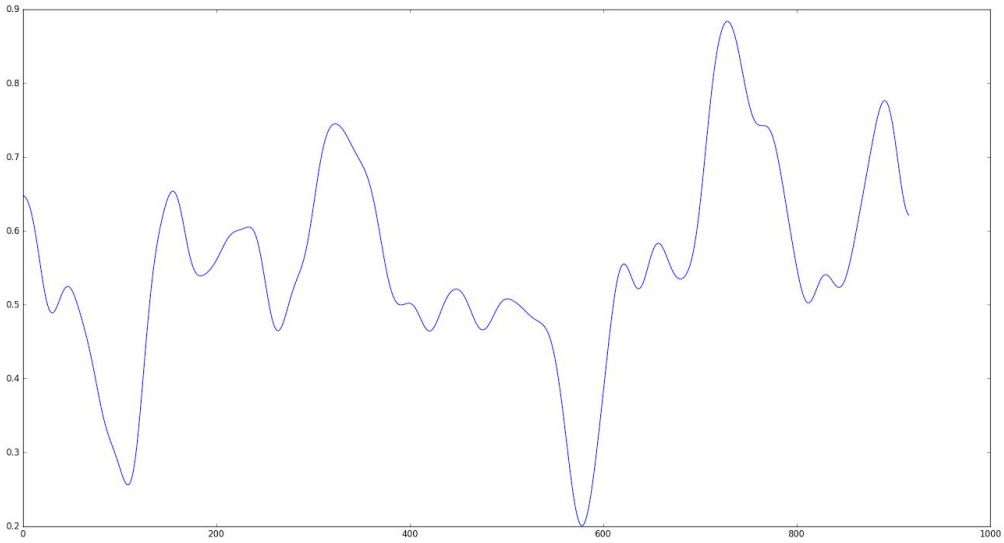
Ranking by Rotational Symmetry from RGB image (using differencing)



Ranking by Rotational Symmetry from Conv layers (using differencing)



Symmetry of Movie frames over time (The Shining)



Link to animated GIF (The Shining)
http://i.imgur.com/YTLnO1v.gifv

Conclusion

We find that vertically-reflective semantic symmetry is not only much better detected using neural-net activations than the original image (with a few controversial-yet-subjective standout cases), but it also significantly improves the quantification of rotational symmetry - regardless of semantics.

Comparing semantics is always a solutionless task, as it involves assigning a notion of correctness to subjective ideas, unless many cases are specifically and strongly defined beforehand. The configurations for each visualization are the most semantically-varying rankings or calculations between the data space and feature space evaluations. From the range of different symmetries found in the images, although they are subjective and the amount of semantic symmetry varies from person to person, a trend can be noticed that the activation-space calculations put meaning ahead of simply superficial colors and patterns. It should be noted that in some cases, it will worsen some results, placing semantically symmetric images lower than the image-domain did; though these are usually a result of contrasting with my personal expectation rather than being wrong in any sort of ground-truth way.

With the task of finding the best reflective axis, the results that matched closely enough between both methods are not shown, but rather the ones with a stark, semantic difference. The

image-based method simply prefers like-colored areas, while the semantic method bases its judgement off the content of the images, identifying two different-looking but meaningfully-similar items on both sides of some of the images.

When analyzing rotational symmetry, the image domain fails almost completely, matching large amounts of pixel values instead of useful features. Images that should, even with subjective bias, not be ranked ahead of others are. Meanwhile the semantic method does this correctly, resulting in an unarguably sizable performance gain.

Now we contrast the end results of the four different symmetry-distance functions. The Difference and Dual function are practically the same operation, except one is 2D while the other is 3D, respectively; the difference in outputs comes down to how the 2D one is flattened. The Reflection function is harder to work with since it is not bounded; the denominator does not restrict the values. Instead it must be normalized, so outputs of this function can only be compared relative to one another - hence ranking. With the Histogram function, each side is normalized before intersection, resulting in values between 0 and 1 automatically. But this means images with different intensity values on both halves will be seen as equals, explaining why the half-black, half-white image in the test set is ranked highly by this algorithm.

When comparing the networks, it is clear the activation features from the VGG network are much more precise and detailed than the HybridNet, yet their drawback is that it maintains spatial information a bit too much, making comparisons dependent on more exact lining-up of pixel values. So while HybridNet's activations have this desirable blurry-blob characteristic, they tend to be sensitive to areas that sometimes are irrelevant to the desired semantic extraction and not as sensitive to the areas we want. This tradeoff is considered in the specific task at hand - whether specific spatial layout should be punished more harshly or not.

Notice that visual results from the Dual method are excluded here; this is because their ranking on the evaluation set usually gave matches too dependant on low-level visual features and, from inspection, seems to ignore higher-level semantics. The authors of the paper this function was derived from may have achieved more desirable results using a different one.

Future Work

Some other interesting directions to take this work include finding other symmetries, namely translational and rotational, within an image using the neural network activations - and preferably not by using the reflective symmetry repeatedly as was done here to find "rotational"

symmetry. Existing algorithms made to run on regular images can be applied to the activations directly, or with slight modification, instead.

Another interesting idea would be to train a network to learn to score semantic symmetry on its own. To acquire training data, the current algorithms here could be used to generate scores for a decent number of pictures, then those could be used to fine-tune an existing network for this task. Training one from scratch given many labels would simply replicate the function used to generate the scores, unless regularization is done very cleverly. With fine-tuning, the regularization is embedded already, and the smaller training set ensures it does not fit to the function as easily, but rather learns to generalize the idea for other types of semantics it encounters.

Additionally, since the start of this project, multiple new pre-trained CNNs have been uploaded to the Caffe Model Zoo, including some trained on human faces, but unfortunately none with a mixture of faces and other common pictures. Should one be trained as such, it may perform better for images with semantic symmetry involving humans, as the current networks should, in theory, not be very sensitive to human faces.

Works Cited

[1] Michael Kazhdan, Bernard Chazelle, David Dobkin, Adam Finkelstein, & Thomas Funkhouser; *A Reflective Symmetry Descriptor;* ECCV 2002
https://www.cs.princeton.edu/~funk/eccv02.pdf

[2] Seyed Ali Amirshahi, Michael Koch, Joachim Denzler, & Christoph Redies; *PHOG analysis of self-similarity in aesthetic images*; SPIE Feb. 2012
https://www.researchgate.net/publication/258712464_PHOG_analysis_of_self-similarity_in_esthetic_images

[3] Louis Lettry & Kenneth Vanhoey; *Detecting Repetitions using Deep Features*; (in-progress)

[4] Anselm Brachmann and Christoph Redies; *Using Convolutional Neural Network Filters to Measure Left-Right Mirror Symmetry in Images*; Symmetry Dec. 2016
http://www.mdpi.com/2073-8994/8/12/144

[5] Hybrid CNN
https://github.com/BVLC/caffe/wiki/Model-Zoo

[6] Karen Simonyan, Andrew Zisserman; *Very Deep Convolutional Networks for Large-Scale Image Recognition*; ILSVRC 2014
https://arxiv.org/pdf/1409.1556v6.pdf