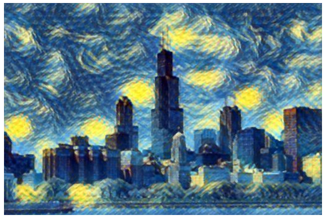
Neural Style Transfer: Feature extractor, Training Loop

The objective of *Neural Style Transfer*:

- Given Content Image C
- Given Style Image S
- Create Generated Image G that is the Content image re-drawn in the "style" of the Style image

+

=



Style image S	Content image C	Generated image G

We used this example to preview the concept that Deep Learning is all about defining a Loss Function that captures the semantics of the task.

Content Loss and Style Loss

Neural Style Transfer is solved, like most other Machine Learning tasks, by minimizing a loss

$$G = \operatorname*{argmin}_I \mathcal{L}$$

- where I is an image.
- $\mathcal{L} = \mathcal{L}_{\text{content}} + \mathcal{L}_{\text{style}}$
    - where
        - $\mathcal{L}_{\text{content}}$ measures the dissimilarity of the "content" of G and "content" f C
        - $\mathcal{L}_{\text{style}}$ measures the dissimilarity of the "style" of G and "style" of C

That is: the "weights" we are optimizing are the *pixels of image* I.

How do we measure the dissimilarity of the "content" ?

We can't just use plain MSE of the pixel-wise differences

- G is different than C, by definition (the "styles" are different)

And how do we define what the "style" of an image is ?

- And how do we measure dissimilarity of the "style" ?

Recall that each layer in a multi-layer Neural Network is creating an *alternate representation* of the input.

Rather than directly comparing G with C (and G with S) our dissimilarity will be measured

- Not on raw images as seen by the human eye
- But on their alternate representations as created at some layer of a multi-layer Neural Network

We will

- Use a pre-trained multi-layer Image Classifier $\mathbb{C}$ (e.g., VGG19)
- Define some layer l_c to be the "content" layer
- Define some layer l_s to be the "style" layer
- And measure the dissimilarity via the alternate representations created at the respective layers

Suppose $\mathbb{C}$ consists of a sequence of CNN Layers

Let $\mathbb{C}_{(l)}$ denote the set of $n_{(l)}$ feature maps produced at layer l

- Feature map: value of one feature, at each spatial location

We choose

- One layer l_c of $\mathbb{C}$ and call it the "content representation" layer
    - Will tend to be shallow: closer to the input
    - Features of shallow layers will be more "syntax" than "semantics"
- One layer l_s of $\mathbb{C}$ and call it the "style representation" layer
    - Will tend to be deep: closer to the output
        - Features of deep layers will be more "semantics" than "syntax"

For arbitrary image I, let

- $\mathbb{C}_{(l_c)}(I)$
  - denote the feature maps of the Classifier $\mathbb{C}$, on image I, at the "content representation" layer
- $\mathbb{C}_{(l_s)}(I)$
  - denote the feature maps of the Classifier $\mathbb{C}$, on image I, at the "style representation" layer

We can now define the dissimilarity of the "content" of Content Image C and "content" of Generated Image G

- by comparing $\mathbb{C}_{(l_c)}(C)$ and $\mathbb{C}_{(l_c)}(G)$

Similarly, we can define the dissimilarity of the "style" of Content Image C and "style" of Generated Image G

- by comparing $\mathbb{C}_{(l_s)}(S)$ and $\mathbb{C}_{(l_s)}(G)$

For any image I: $\mathbb{C}_{(l)}(I)$ consists of $n_{(l)}$ feature maps.

We need to define what it means to compare $\mathbb{C}_{(l)}(I)$ and $\mathbb{C}_{(l)}(I')$.

The *Gramm Matrix* $\mathbb{G}$ of $\mathbb{C}_{(l)}(I)$

- Has shape ($n_{(l)} \times n_{(l)}$)
- $\mathbb{G}_{j,j'}(I) = \text{correlation}(\text{flatten}(\mathbb{C}_{(l),j}(I)), \text{flatten}(\mathbb{C}_{(l),j'}(I)))$
    - the correlation of the feature map j of $\mathbb{C}_{(l)}(I)$ with feature map j' of $\mathbb{C}_{(l)}(I')$

Intuitively, the Gramm Matrix

- measures the correlation of the values across pixel locations (flattened feature maps) of two feature maps of image I

We can now define the dissimilarity of $\mathbb{C}_{(l)}(I)$ and $\mathbb{C}_{(l)}(I')$

- As the MSE of $\mathbb{G}(I)$ and $\mathbb{G}(I')$

Using this dissimilarity measure, we can define the

- $\mathcal{L}_{\text{content}}$ as the dissimilarity of $\mathbb{C}_{(l_c)}(C)$ and $\mathbb{C}_{(l_c)}(G)$
- $\mathcal{L}_{\text{style}}$ as the dissimilarity of $\mathbb{C}_{(l_s)}(S)$ and $\mathbb{C}_{(l_c)}(G)$

Gradient ascent: generating G

We can find image G via Gradient Ascent

- Initialize G to noise
- Update pixel $G_{i,i',k}$ by $-\frac{\partial \mathcal{L}}{G_{i,i',k}}$

Feature extractor

One key coding trick that we will illustrate

- Obtaining the feature maps of the Classifier $\mathbb{C}$, on image I, at an arbitrary layer

We will call this tool the *feature extractor*

Here (https://www.tensorflow.org/tutorials/generative/style_transfer) is a tutorial view of the notebook.

In [2]:

```python
print("Done")
```

```
Done
```