# **Furniture & Home Goods Recognition[¶](https://render.githubusercontent.com/view/ipynb?commit=2f3bd629f7ac539fffd27ccab4381c1a49cb1da4&enc_url=68747470733a2f2f7261772e67697468756275736572636f6e74656e742e636f6d2f6a6f616f6265636b65722f646565705f6c6561726e696e675f6675726e69747572652f326633626436323966376163353339666666643237636361623433383163316134396362316461342f6675726e69747572655f636170732e6970796e62&nwo=joaobecker%2Fdeep_learning_furniture&path=furniture_caps.ipynb&repository_id=175716967&repository_type=Repository#Furniture-&-Home-Goods-Recognition)**

As shoppers move online, it’d be a dream come true to online stores have products in photos classified automatically. But, automatic product recognition is challenging because, for the same product, a picture can be taken in different lighting, angles, backgrounds, and levels of occlusion. Meanwhile different fine-grained categories may look very similar, for example, ball chair vs egg chair for furniture, or dutch oven vs french oven for cookware. Therefore, it is time-consuming and hard for a human to categorize manually each type of furniture. This project aims to end with this problem by developing an algorithm that will help online stores move towards automatic product recognition and accurately assign category labels for furniture and home goods images.

### **Data**

The project will be using the data available on Kaggle competition - *iMaterialist Challenge (Furniture) at FGVC5*, and it will apply deep learning, convolutional neural networks, to learn from previous images and identify which category each furniture falls into.

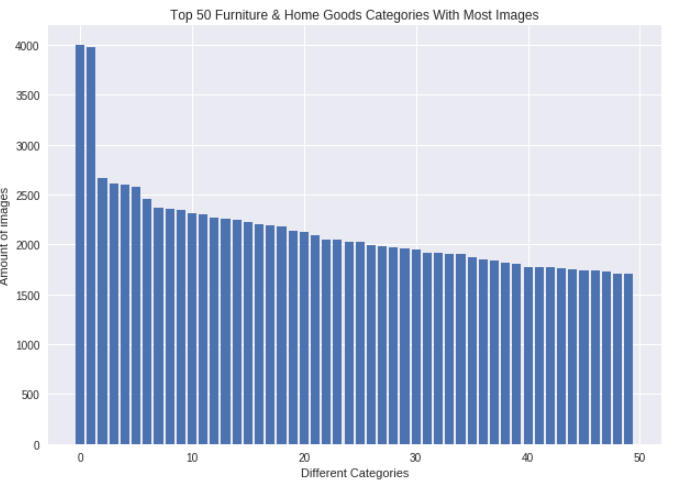
### **Data Wrangling**

The data provided on Kaggle was on a json file, which means that it only provided the URLs for the images. Thus, first, we had to transform the json file into a data frame and then clean it to make the URLs available in a format that we can easily use to fetch the images from the web and download to our machine.

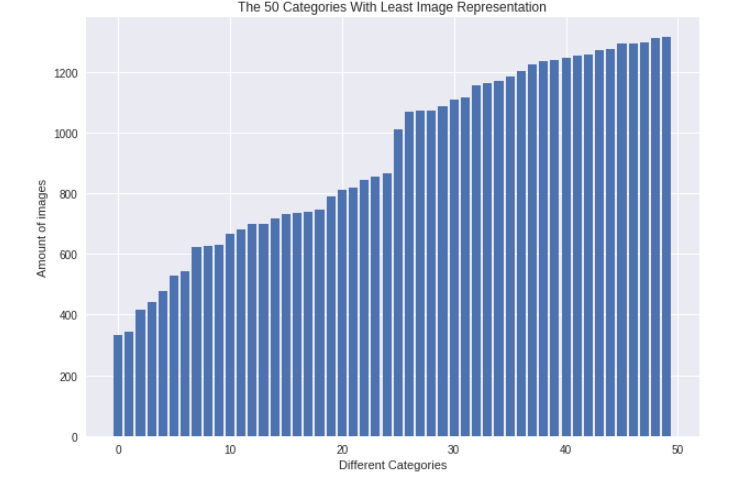
### **Exploratory Data Analysis**

By performing exploratory data analysis we found that the data comprises of 128 different categories of furniture and home goods, and a total of 194828 images. We found that the two categories with most images had close to 4000 images each, then the amount of images lowers down to 1000 to 2500 for most of the categories.

**Top 50 Furniture & Home Goods Categories w/ MOST Images**



**Top 50 Furniture & Home Goods Categories w/ LEAST Images**



To be able to perform different deep learning models, we decided to use only 8 categories with the least images on the dataset. This resulted in us having a total of 3699 images. Finally, we load some of the images to visualize how the furniture and home goods in our dataset looks like.

### 

### **Image Manipulation**

To increase the quantities of images in our dataset, we used image augmentation. Image augmentation uses the images from our dataset to create new ones by using techniques as flipping, zooming and adding noise.

### **Deep Learning Models**

The project leverages the power of transfer learning to use the weights and architecture of two models that have been previously trained in millions of images. These two models are VGG-16 and InceptionV3.

#### **VGG-16**

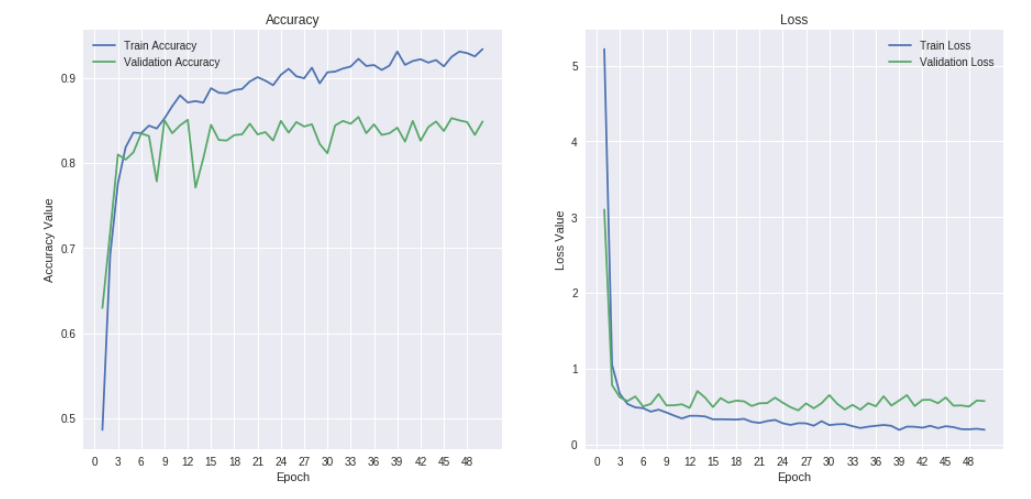
Above is the architecture used in the VGG-16 model. The interesting thing about the model is its simplicity. The VGG-16 model uses 3x3 convolutional layers stacked on top of each other in increasing depth. The volume of the image is reduced by using max pooling, then it uses flatten to create a one-dimensional array. Finally, it uses two fully connected layers each with 4,096s nodes, followed by a softmax classifier.

The project applied different transfer learning techniques on the VGG-16 model. We first use the pre-trained model as a feature extractor and then we fine-tuned the pre-trained model.

**Feature Extractor**

By using feature extractor we froze all the layers and weights on the model but the fully connected layers, meaning that we will be using the layers and weights that have been previously trained on VGG-16 and train our images on the fully connected layers of the model.

* The pre-trained model as feature extractor without regularization achieved an accuracy of 93.4% and validation accuracy of 85%.



We also tested the feature extraction method with regularization to see how the model performs, by using the dropout method. The dropout method is called like that because it drops units out in a neural network. Thus, during the training phase, a set of neurons are going to be chosen, at random, and they will be ignored from the model, meaning that these dropouts neurons are not going to be used during the forward and backward propagation.

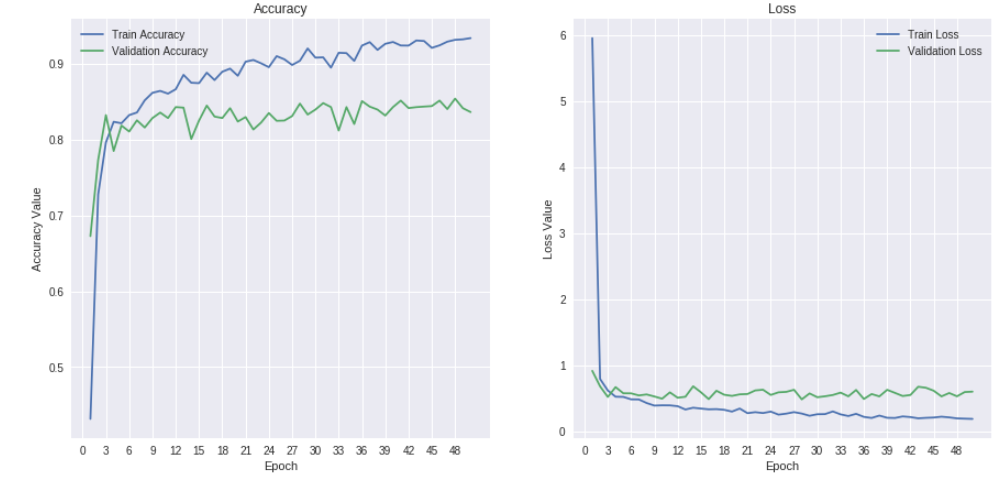
* The pre-trained model as feature extractor with regularization achieved an accuracy of 78.9% and validation accuracy of 83%. While accuracy on the trained set was not very high, it performed quite well on unseen categories. However, still slightly lower than the model without regularization.



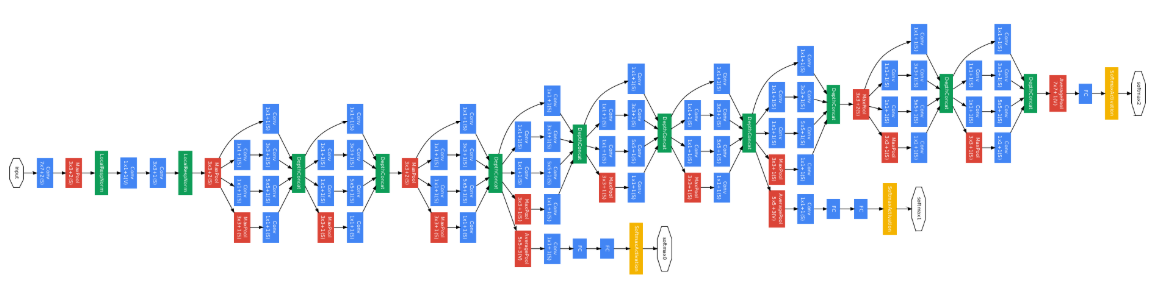
**Fine-tuning**

Fine-tuning is similar to feature extractor, in which we freeze a great part of the model layers and weights. But, in addition to the fully connected layers, we also leave some blocks of the model to be trainable and test with our images. This means that we will still use the architecture of the model, but the weights on the trainable blocks are going to be computed by testing our images rather than what was previously computed. In this project, we set block 4 and 5 of the VGG-16 model to be trainable

* By fine-tuning the pre-trained model we achieved a result very similar to feature extraction without regularization, an accuracy of 93.4% and validation accuracy of 83.6%



#### **InceptionV3**



Above is the architecture of the InceptionV3 model. The model is well-known for how deep and complex it is. For comparison, the VGG-16 has a total of 23 layers, while InceptionV3 has a total of 313 layers! The key difference with Inception and other models is that rather than choosing a single convolutional filter of 5x5 or 3x3 or 2x2, it uses them all at different flows and then it stacks them up together with a concatenate function. Other characteristics of the model are: The model uses a combination of Max pooling and Global Average pooling and it uses the Batch Normalization method. Batch Normalization is a technique used for improving the performance and stability of neural networks. It normalizes the inputs of each layer in such a way that they have a mean output activation of zero and a standard deviation of one. It is seen as a regularization method, because it helps to add some noise to the network, causing a similar effect to the Dropout method.

* We used this pre-trained model just as feature extractor. The model was able to achieve slightly higher accuracy than VGG-16, 93.6% accuracy. However, the validation accuracy, the accuracy on training on unseen data, was significantly lower, 78.2%.



### **Results**

By leveraging transfer learning from VGG-16 model as feature extractor to our model, we achieved an impressive result of 93.6% accuracy and 85% validation accuracy, even though we only had a total of 3’699 images available in our dataset. Thus, helping online stores automate product recognition with high accuracy and making them more efficient

