# Neural Style Transfer for Text

Ryan Butler, Yuji Akimoto, Luca Leeser, Cameron Ibrahim

*NLP Team, Cornell Data Science*

**Abstract**

Neural Style Transfer is a technique where the artistic style and content of different images can be separated and mixed back together to create new images. This can be used, for example, to transform photographs into paintings into a cubist or impressionist style. We plan to develop a similar technique for style transfer for text.

## 1. Introduction

Neural Style Transfer is a technique conventionally used to stylize images, introduced by Gatys et. al. in 2015. It was observed that due to the layered structure of convolutional neural networks, later layers in the network produce feature representations of the content of the image that are less sensitive to individual pixels, while correlations between features give a good representation of artistic style. Using a loss function with a content term and a style term, this algorithm is able to iteratively produce an image that combines the content of one image with the style of another.

*images here*

In the original Gatys et. al. paper, a very deep convolutional neural network called VGG19 was used with slight modifications to generate the content and style feature embeddings. VGG19 was originally trained on the ImageNet dataset, which means that the architecture had to learn useful features for a broad variety of image classes rather than just a select few. We hypothesize that because VGG19 could identify useful features in such a large number of classes, it had to learn features that would generalize for almost any image, and only the fully connected layers of the network performed any class specific feature transformation. This was important because the images that the network would be extracting style and content from were things the architecture never was trained for.

Once a suitable trained neural network architecture was identified for images, one or more layers were selected as the content layers and several layers for the style layers. The content layers were usually later in the network in order to ensure that the content was tolerant of per-pixel changes in the image. The network was then fed a style image such as /textitThe Starry Night, and the style embeddings were saved. It was then fed a content image such as a photo of a city, and the content embeddings were saved. Then, an L2 loss function was used to perform regression on the content and style embeddings simultaneously. This loss function was then minimized by using gradient descent to iteratively update what initially was a random noise image that was fed into the neural network.

*show style and content reconstructions*

As shown above, this has proven very successful for images, with variations that provide improvements on speed and quality. However, attempts at style transfer for text have been limited and have not been as successful. We plan to use the concepts from neural style transfer for images to expand on existing research in writing style conversion. This research has several useful applications. It could be used to make older texts, such as those written in old English, more readable by converting them to modern English. It could be used creatively to make text sound like Shakespeare or Hemingway. It could even be used to make text written by users online more coherent and understandable.

## 2. Research Goals

GOL!!!!! GANEEEEEE!!!
fnnfnfkldlkdlk;
aksdjfklasj;fdljkas;ldf
asdfjfjalkdsl;fjl;sdjf

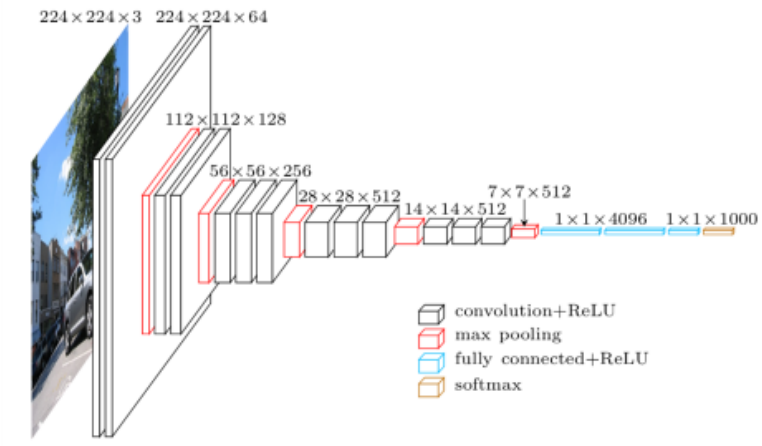



Figure 1: The VGG16 Network [1]

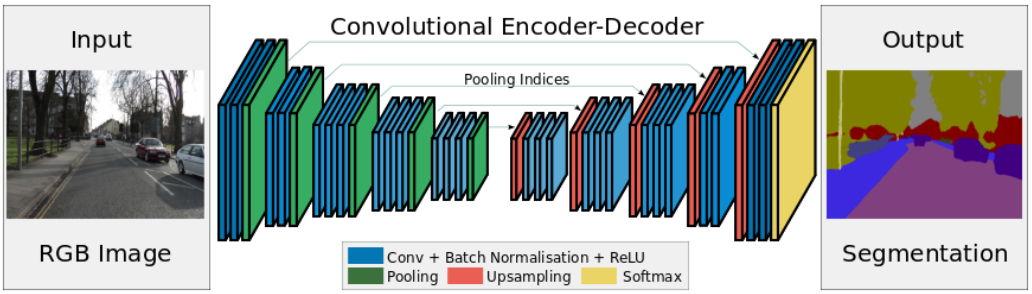



Figure 2: The SegNet Architecture [2]

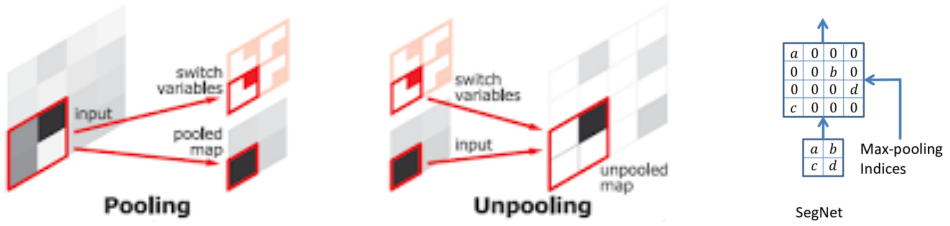



Figure 3: In this image, the switch variables are the masks, which indicate the indices of the max-pooled values. They are then passed into the unpooling layer to upsample the layer, an example of which is given in the image to the right. [3][2]

## 3. Creating an Architecture for Text

List the requirements for the architecture to have.

### *3.1. Architecture 1*

Explain architecture structure. Outline Classification/Regression results. Present reconstruction results.

### *3.2. Architecture 2*

Explain architecture structure. Outline Classification/Regression results. Present reconstruction results

jfjfjdjkdkdkdkdkdkd

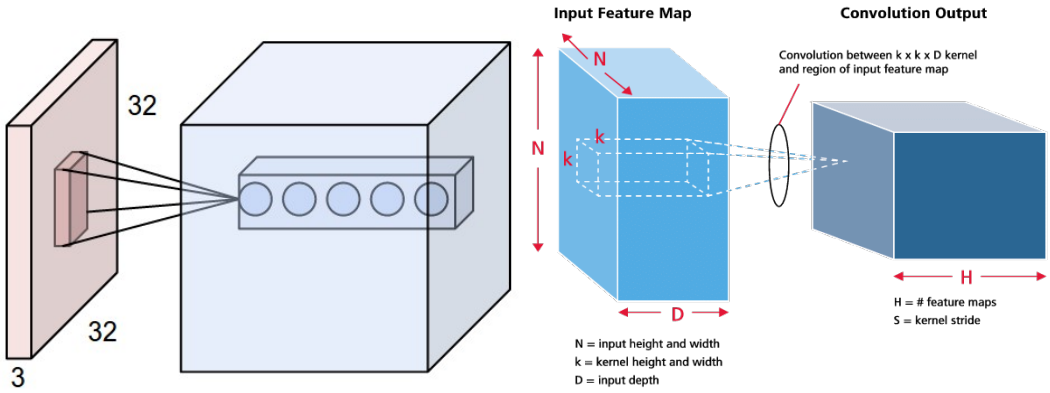



Figure 4: Left[4]: An example input volume in red (e.g. a 32x32x3 image), and an example volume of neurons in the first 2D Convolutional layer. Each neuron in the convolutional layer is connected only to a local region in the input volume spatially, but to the full depth (i.e. all color channels). Note, there are multiple neurons (5 in this example) along the depth, all looking at the same region in the input. This means that there will be 5 features for each pixel in the output volume of that layer. Right[5]: Another example of 2D convolution, this time also outlining the shape of the kernel. Note how the kernel (the smaller dotted box) extends the full channel depth of the input volume, but only k units in each spatial dimension.

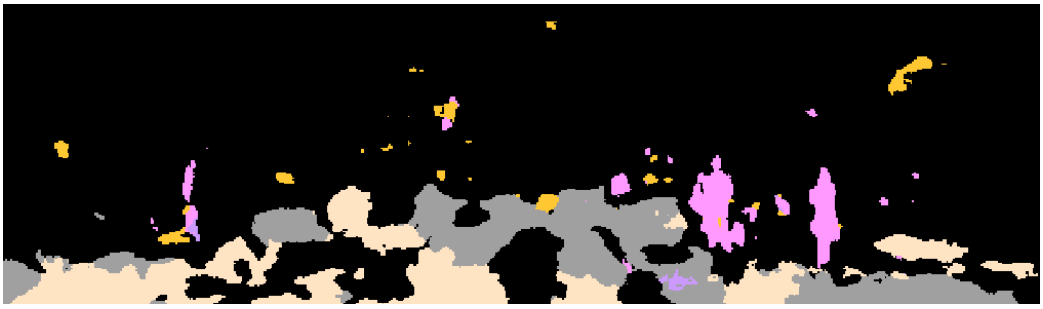
## 4. Style Transfer Results

waow!

## 5. Conclusion

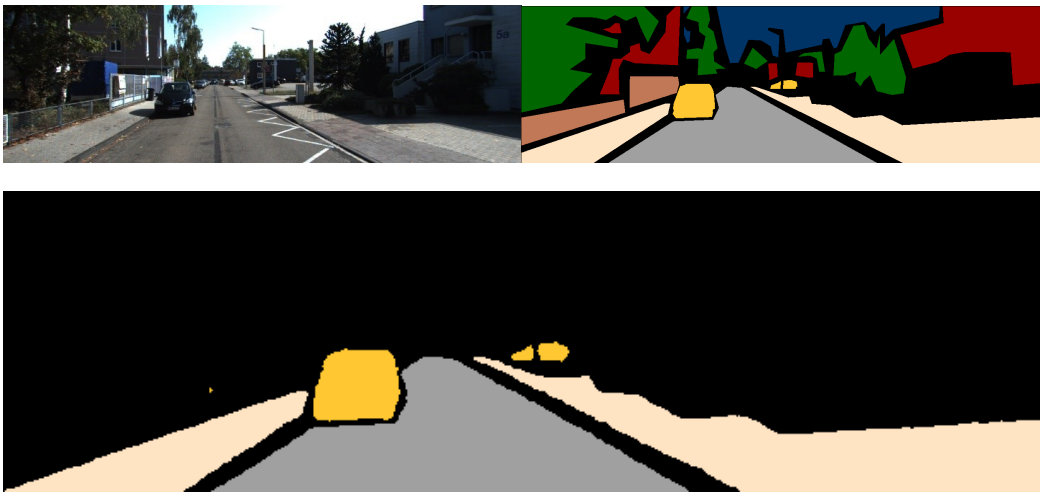
dkdkdkdk

mmkmmbnnfmm



xvzvbcbbzn

uooyyotptp



bnvcdms,a

68954ortkjgbkvo

| | Training | Testing |
|---|---|---|
| Image | 99.61035% | 85.46990% |
| LiDAR | 99.87711% | 99.87565% |

Table 1: The accuracy of the two models on training and testing data.

ewrtrweqw eqrtwrewt

## References


[1] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, CoRR abs/1409.1556 (2014).

[2] V. Badrinarayanan, A. Kendall, R. Cipolla, Segnet: A deep convolutional encoder-decoder architecture for image segmentation, CoRR abs/1511.00561 (2015).

[3] H. Noh, S. Hong, B. Han, Learning deconvolution network for semantic segmentation, CoRR abs/1505.04366 (2015).

[4] A. Karpathy, Cs231n convolutional neural networks for visual recognition, 2016.

[5] D. A. Shoieb, Computer-aided model for skin diagnosis using deep learning, 2016.