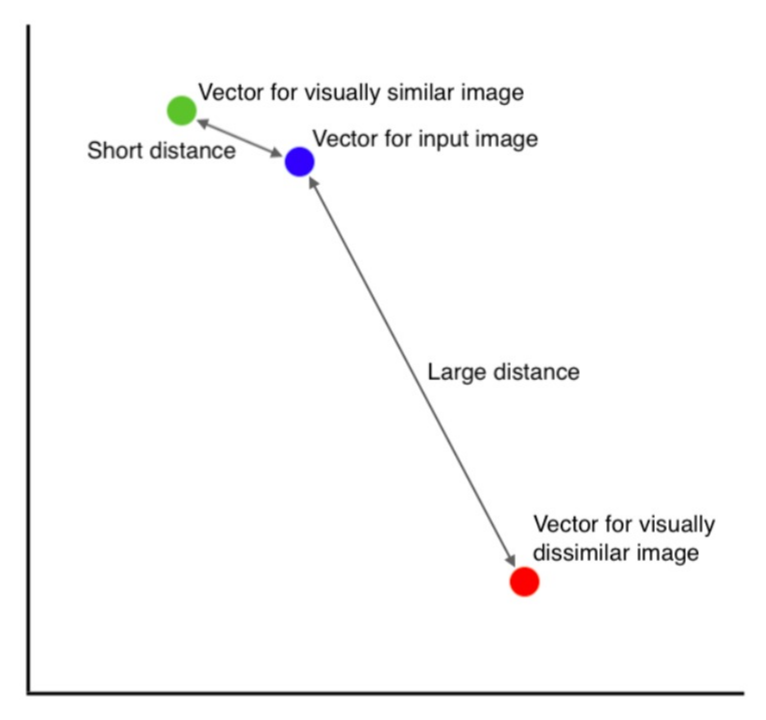
Developing Art Style Embeddings for Visual Similarity Comparison of Artworks

This task was undertaken as part of a proof of concept for Look and Learn, an online library of high definition historical pictures.

Grant Holtes Sep 1, 2019 · 6 min read

The aim of this exercise is to find a function that transforms images into embedding vectors where the euclidean distance between vectors represents how visually similar the images are. This allows a nearest neighbors search on one image's embedding to return images that are visually similar, empowering image recommendation and clustering.



This task is complicated by the images being illustrations rather than photographs. While photographs are true representation of the subject, and hence similar subjects and composition would yield visually similar images, illustrations include the extra dimension of style.

> *Style: "a way of painting, writing, composing, building, etc., characteristic of a particular period, place, person, or movement."*

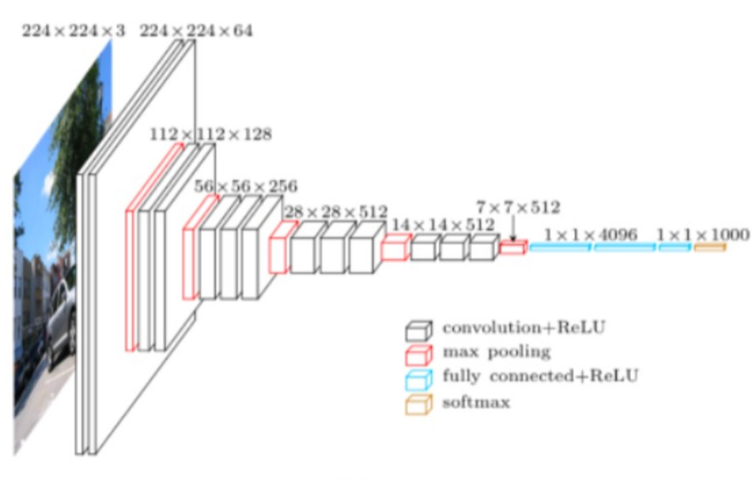
To create the image embeddings two methods are used:

- Convolutional Neural Network embeddings
- 3D colourspace nearest neighbours

Convolutional Neural Network embeddings

The model used to transfer images to embeddings borrows findings from both **neural-style transfer** and **ecommerce image content similarity work**. *A Neural Algorithm of Artistic Style* by Gatys et. al and writing by @Raghul Asokan was used to research neural-style transfer. Creating an image similarity function with TensorFlow and its application in e-commerce by Nina Pakhomova was used to develop an understanding of similar image retrieval techniques.

Both style transfer and image similarity techniques leverage pre-trained convolutional (CNN) image classification models to extract and abstract information from images. These models have been trained to extract information for classification purposes, with the intermediate convolutional layers capturing increasingly complex information about the contents of the image as the depth of the network increases.

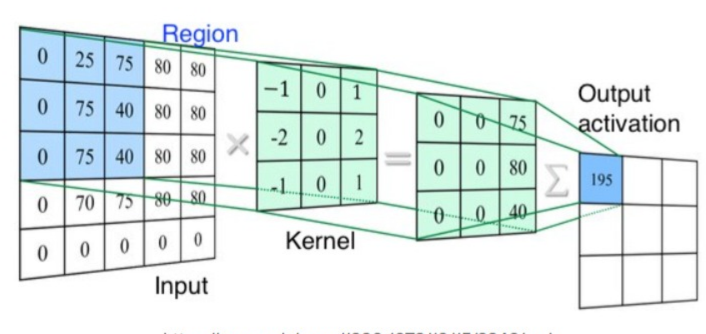


source

For this I will be using VGG-19 model trained on the image-net database, the architecture of which is shown in the diagram above. To extract artistic style using this network we must first make a few subjective assumptions about what defines style in the context of the task at hand.

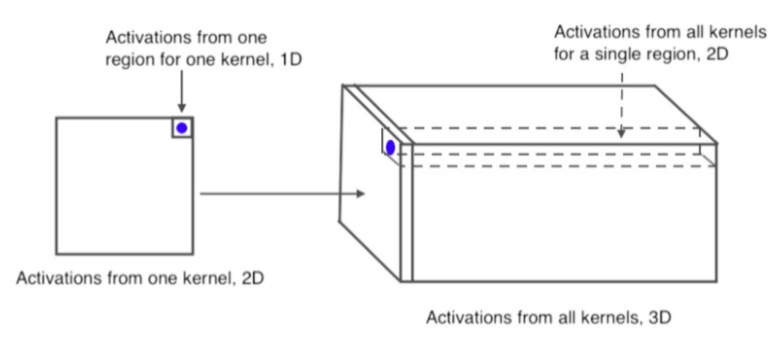
- We assume that a style can be described as a combination of various textures and colours used together.
- We also assume that sub-features, small repeating parts of the overall image, are important to the style we experience. For example, a pencil drawing of a building and of a landscape my have similar textures, but the sharp right angles and windows of the building lend the architectural drawing a style that is distinct to that of the landscape.

Luckily we know that CNNs capture and extract this information in each convolutional layer, with each kernel "searching" for a texture or sub-feature. When that texture or sub-feature is present in the underlying image, the kernel outputs a high value when it samples that region of the image. The activation process for a single kernel is shown below.



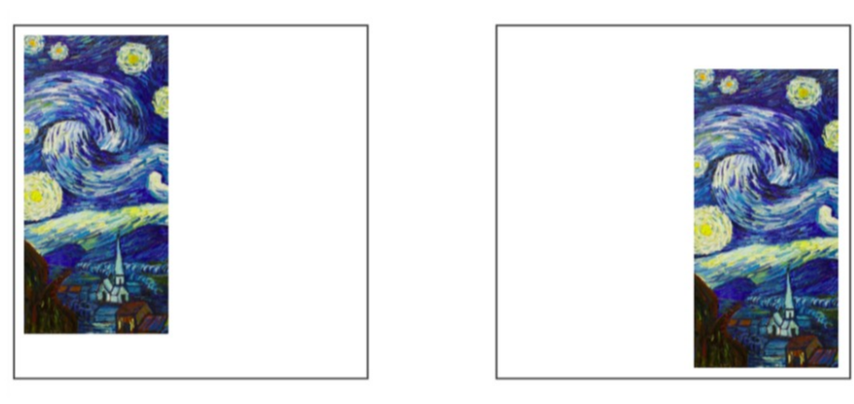
https://www.mdpi.com/1996-1073/12/15/2846/xml

When all kernels for a layer are passed over the input, a 3D matrix with a depth equal to the number of kernels, k is output.



The deeper into the network these kernels are, the more complex the extracted textures, with the deepest kernels activating with entire structures such as eyes or windows. This behavior is demonstrated in deep dream images.

Using our above assumptions about what defines art style and knowing that each kernel captures different distinct elements of art style, we can use the combination of activations from all kernels at a specific region as a mathematical description of the art style that region. To summarize these findings over all regions, one could simply take every single activation and flatten them into an embedding vector, as is done in **ecommerce image content similarity.** However, that approach is inefficient as it requires that every activation is stored and is sensitive to spacial variations over the image. That would result in the two images below being rated as very dissimilar in style despite only varying spatially.



To overcome these issues *Gatys et. al* use gram matrices in their neural style transfer paper. Gram matrices compute the correlation between the outputs of every kernel. Kernel combinations that show high correlation in their outputs are taken to be descriptive of the style of the image. For example in Starry Night above, the textures of individual brushstrokes and swirls would be highly correlated over both of the images, and as such the gram matrices of the images would be similar.

Computationally, gram matrices are calculated by flattening the convolutional output into a 2D array, with columns for each kernel and rows for every activation, A. The dot product of this array with its transpose is taken and output as the gram matrix, G. Here G(i,j) gives the correlation between the activations of kernels i and j. *Note that the term "correlation" is used rather loosely as this is not the statistically true correlation. Potentially the term cumulative co-activation would be more accurate.*

$$A = \begin{bmatrix} A_{0,0,0} & A_{1,0,0} & \cdots & A_{k,0,0} \\ A_{0,1,0} & A_{1,1,0} & \cdots & A_{k,1,0} \\ \vdots & \vdots & & \vdots \\ A_{0,n,m} & A_{1,n,m} & \cdots & A_{k,n,m} \end{bmatrix} \qquad G = X \,.\, X' = \begin{bmatrix} G_{0,0} & \cdots & G_{0,k} \\ \vdots & \ddots & \vdots \\ G_{k,0} & \cdots & G_{k,k} \end{bmatrix}$$

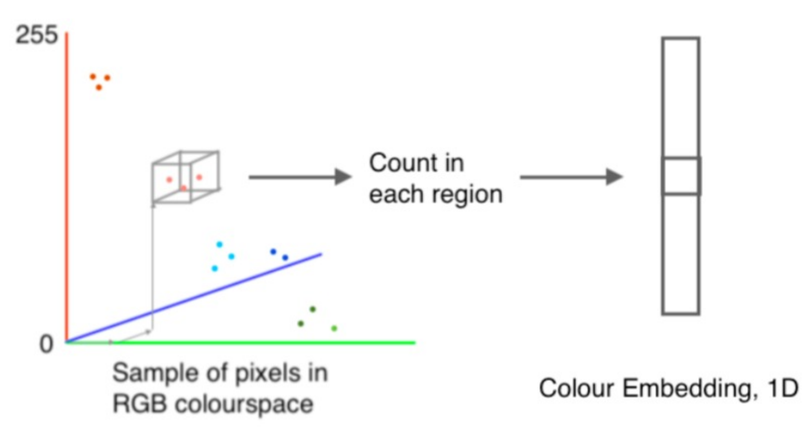
$A_{\text{Kernel, Row, Column}}$

The size of the gram matrix produced is k^2 which is flattened into a vector for storage. Which intermediate layer(s) to extract the kernel outputs from was determined by trial and error, with emphasis placed on performance, as the deeper the layer(s) used the more calculations and compute time required.

3D colourspace nearest neighbours

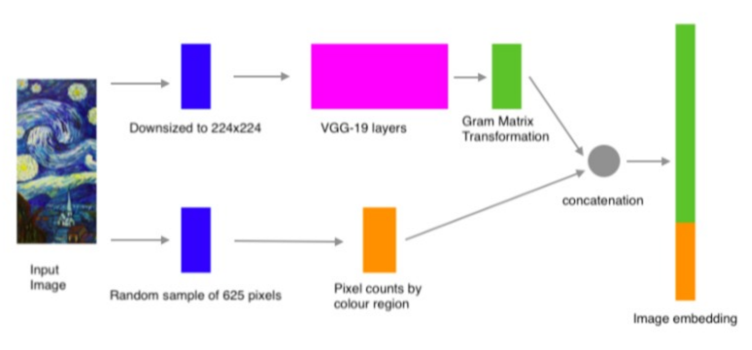
Which colours are present is an obvious visual descriptor and one that we did not want to overlook. While gram matrices will extract colour information, we used a secondary sampling approach to explicitly encode the dominant colours present. Colour dimensionality reduction is well documented and usually to extract the dominant colours in an image I would use a clustering approach such a K-means. However, in this case I want the express the colours of the image as a vector where each element always corresponds to the same colour, allowing comparison between images.

To achieve this the 3D RGB colourspace of an image is divided into n^3 mutually exclusive cubes, each with side length 255/n. The number of pixels within each cube are counted, and the counts are stored in a vector of length n^3.



Embedding construction

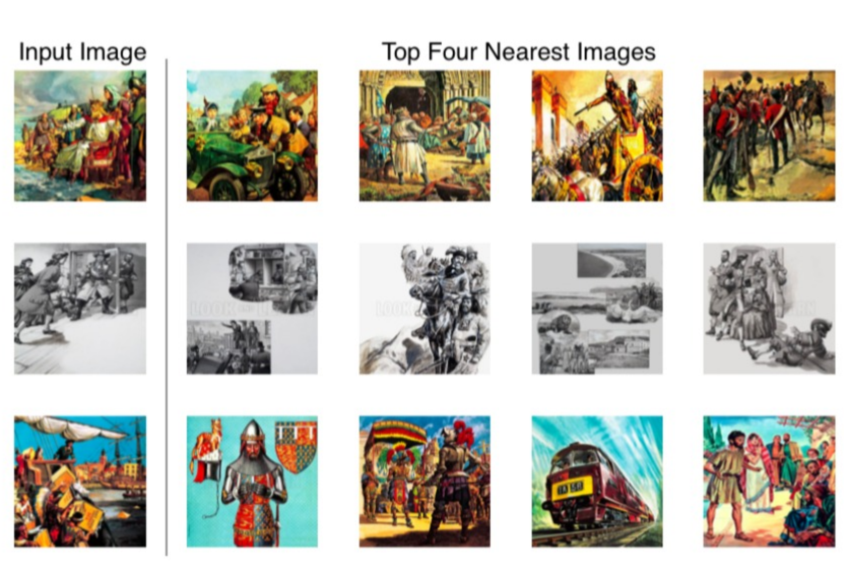
Finally, the flattened gram matrices are concatenated with the colour embeddings to form embeddings of dimensionality $N = k^2 + n^3$.



Model Architecture

Results

A sample set of 10,000 images was used to test the model. As this method is unsupervised and requires no further training this small set is sufficient.



. . .

If you are reading this, thanks for making it this far! Feel free to follow along as this project continues.