

IMAGE CLASSIFICATION (FASHION APPAREL DETECTION) PROJECT

Submitted by:

LAKSHITA KAIN

DATA SCIENCE INTERN

**ACKNOWLEDGMENT**

I would like to express my deepest appreciation to all those who provided me the possibility to complete this report. A special gratitude I give Data Trained and Flip Robo Technologies, whose contribution in stimulating suggestions and encouragement, helped me to coordinate my project. Furthermore, I would also like to acknowledge with much appreciation the crucial role of our SME of Ms. Sapna Verma.

**1. INTRODUCTION**

Image classification (or Image recognition) is a subdomain of computer vision in which an algorithm looks at an image and assigns it a tag from a collection of predefined tags or categories that it has been trained on.

Vision is responsible for 80-85 percent of our perception of the world, and we, as human beings, trivially perform classification daily on whatever data we come across. Therefore, emulating a classification task with the help of neural networks is one of the first uses of computer vision.

Visual classification of commercial products is a branch of the wider fields of object detection and feature extraction in computer vision, and, in particular, it is an important step in the creative workflow in fashion industries. Automatically classifying garment features makes both designers and data experts aware of their overall production, which is fundamental in order to organize marketing campaigns, avoid duplicates, categorize apparel products for e-commerce purposes, and so on. There are many different techniques for visual classification, ranging from standard image processing to machine learning approaches

Our study describes a real-world study aimed at automatically recognizing and classifying 3 different objects of clothing such as Saree (Women), Jeans (Men) and Trousers (Men), solely from ﬁnal rendering images of their products.

* 1. PROBLEM DEFINITION:

Object detection is one of the important technologies in the field of computer vision. In

the area of fashion apparel, object detection technology has various applications, such as apparel recognition, apparel detection, fashion recommendation, and online search. The recognition task is difficult for a computer because fashion apparel images have different characteristics of clothing appearance and material. In fashion industries, obtaining a visual analysis of the overall production is a key aspect, both in developing marketing strategies and for helping fashion designers in the creative workﬂow of new products. Transfer learning is the process of applying knowledge and skills learned in previous tasks to target tasks.

* 1. OBJECTIVE:

Our dataset consists of 3 labels of Apparels such as Saree (Women), Jeans (Men) and Trousers (Men). This project aims to implement Deep Learning algorithms transfer learning using VGG-16, VGG-19 and Inception v3 using keras. Data used in this project are online products collected from “amazon.com”.

**2. Literature Review**

2.1 Deep Learning Algorithms

Deep learning is an important element of data science, which includes statistics and predictive modeling. It is extremely beneficial to data scientists who are tasked with collecting, analyzing and interpreting large amounts of data; deep learning makes this process faster and easier.

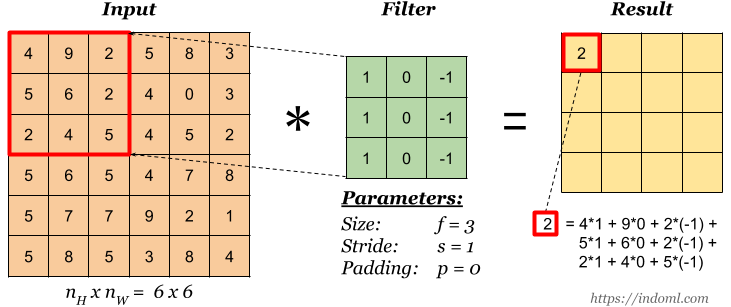
At its simplest, deep learning can be thought of as a way to automate predictive analytics. While traditional machine learning algorithms are linear, deep learning algorithms are stacked in a hierarchy of increasing complexity and abstraction.

In traditional machine learning, the learning process is supervised. The advantage of deep learning is the program builds the feature set by itself without supervision. Unsupervised learning is not only faster, but it is usually more accurate.

2.1.1 CNN

Architecture of Convolutional neural network Convolutional neural network is a type of artificial neural network that uses multiple perceptron that analyze image inputs and have learnable weights and bases to several parts of images and able to segregate each other. One advantage of using Convolutional Neural Network is it leverages the use of local spatial coherence in the input images, which allow them to have fewer weights as some parameters are shared. This process is clearly efficient in terms of memory and complexity. The basic building blocks of convolutional neural network are as follows:

a. Convolution Layer – In convolutional layer, a matrix named kernel is passed over the input matrix to create a feature map for the next layer. We execute a mathematical operation called convolution by sliding the Kernel matrix over the input matrix. At every location, an element wise matrix multiplication is performed and sums the result onto the feature map. Convolution is a specialized kind of linear operation which is widely used in variety of domains including image processing, statistics, physics. Convolution can be applied over more than 1 axis. If we have a 2-Dimensional image input, I, and a 2-Dimensional kernel filter, K, the convoluted image is calculated as follows:



b. Non-linear activation functions (ReLU) – Activation function is a node that comes after convolutional layer and the activation function is the nonlinear transformation that we do over the input signal. The rectified linear unit activation function (ReLU) is a piecewise linear function that will output the input if is positive, otherwise it will output zero.

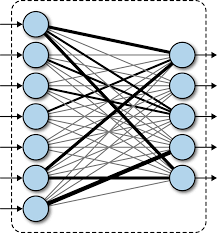


c. Pooling Layer – The drawback of the feature map output of convolutional layer is that it records the precise position of features in the input. This means during cropping, rotation or any other minor changes to the input image will completely results in a different feature map. To counter this problem, we approach down sampling of convolutional layers. Down sampling can be achieved by applying a pooling layer after nonlinearity layer. Pooling helps to make the representation become approximately invariant to small translations of the input. Invariance to translation means that if we translate the input by a small amount, the values of most of the pooled outputs do not change.

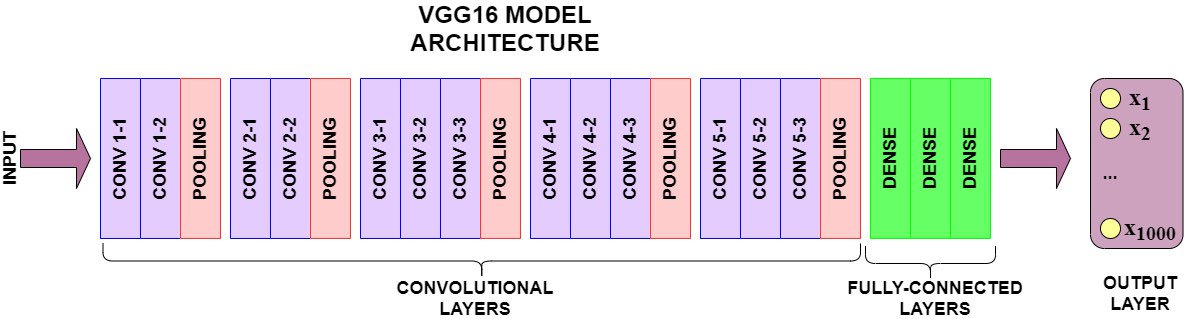


d. Fully Connected Layer - At the end of a convolutional neural network, the output of the last

Pooling Layer acts as input to the Fully Connected Layer. There can be one or more of these layers. Fully connected means that every node in the first layer is connected to every node in the second layer.



2.1.2 Transfer Learning using VGG16



VGG-16 model architecture – 13 convolutional layers and 2 Fully connected layers and 1 SoftMax classifier VGG-16 - Karen Simonyan and Andrew Zisserman introduced VGG-16 architecture in 2014 in their paper Very Deep Convolutional Network for Large Scale Image Recognition. Karen and Andrew created a 16-layer network comprised of convolutional and fully connected layers. Using only 3×3 convolutional layers stacked on top of each other for simplicity.

The precise structure of the VGG-16 network shown in Figure. 7. is as follows:

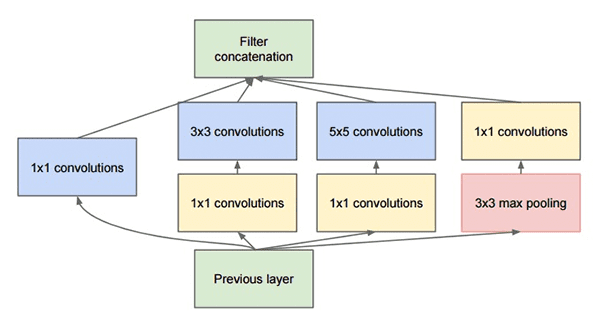
* The first and second convolutional layers are comprised of 64 feature kernel filters and size of the filter is 3×3. As input image (RGB image with depth 3) passed into first and second convolutional layer, dimensions changes to 224x224x64. Then the resulting output is passed to max pooling layer with a stride of 2.
* The third and fourth convolutional layers are of 124 feature kernel filters and size of filter is 3×3. These two layers are followed by a max pooling layer with stride 2 and the resulting output will be reduced to 56x56x128.
* The fifth, sixth and seventh layers are convolutional layers with kernel size 3×3. All three use 256 feature maps. These layers are followed by a max pooling layer with stride 2.
* Eighth to thirteen are two sets of convolutional layers with kernel size 3×3. All these set of convolutional layers have 512 kernel filters. These layers are followed by max pooling layer with stride of 1.
* Fourteen and fifteen layers are fully connected hidden layers of 4096 units followed by a softmax output layer (Sixteenth layer) of 1000 units.

2.1.3 Leveraging Transfer Learning with Pretrained Models

ImageNet is a research project to develop a large database of images with annotations e.g. images and their labels. Pretrained models like InceptionV1, Inception V2, VGG-16 and VGG-19 are already trained on ImageNet which comprises of disparate categories of images. These models are built from scratch and trained by using high GPU’s over millions of images consisting of thousands of image categories. As the model is trained on huge dataset, it has learned a good representation of low-level features like spatial, edges, rotation, lighting, shapes and these features can be shared across to enable the knowledge transfer and act as a feature extractor for new images in different computer vision problems. These new images might be of completely different categories from the source dataset, but the pretrained model should still be able to extract relevant features from these images based on the principles of transfer learning. In this paper we will unleash the power of transfer learning by using pretrained model - VGG-16 as an effective feature extractor to classify dog vs cat even with fewer training images.

2.1.4 Inception V3 using Keras

The “Inception” micro-architecture was first introduced by Szegedy et al. in their 2014 paper, Going Deeper with Convolutions:



The goal of the inception module is to act as a “multi-level feature extractor” by computing 1×1, 3×3, and 5×5 convolutions within the same module of the network — the output of these filters are then stacked along the channel dimension and before being fed into the next layer in the network.

The original incarnation of this architecture was called GoogLeNet, but subsequent manifestations have simply been called Inception vN where N refers to the version number put out by Google.

The Inception V3 architecture included in the Keras core comes from the later publication by Szegedy et al., Rethinking the Inception Architecture for Computer Vision (2015) which proposes updates to the inception module to further boost ImageNet classification accuracy.

The weights for Inception V3 are smaller than both VGG and ResNet, coming in at 96MB.

1. **Methodology**
   1. Data Collection

Data has been scrapped using Selenium web driver with python. Selenium is a powerful tool for controlling web browsers through programs and performing browser automation.

Our data has been retrieved from <https://www.amazon.com> which consists of Fashion Apparel Images such as Men’s Jeans, Trousers and Women’s Saree.

* 1. About Dataset

Our Dataset consists of Our dataset consists Images of the product. 900 images were retrieved of each of the three categories to create a balanced dataset and overcome overfitting/underfitting of any particular label.

3.3 Data Preprocessing using ImageNet

ImageNet is formally a project aimed at (manually) labeling and categorizing images into almost 22,000 separate object categories for the purpose of computer vision research. However, when we hear the term “ImageNet” in the context of deep learning and Convolutional Neural Networks, we are likely referring to the ImageNet Large Scale Visual Recognition Challenge, or ILSVRC for short.

The goal of this image classification challenge is to train a model that can correctly classify an input image into 3 separate object categories. Models are trained on 712 training images with another 172 images for validation.

When it comes to image classification, the ImageNet challenge is the de facto benchmark for computer vision classification algorithms — and the leaderboard for this challenge has been dominated by Convolutional Neural Networks and deep learning techniques since 2012.

The state-of-the-art pre-trained networks included in the Keras core library represent some of the highest performing Convolutional Neural Networks on the ImageNet challenge over the past few years. These networks also demonstrate a strong ability to generalize to images outside the ImageNet dataset via transfer learning, such as feature extraction and fine-tuning.

3.4 Training–Test Set Split

For all the models fitted in this study, we split the balanced data into 80% for the training set and 20% for the validation set.

1. Hardware and Software Requirements and Tools Used
   1. Software Requirements :

* Python
* Anaconda (Jupyter Notebook)
* Python Libraries (Scikit - Learn, Keras, Tensorflow, Pandas, Numpy , etc)
  1. Hardware Requirements :
* Minimum 4 GB RAM
* Intel Core-i3 or above processor

1. **Results**

In this section, I present analytic results of the various machine learning models adopted in this research. All model diagnostic metrics in this paper are based on the validation/test set.

* 1. Training/Testing Accuracy

Accuracy is a method for measuring a classification model’s performance. It is typically expressed as a percentage. Accuracy is the count of predictions where the predicted value is equal to the true value. It is binary (true/false) for a particular sample. Accuracy is often graphed and monitored during the training phase though the value is often associated with the overall or final model accuracy. Accuracy is easier to interpret than loss.

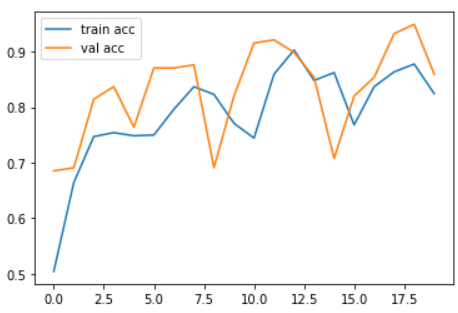


Fig: VGG16 Accuracy Graph

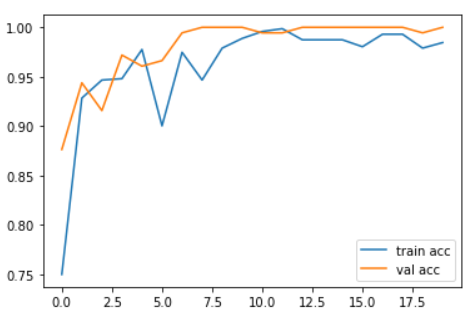


Fig: Inception V3 Accuracy Graph

* 1. Training/Testing Loss
     1. Cross – Entropy

Cross-entropy is the default loss function to use for multi-class classification problems.

In this case, it is intended for use with multi-class classification where the target values are in the set {0, 1, 3, …, n}, where each class is assigned a unique integer value.

Mathematically, it is the preferred loss function under the inference framework of maximum likelihood. It is the loss function to be evaluated first and only changed if you have a good reason.

Cross-entropy will calculate a score that summarizes the average difference between the actual and predicted probability distributions for all classes in the problem. The score is minimized and a perfect cross-entropy value is 0.

Cross-entropy can be specified as the loss function in Keras by specifying ‘categorical\_crossentropy‘ when compiling the model. The function requires that the output layer is configured with an n nodes (one for each class), in this case three nodes, and a ‘softmax‘ activation in order to predict the probability for each class.

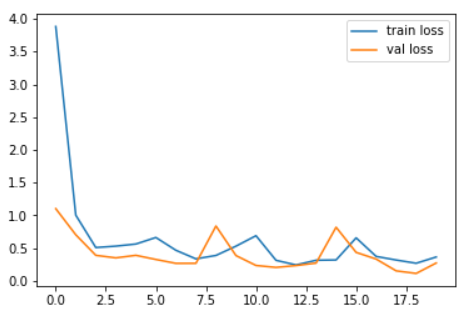


Fig: Loss for vgg 16

* + 1. Sparse – Cross -Entropy

A possible cause of frustration when using cross-entropy with classification problems with a large number of labels is the one hot encoding process.

For example, predicting words in a vocabulary may have tens or hundreds of thousands of categories, one for each label. This can mean that the target element of each training example may require a one hot encoded vector with tens or hundreds of thousands of zero values, requiring significant memory.

Sparse cross-entropy addresses this by performing the same cross-entropy calculation of error, without requiring that the target variable be one hot encoded prior to training

The function requires that the output layer is configured with an n nodes (one for each class), in this case three nodes, and a ‘softmax‘ activation in order to predict the probability for each class.

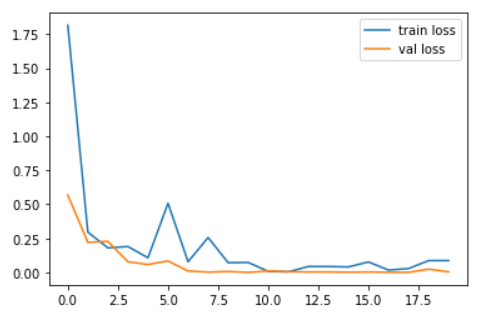


Fig: Loss for Inception V3

1. **Conclusions:**

This research evaluated the image classification of fashion apparel using deep learning models. Using real time data, we compared two CNN models VGG16 and InceptionNet V3 using transfer learning with Keras by performing detailed experimental analysis while classifying the Image detected into 3 categories.

It was further concluded that our Inception v3 using keras model outperformed vgg16.

1. **Future Scope:**

The future of sentiment analysis is going to continue to dig deeper, far past the surface of the Image classification and can focus for further smaller details for example: logos and brand names to further categorize the object into brands. Based on Faster R-CNN to produce clothing proposals

and category classifications, and we add extra branches for attribute detection.

There have been a number of works tackling the issue of cross-domain clothing detection. The most popular topic is to retrieve similar fashion images from different domains [22–29], many based on deep neural network [24–29]. Most of the works in this area has focused on learning a transformation that aligns the source and target domain representations into a common feature space, or dealing with the cross-domain problem with limited number of labeled datasets available in the target domain [19–21]. We will test our model on different domains, however, we haven’t try to tackle this issue yet.

Early work on trend analysis [30] broke down catwalk images from NYC fashion shows to find style trends in high-end fashion. Recent advance in deep learning enabled more work on this area. [31–33] utilized deep networks to extract clothing attributes from images and created a visual embedding of clothing style cluster in order to study the fashion trends in clothing around the globe.