Paper 1 -

BAM! The Behance Artistic Media Dataset for Recognition Beyond Photography (2017 ICCV)

65 Million Images in Dataset - From Behance Website

High Level Image Categorization - Object Categories, Artistic Media, Emotions

Media Attributes - Computer Graphics, Comics, Oil Painting, Pen ink, Pencil Sketches, Vector art, Water Color.

Emotion Attributes - Calm/Peaceful, Happy/Cheerful, Sad/Gloomy, Scary/Fearful

Object Attributes - Bicycles, Birds, Buildings, Cars, Cats, Dogs, Flowers, People, Trees.

Applications -

- Detecting Objects in ArtWork. - Fusion of their Model and ImageNet scores performed better than their model alone.
- Style aware image search, how this dataset can be used to search for images based on their content, media or emotion.

Their Models are not available. Will have to either implement them on my own or look for alternatives.

Paper 2 -

Recognizing Image Style (2013)

Define different types of Image style and collect a dataset with style labels.

NOTE - Most previous work in aesthetic style analysis has used hand-tuned features, such as color histograms. We find that deep convolutional neural network (CNN) features perform best for the task.

This is surprising for several reasons:

- These features were trained on object class categories (ImageNet), and many styles appear to be primarily about color choices, yet the CNN features handily beat color histogram features.
- This leads to one conclusion of our work: mid-level features derived from object datasets are generic for style recognition, and superior to hand-tuned features.

DIfferent aspects of visual style are captured -

- photographic techniques (“Macro,” “HDR”)
- composition styles (“Minimal,” “Geometric”)
- moods (“Serene,” “Melancholy”)
- genres (“Vintage,” “Romantic,” “Horror”)
- types of scenes (“Hazy,” “Sunny”).

These styles are not mutually exclusive, and represent different attributes of style. We also gather a large dataset of visual art (mostly paintings) annotated with art historical style labels, ranging from Renaissance to modern art.

- Optical techniques:Macro, Bokeh, Depth-of-Field, Long Exposure, HDR
- Atmosphere: Hazy, Sunny
- Mood: Serene, Melancholy, Ethereal
- Composition styles: Minimal, Geometric, Detailed, Texture
- Color: Pastel, Bright
- Genre: Noir, Vintage, Romantic, Horror

4000 positive examples for each label, for a total of 80,000 images.

Learning -
Stochastic Gradient Descent with adaptive subgradient.
Our setup is of multi-class classification; we use the One vs. All reduction to binary classifiers.

Image Features Used -
L*a*b color histogram
GIST
Graph-Based Visual Saliency
Meta-Class Binary Features
Deep Convolutional Neural Network
Content Classifiers.

Paper 3 -
What makes Paris Look like Paris? (2012)
Most distinctive parts of a geo-spatial area - cities.
Find patches which are **frequently occurring** and **geographically informative**.

10,000 images per location. (12 locations and size 936x537)

Discovering geo-informative elements
The database has 2 parts -

1. Positive set containing images from the location under examination.
2. Negative Set containing images from the rest of the locations

K-Means clustering on image patches (described by SIFT features) - bad.

Method -

1. Compute the NN of each candidate - reject candidates with too many neighbours in the negative set.
2. Then build clusters by applying iterative discriminative learning to each surviving candidate.

Each patch represented by a **HOG + color descriptor.**

Algorithm-

DATA

1. 25000 high contrast patches randomly selected (from positive set) serve as candidates for seeding the clusters.
2. Each patch represented by a **HOG + color descriptor.**
3. Initial geo-informativeness of each patch is estimated by finding the top 20 NN patches in full dataset (positive and negative).
4. Keep candidate patches that have the highest proportion of their NN in the positive set.
5. (Reject near duplicate patches - measured by spatial overlap of > 30% between any 5 of their top 50 NN)

TRAINING - (After getting 1000 images and their nearest neighbours)

6. Training a linear SVM detector for each visual element in an iterative manner.
    a. This produces a weight vector which corresponds to a new, per-element similarity measure that aims to be geo-discriminative.
7. SVM works as follows -
    a. For each sample, use top k NN in the positive set as positive examples (and all negative-set patches as negative examples) (k = 5)
    b. Do this iteratively. Use now top k detections from previous round. Top detections will become better with every round.
    c. For further improvements and to prevent overfitting - apply cross-validation by dividing both the positive and the negative parts of the dataset into l equally-sized subsets (l = 3)
    d. Now at each iteration, apply the detectors trained on the previous round to a new, unseen subset of data to select the top k detections for retraining.
    e. After 3 iterations, rank the resulting detectors based on their accuracy: percentage of top 50 firings that are in the positive dataset.

Implementation Details -

- Square patches, ranging from 80x80 to height of image size.
- Each patch is represented by standard HOG (8x8x31), plus a 8x8 color image in L*a*b space (a and b only). Final resulting feature has 8x8x33 = 2112 dimensions.
- SVM - soft-margin = 0.1

Using the Paris Paper Pipeline on Our work -

What Makes Paris Look Like Paris? -
Their flow -
City -> Images -> Elements
Our flow -
Genre -> Poster

Reverse flow -
Poster Elements -> Posters -> Genres

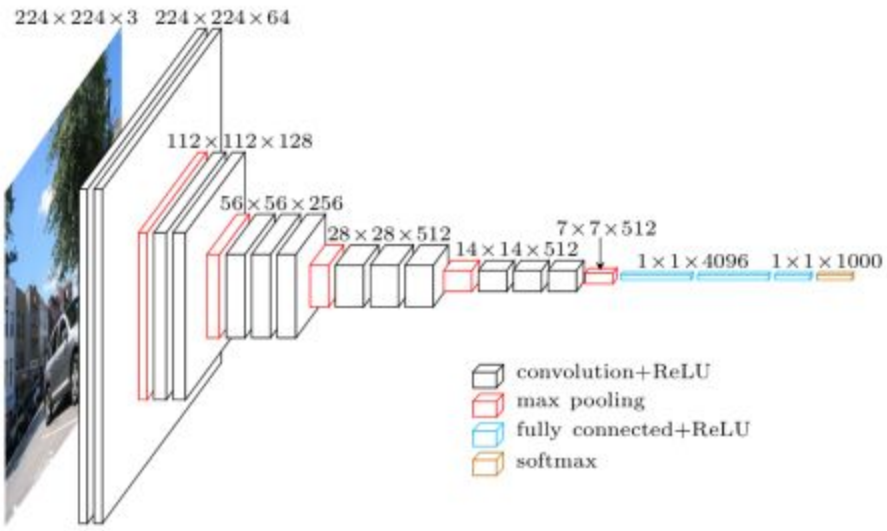
Basically we will say that there are some poster elements which are key elements to a particular genre or a set of genres.
These possible Poster Elements are -

1. **Text Blocks** (Text Detection Pipeline)
2. **Object Detection Blocks** - (This can be done by YOLO objected detector or Faster R-CNN pre-trained object detection networks)

To begin with we can take following genre tuples which appear to be **visually distinct** and are mostly **disjoint** -

1. Adventure, Animation, Comedy
2. Action, Crime, Thriller
3. Biography, Drama, History
4. Horror, Mystery, Thriller

APPROACH 1 -



1. For each image - select patches 80x80 (hierarchically) and pass them through the network to get pre-trained feature vector FC7 (4096 dimensional)
    a. Then run the Paper Pipeline to find discriminatory patches in the above genre tuples.
2. For each image work in the filter space of the VGG pre-trained network. Pick Conv layer 4_3 where each filter size is 28x28 and there are 512 filters. We get 512 filter spaces.
    a. Now try to do clustering in this space to find which filter see what.
3. Now reverse map a patch in the filter space - say 4x4 patch in the Conv Layer filter is mapped to some original image patch. Now for each patch in the original image we get a 4x4x512 (8192) dimensional feature vector.
    a. Use these patch as your discriminatory patches.

APPROACH 2 -

Instead of using the patches from the pre-trained network filter space. Try the following
Find poster elements which are key elements to a particular genre or a set of genres.
These poster elements can be -

1. Text Blocks
2. Object Detection Blocks

Represent each element block by some feature representation, and apply the Paper pipeline.

Task or Experiment to perform -
Break an image into a kind of jigsaw puzzle and show people parts of images from 2 genres. See if they are able to discriminate between the 2 and identify which image is closer to a particular genre.

STYLE TRANSFER PAPERS-

PAPER 4
Image Style Transfer Using Convolutional Neural Networks

Image representations derived from Convolutional Neural Networks optimised for object recognition, which make high level image information explicit.

GIven - Content Image and Style Image. We need to keep the content of the former image and style of the latter.
This is achieved by the following way -

Content Representation and Loss

Construct images whose feature maps at a chosen convolution layer match the corresponding feature maps of a given content image. We expect the two images to contain the same content but not necessarily the same texture and style.
Given a chosen content layer **I**, the content loss is defined as the Mean Squared Error between the feature map **F** of our content image **C** and the feature map **P** of our generated image **Y**.

$$\mathcal{L}_{content} = \frac{1}{2}\sum_{i,j}(F_{ij}^{l} - P_{ij}^{l})^2$$

When this content-loss is minimized, it means that

1. The mixed-image has feature activation in the given layers that are very similar to the activation of the content-image.
2. Depending on which layers we select, this should transfer the contours from the content-image to the mixed-image.

Style Representation and Loss

We will do something similar for the style-layers, but now we want to measure which features in the style-layers activate simultaneously for the style-image, and then copy this activation-pattern to the mixed-image.
One way of doing this, is to calculate the **Gram-matrix**(a matrix comprising of correlated features) for the tensors output by the style-layers. The Gram-matrix is essentially just a matrix of dot-products for the vectors of the feature activations of a style-layer.
If an entry in the Gram-matrix has a value close to zero then it means the two features in the given layer do not activate simultaneously for the given style-image. And vice versa, if an entry in the Gram-matrix has a large value, then it means the two features do activate simultaneously for the given style-image. We will then try and create a mixed-image that replicates this activation pattern of the style-image.
If the feature map is a matrix **F**, then each entry in the Gram matrix **G** can be given by:

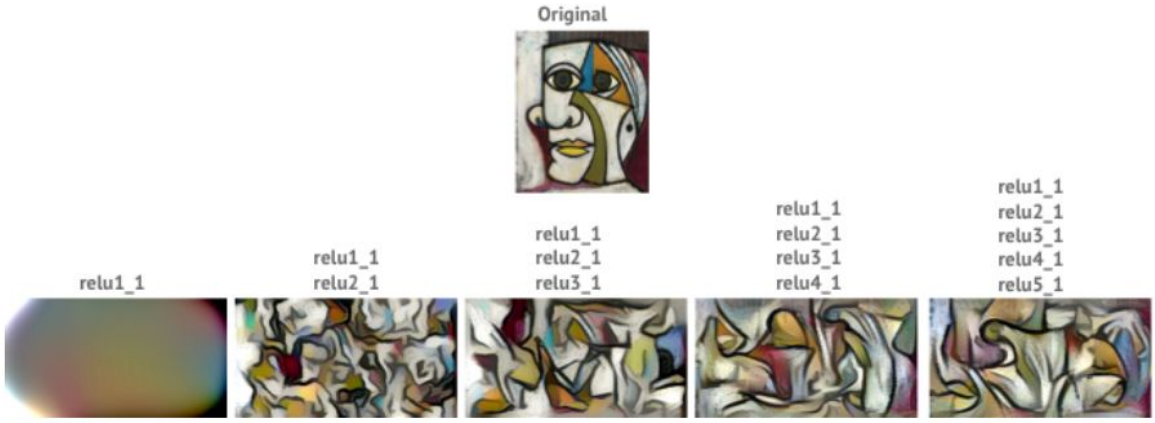
$$G_{ij} = \sum_k F_{ik} F_{jk}$$

The loss function for style is quite similar to out content loss, except that we calculate the Mean Squared Error for the Gram-matrices instead of the raw tensor-outputs from the layers.

$$\mathcal{L}_{style} = \frac{1}{2} \sum_{l=0}^{L} (G_{ij}^l - A_{ij}^l)^2$$

As with the content representation, if we had two images whose feature maps at a given layer produced the same Gram matrix we would expect both images to have the same style, but not necessarily the same content. Applying this to early layers in the network would capture some of the finer textures contained within the image whereas applying this to deeper layers would capture more higher-level elements of the image's style. Gatys et. al found that the best results

were achieved by taking a combination of shallow and deep layers as the style representation for an image.



The total loss can then be written as a weighted sum of the both the style and content losses.

$$\mathcal{L}_{total} = \alpha \mathcal{L}_{content} + \beta \mathcal{L}_{style}$$

Use SGD to minimize this loss. Take a white noise image as target image and get it closer to the combination.

1. **Reference :** https://medium.com/data-science-group-iitr/artistic-style-transfer-with-convolutional-neural-network-7ce2476039fd
2. https://blog.slavv.com/picking-an-optimizer-for-style-transfer-86e7b8cba84b

Some notes from the paper which are critical and which need inspection

- We used the feature space provided by a normalised version of the 16 convolutional and 5 pooling layers of the 19-layer VGG network. We normalized the network by scaling the weights such that the mean activation of each convolutional filter over images and positions is equal to one.
- For image synthesis we found that replacing the maximum pooling operation by average pooling yields slightly more appealing results, which is why the images shown were generated with average pooling.