Learning Furniture Shapes from Stock Images with Transparent Backgrounds

Ian Torres
University of Massachusetts Amherst

Vicente Ordóñez
University of Virginia

August 8, 2017

1 Introduction

We have drafted a Convolutional Neural Network (CNN) that can be used to identify chair object masks that have been extracted through a preprocessing procedure. The preprocessing procedure that we had used during the study was to extract the alpha channel, transparency layer, of a given image and shape it into a grayscale image to be further manipulated through our CNN pipeline. This pipeline, modeled after LeNet, operated on chair object masks that were collected from the internet using n-gram queries generated from LEVAN [1]. Our goal with this study is to include other furniture categories into our CNN model to identify further furniture foregrounds that can be used in image composition tasks.

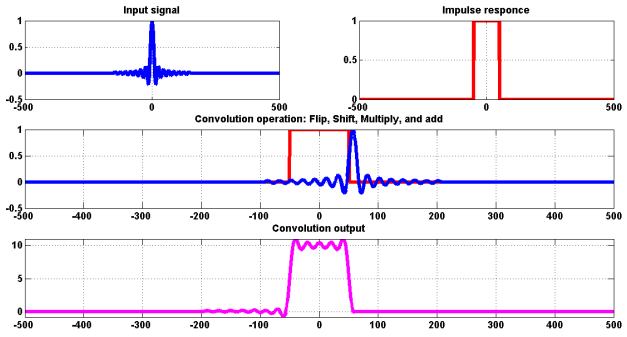

Figure 1: Signal Convolution
Top Left: Input Signal, Top Right: Signal Impulse, Middle: Signal Convolving with Signal Impulse, Bottom: Resultant Signal

2 Background

2.1 What is Convolution?

Convolution is a mathematical operation that is used to modify signals (**Figure 1**) such as image color channels by combining these signals with special function signals to produce a third signal [2]. To put this definition simply this operation takes data that is rather unclear to an algorithm or equation and makes it fall into a standard distribution of values. Edges for example are important features in an image (**Figure 2**).

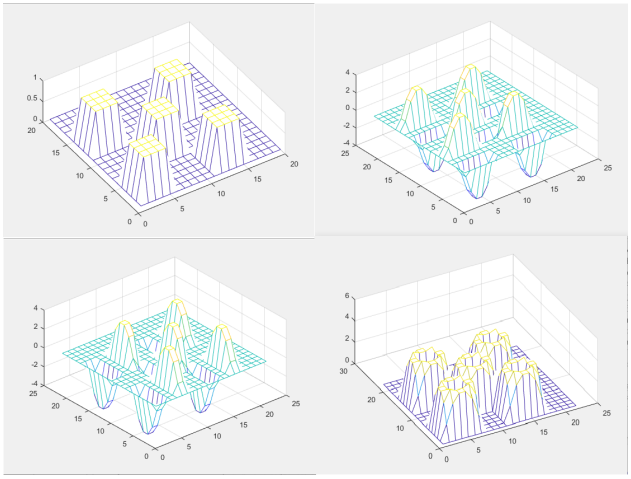

Figure 2: Top Left: Pedestals
Top Right: Vertical Edges
Bottom Left: Horizontal Edges
Bottom Right: Combined edges

2.2 What is a Neural Network?

A neural network is a biologically-inspired programmatic method that enables a computer to learn from input datasets. For example, tasks such as visual recognition of handwritten alphanumeric sequences seem trivial to humans, because most people learn at a very young age how to interpret such sequences and know what they represent. However, a computer does not have such capabilities. An algorithm must be programmed into the computer to tell it how to interpret such patterns, but handwriting is never precisely consistent, so there are many different forms of the same number or letter and, by extension, the sequence itself [3].

Neural networks take care of this problem by taking a page out of the human minds innermost functionality and representing it on a machine. Much like humans learn from example, machines are designed to learn the same way. Vast collections of datasets, handwritten letters or digits for example, are stored on the machines drive, disk, or storage device, which is then used as training examples. Nodes, a data structure that holds specific operations or values, represent a neuron while the node links represent synapses. Collections of these nodes, form various layers of the neural network, which have weights that dictate how important a piece of data is to the network. As the number of training examples increases so does the computers ability to recognize variation. A small caveated to this solution is the level of complexity that is required to solve such a problem and the hardwares ability to process such complexity (**Figures 3 and 4**)[4].

2.3 What is a CNN?

Convolutional neural networks are a type of multilayer neural network that primarily operates on image data using operations such as convolution, subsampling, and pooling to compose the various layers of nodes in the network. The advantage of using CNNs is the fact that the number of parameters needed for classification are far fewer than fully connected multilayer neural networks, and are simpler to train [6]. These CNNs, however, can also be applied to natural language processing, sentence classification, and speech signals [5,6]. LeNet is a type of CNN architecture that has been used to teach scientists the relative basics of how CNNs operate on input images. Originally, it was designed to operate on the MNIST handwritten digit dataset, one of the most well-known datasets in machine learning (**Figure 5**)[4]. However, this CNN architecture of several layers of convolution, subsampling, and pooling, can be extended and modified for tasks other than recognizing digits such as identifying chair masks (**Figure 6**).

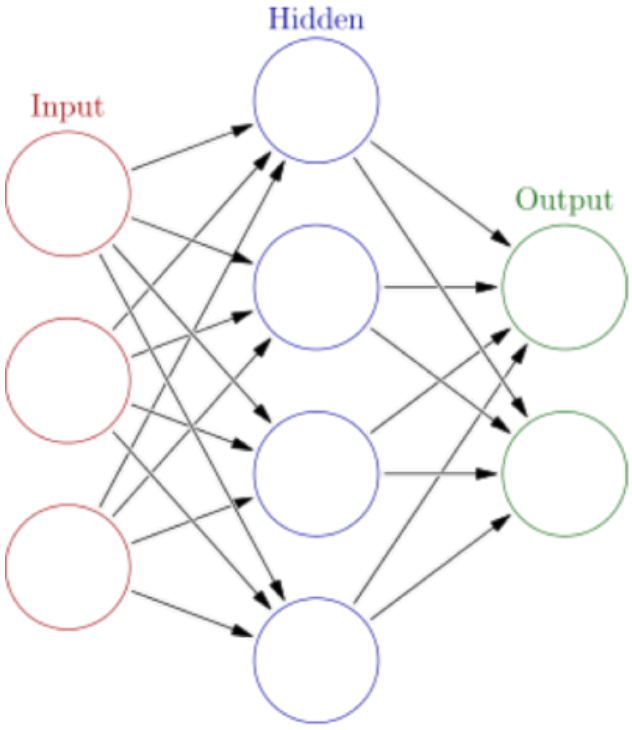

Figure 3: Neural Graph
Circle: node/neuron in the network, Arrows: links/synapses, Labels: layers of network that have operations attributed to them

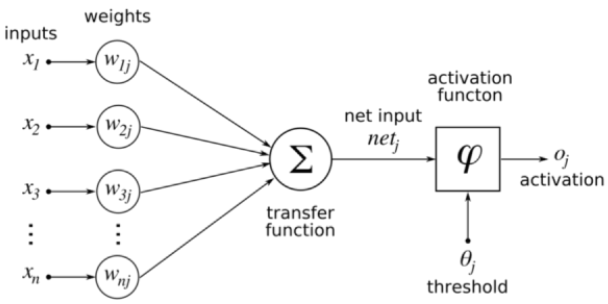

Figure 4: Neural Network

Figure 5: MNIST dataset sample

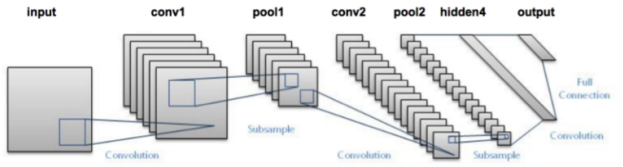

Figure 6: LeNet Architecture

3 Experiment

3.1 Data Collection

To collect reliable research data, we used Google Images coupled with transparency search filters along with Levan n-grams queries. An n-gram is simply a Markov chain of words that complement each other (e.g. wooden chair, metal chair, etc.) [6]. By searching through different chair image queries, we could collect far more chair images than with a general search query chair. Though some queries contained duplicate results this did not matter too much, since most of the query results contained vastly different types of chairs with unique shapes. However, some of these results contained images of non-chair objects such as people and logos. To remedy this, the tedious task of annotating over 5000 images of the total number of images was required to ensure all training data was consistent and reliable. From this data, we collected an additional 16000 training images in total by taking the rest of the images that were queried under an n-gram category and label them as non-chair objects.

After training the model using the previously labeled data, we generated additional image data by taking the chair foregrounds that were previously labeled, compositing them with pictures of rooms, and running the GrabCut algorithm on the composited image. GrabCut is an algorithm that takes labeled regions of an image designated by a user input mask and calculates a statistical distribution of grayscale intensities in the mask region to generate a foreground estimate of the image (**Figure 7**). This process was repeated for 10000 composite images. In total, the research data collected and generated numbered over 36000 chair image masks each of size 48x48 pixels. In the future, we will be able to improve the efficiency of the CNN by incorporating the remaining chair mask data into the model for chair classification. The same pipeline can be used to collect image data for other furniture categories in the future.

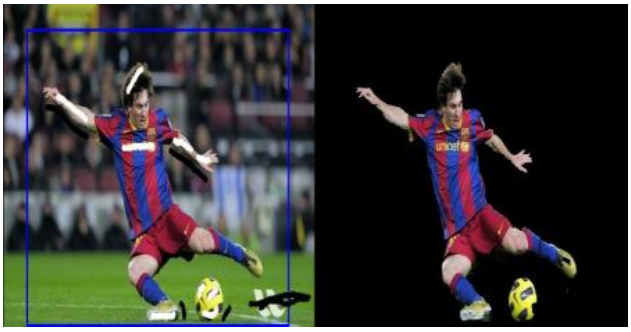

Figure 7: GrabCut Algorithm
Left: input image and mask area
Right: resultant image foreground

3.2 Experimental Setup

To run the model and manipulate the images, we chose to use the Python programming language along with the pytorch/torchvision neural network framework. Other modules such as selenium, beautiful soup, and icrawler were used to collect the image data.

3.3 Model

Our model (**Figure 9**), as mentioned previously, was drafted using the same architecture as LeNet, but instead of operating on 32x32 pixel images it operated on 48x48 pixel images as well as using the ReLU (Rectifier) function:

$$f(x) = x^+ = max(0, x) \tag{1}$$

as an activation, node output, instead of a sigmoid function.

$$f(x) = tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{2}$$

Such activation functions are necessary to make sure that the CNN weights that are calculated fall within an acceptable range of values, within

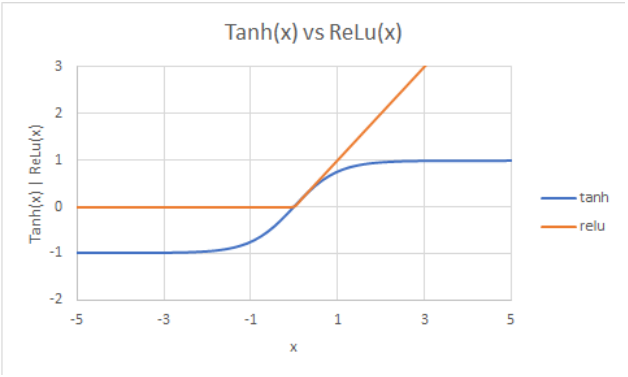

Figure 8: Orange: ReLu(x), Blue: tanh(x)

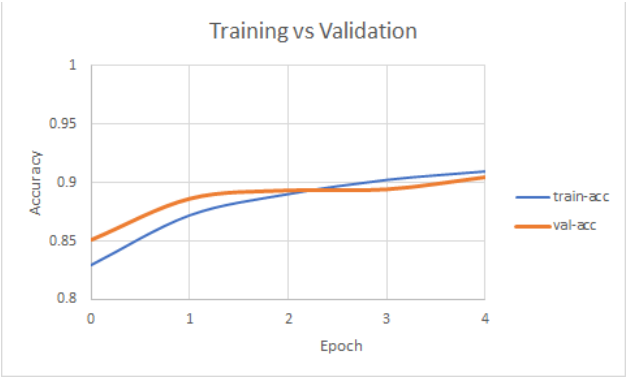

Figure 10: First Experiment Accuracies

some distribution [0, n], in the case of ReLU. If negative outliers can pass through each nodal layer than the classification task will be extremely influenced by image variations that do not occur too often throughout the training data, such as a person or animal outline being visible in the chair mask. Pooling operations are used to minimize the number of data parameters that are passed through each network layer. The network, unless these parameters are needed, will often run more efficiently and quickly if the level of complexity between each layer is essentially halved in terms of data output. These operations are then repeated a certain number of times, these repeat units are called epochs, and for each epoch a backpropagation is computed over all network layers to calculate the network weights and ensure that they do not assume a random sequence but are driven towards a certain direction [3,7].

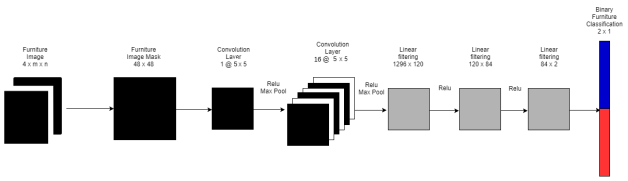

Figure 9: Furniture Foreground Mask CNN Pipeline

4 Conclusion

4.1 Results

When running the validation experiments, we found that we could obtain the following accuracies (**Figure 10 and 12**). However, when the generated GrabCut composite image data was incorporated into the training data set there was a slight decrease of accuracy for both validation and testing experiments, but this is to be expected since the data that was incorporated was finer-grained in terms of chair image mask qualities. Though the internship has ended with

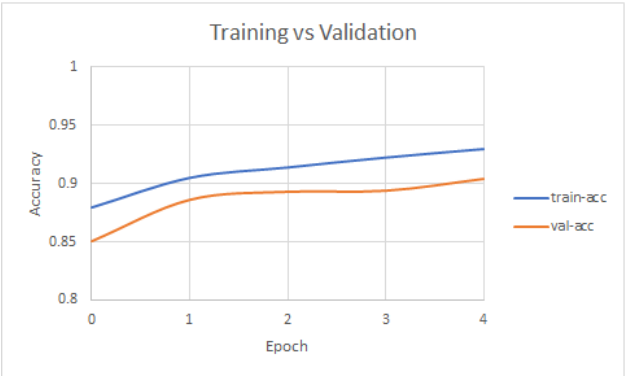

Figure 11: Second Experiment Accuracies

a few tweaks to convolution filter dimensions as well as removing pooling operations these fine grained parameters should be conserved in order to increase classification accuracy (**Figure 11 and 12**).

4.2 Future Work

In the future, we hope to incorporate the rest of the chair images into the training, validation, and testing datasets, as well as incorporating further image classifications into the model. This CNN will hopefully be able to be incorporated into an automated furniture foreground data collection component that can be used in a furniture image composition application, such as an interior decorator tool. The tool can be used

to help people find furniture that appeals to an overall semantic theme of a room and place said pieces of furniture in a logical location of the room.

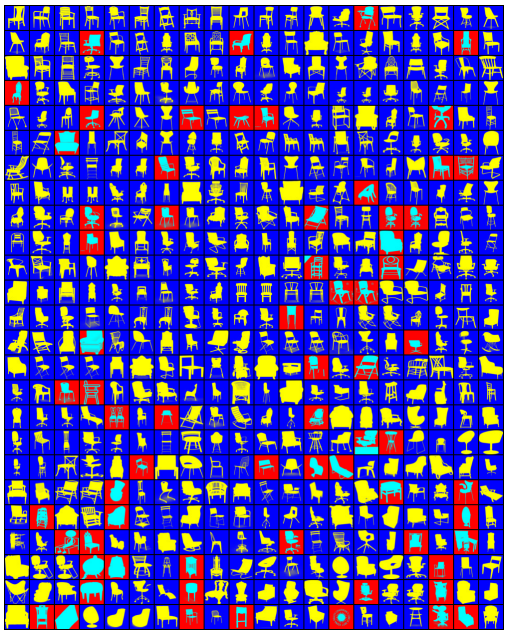

Figure 12: Blue and Yellow: Chair Object Mask, Red and Teal: Non-Chair Object Mask **with GrabCut data**

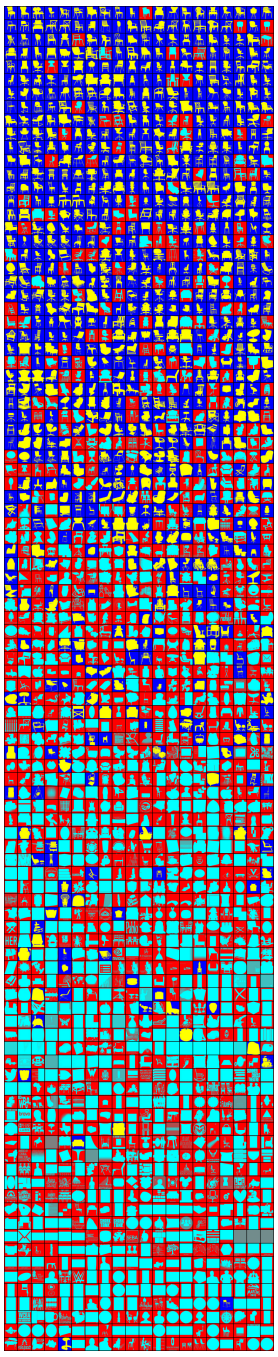

Figure 13: Blue and Yellow: Chair Object Mask, Red and Teal: Non-Chair Object Mask **without GrabCut data**

Acknowledgement

We thank the UVA Department of Computer Science in Partnership with DREU CRA-W, for hosting the intership, NSF #CNS-1246649, IAAMCS, and Access Computing for providing funding.

References

[1] Levan: Chair N-Grams

[2] Smith, Steven W., Ph.D. "Convolution" *The Scientist and Engineer's Guide to Digital Signal Processing.* Accessed July 2017. Website link

[3] Nielsen, Michael. "Neural Networks and Deep Learning." Neural Networks and Deep Learning. May 2017. Accessed July 2017. Website link

[4] Schmidhuber, Jürgen. "Deep learning in neural networks: An overview." *Elsevier* 61 (January 2015): 85-86. Accessed August 2017. PDF link

[5] Lecun, Yann, León Bottou, Yoshua Bengio, and Patrick Haffner. "Gradient-Based Learning Applied to Document Recognition." *IEEE*, November 1998, 1. Accessed August 2017. PDF link

[6] Kim, Yoon. "Convolution Neural Networks for Sentence Classification." *arXiv e-print* 1408:5882 (August 2014): 1. Accessed August 2017. PDF link

[7] Jurafsky, Daniel, and James H. Martin. *Speech and Language Processing.* Ch 4. Stanford. September 1, 2014. Accessed August 2017. PDF link

[Figure 1] Convolution Graphic

[Figure 2] Inspired by Matlab Example: Conv2

[Figure 3] Neural Graphic

[Figure 4] Neural Network Graphic

[Figure 5] MNIST Dataset

[Figure 6] LeNet Architecture

[Figure 7] Grab Cut Tutorial: Messi