Image Classification of Fashion Garments using Convolutional Neural Networks

Niharika Ganji, Dwaraka Mannemuddu, Poojitha Mathi[1]*

Abstract

Considering the significant use of classification of garment images in the fashion industry and online advertising, we have focused on solving the garment classification problem in several ways. Gathering initial motivation from the fashion MNIST, we have explored other datasets with similar characteristics. Our work focuses on utilizing the advantages of Convolutional neural networks (CNNs) for image classification. The three state-of-the-art algorithms considered for experimentation are VGG16, ResNet50 and DenseNet201. To expand the applicability of our classification methods, we have implemented Visual Similarity Search, a feature that suggests products from the data set when a new product is given. This feature is particularly useful in today's online fashion industry, where customers can quickly find and purchase products that match their preferences and interests.

Keywords

Neural Networks —- Image Classification —- Fashion Dataset

[1] *Luddy School of Informatics, Computing, and Engineering, Indiana University, Bloomington, IN, USA*

Contents

1. Problem and Data Description

1.1 Problem Statement

Efficient classification of fashion garments has significant use in garment recognition, garment search and garment recommendation. However the task of recognizing the garment from a picture is not trivial for an algorithm as clothes can have many properties and may vary in depth and backgrounds. Considering this as our problem statement, we aim at developing a machine learning model that classifies the garment images with utmost accuracy in the least amount of time.

1.2 Significance

1. Efficient classification of fashion garments help clothing manufacturers and retailers to effectively categorize their products. With products divided into categories, it is easy for the customer to pick the item they are looking for, thereby enhancing customer shopping experience.

2. Classification of fashion garments also helps to predict future demand based on current products and manage inventory accordingly. People are so much into fashion these days that they are willing to spend a lot on their clothing. Different people prefer different sets of clothing as per the situation. So identifying the type of clothing a person prefers will help the clothing manufacturers or retailers to manufacture the clothes according to the customer needs, reducing a lot of time, effort, and money. They can never go out of stock and identify bestsellers to be in the trend.

3. Classification also helps in recommendation of products. Fashion businesses can identify customers interested in a specific product and market to them to increase sales - Targeted marketing.

4. The number of clothes being discarded every year because of customer dislikings reduces with increased customer satisfaction. This in turn reduces the costs to the company and helps in reducing the environmental crisis by the textile industry.

1.3 Background and Novelty

While various classification techniques have been employed over the time, there is always a cap on the accuracy achieved with varying data. Our main focus is to build an efficient classification system with varying data. Our work focuses on developing:

1. A classification system that exhibits utmost accuracy with varying data in the least amount of time.

2. An end-to-end efficient garment suggestion system using classification and recommendation algorithms.

1.4 Data

After exploring quite a lot of data sets, we have considered fashion-small[1] as our working data set. The chosen data set is obtained from kaggle. The data is in the image form. Some of the images from the data set include

Each image is an rgb image of size 256 X 256 pixels. A total of 44,419 images are in the data set which is of the size around 1GB.

Data concerning each image is presented in another csv file named ‘Labels’. The features in the labels.csv include the image id, gender, masterCategory, subCategory, articleType, baseColour, season, year, usage and product display name. All of them are nominal attributes except year. Year is an interval attribute.

1. Image id is the identification number given to each image. This number is different from the index of the labels.csv.

2. The gender feature indicates the intended gender for which the garment is designed. The categorical values in the gender feature include - ‘Boy’, ‘Girl’, ‘Men’, ‘Women’ and ‘Unisex’.

3. Master Category feature represents the broad categorization of garments into Apparel, Accessories, Footwear, Personal care, Free Items, Sporting Goods, and Home.

4. SubCategory feature provides more specific information about the garment category. Examples of some of the categorical values in the subCategory feature include - ‘Topwear’, ‘Bottomwear’, ‘Watches’, ‘Jewelry’ and ‘Bags’.

5. Article Type feature specifies the type of the garment and provides more detailed categorization, including items such as ’Shirt’ , ’Pant’ ’Jeans’, ’Track Pants’ and others.

6. Color represents the color of the garment such as ’Navy Blue’ , ’Green’, ’Yellow’ etc.

7. Season represents the season for which the garment is designed such as ‘Summer’, ‘Winter’ and ‘Fall’.

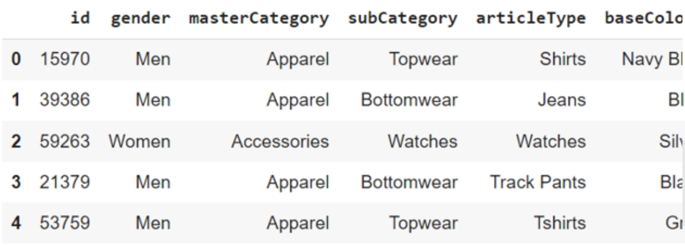
8. Year represents the year in which the garment was manufactured.

9. Usage describes the usage of the garment such as ‘Casual wear’, ‘Ethnic wear’ or so.

[1] https://www.kaggle.com/datasets/bhaskar2443053/fashion-small

10. Product display name feature refers to the name given to the product as it appears on the fashion application.

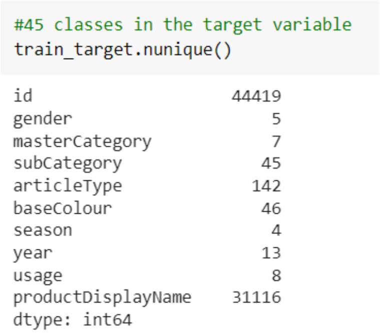
While using a fashion application, we usually focus on the subCategory feature as our preferred categorization. This feature is neither too specific nor too broad in categorization. It strikes a balance between being specific enough to provide relevant information about a garment's category and broad enough to cover a range of products. Hence, we have chosen the subCategory feature as the target variable rather than other features.



	id	gender	masterCategory	subCategory	articleType	baseColo				
0	15970	Men	Apparel	Topwear	Shirts	Navy Bl	[illegible]	[illegible]	[illegible]	[illegible]
1	39386	Men	Apparel	Bottomwear	Jeans	Blue	Summer	2012.0	Casual	Peter England Men Party Blue Jeans
2	59263	Women	Accessories	Watches	Watches	Silver	Winter	2016.0	Casual	Titan Women Silver Watch
3	21379	Men	Apparel	Bottomwear	Track Pants	Bla	[illegible]	[illegible]	[illegible]	nts
4	53759	Men	Apparel	Topwear	Tshirts	Grey	Summer	2012.0	Casual	Puma Men Grey T-shirt

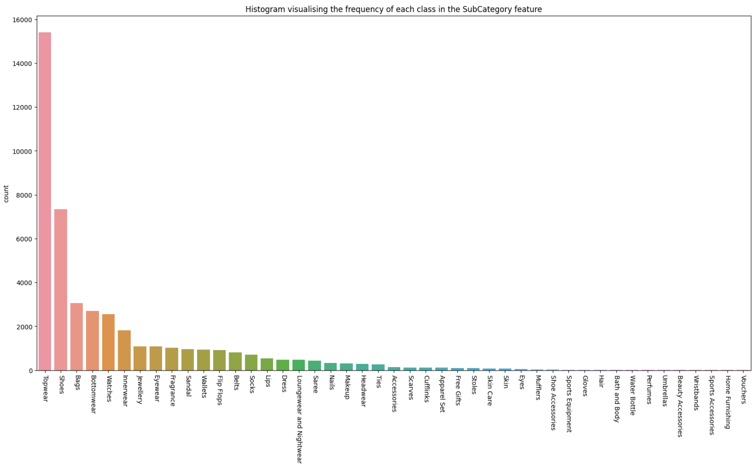
Considering the subCategory feature as the target variable provides enough information for the fashion manufactures to predict demand for specific categories of products. For example, people tend to buy more topwear than bottomwear. This categorization helps manufacturers to produce more topwear than bottomwear.

Now that our chosen target variable is subCategory, information regarding the distribution of the data across the subCategory feature is analyzed. As seen from the picture, the subCategory feature has a total of 45 classes. Some of the classes contain very few records, less than 100. We will tackle this in the next section.



```
#45 classes in the target variable
train_target.nunique()
```

```
id                    44419
gender                    5
masterCategory            7
subCategory              45
articleType             142
baseColour               46
season                    4
year                     13
usage                     8
productDisplayName    31116
dtype: int64
```

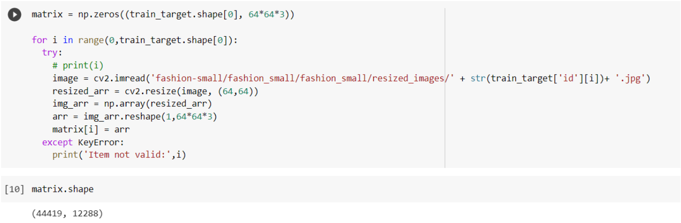


2. Data Preprocessing & Exploratory Data Analysis

2.1 Data Generation

Although the data set fashion-small was mostly clean and simple, it was huge because of the high quality images and could not be uploaded over colab. Hence, we have directly connected colab to kaggle. Using the kaggle API, we could download the data into our collab into a .zip file. After downloading, we extracted the contents of the .zip file, resulting in two files: one containing labels.csv, and other containing a folder of images.

Due to the memory constraints in Colab, we encountered difficulties in flattening the 256X256X3 image data into a matrix of size (44419 , 256 X 256 X 3). Our session crashed multiple times due to insufficient RAM. To overcome this issue, we resized the images to a smaller size and attempted to flatten them. However, even with a size of 128X128, we encountered insufficient RAM errors, and were eventually forced to resize to 64X64 size. Our image data is now in the size of 64X64 and has been flattened into a numpy matrix. (We have also explored other options such as jupyter lab or colab pro, but none of them worked as the data was huge and required a RAM around 24 GB).



```
matrix = np.zeros((train_target.shape[0], 64*64*3))

for i in range(0,train_target.shape[0]):
  try:
    # print(i)
    image = cv2.imread('fashion-small/fashion_small/fashion_small/resized_images/' + str(train_target['id'][i])+ '.jpg')
    resized_arr = cv2.resize(image, (64,64))
    img_arr = np.array(resized_arr)
    arr = img_arr.reshape(1,64*64*3)
    matrix[i] = arr
  except KeyError:
    print('Item not valid:',i)
```

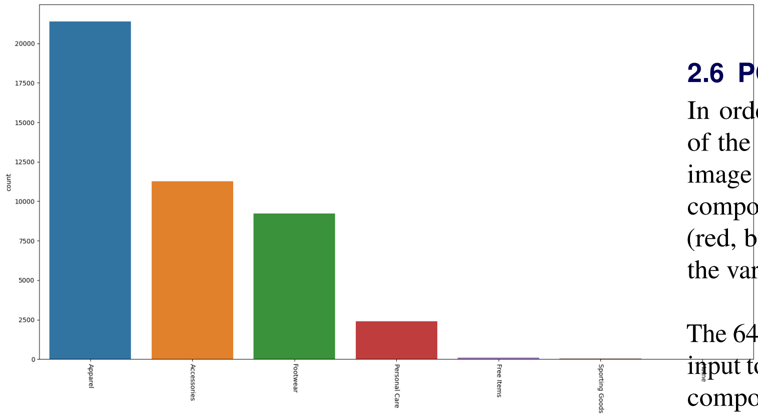
```
[10] matrix.shape

     (44419, 12288)
```

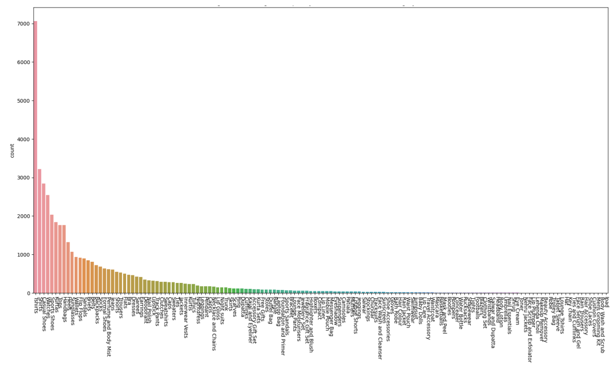
2.2 Handling Missing Values

Fortunately, our dataset did not have any missing labels. Information concerning each of the images is presented in the labels.csv, and we did not encounter any duplicate values in the data. Therefore, we did not employ any measures for removing missing values or duplicate data.

2.3 Exploratory Data Analysis

The distribution across the masterCategory, subCategory and articleType of the labels data is visualised to understand the categorization of the garments. As discussed the subcategory categorization is not too specific nor too broad. The masterCategory is divided into Apparel, Accessories, Footwear, Personal Care, Free Items, Sporting goods and home.



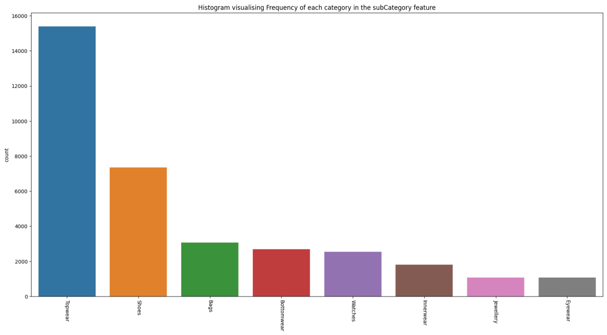


2.4 Data Reduction

During the data reduction process, we explored two approaches: addressing class imbalance and reducing image quality to enhance the model's ability while emphasizing necessary features and ignoring unwanted noise or background.

2.5 Class Imbalance

After careful consideration of the SubCategory feature as target variable, we proceeded to analyze its distribution. As shown in Figure, a significant amount of classes have very few records. We eliminated such records belonging to those classes from the data in order to not influence our model and to achieve a high test accuracy. Records belonging to classes beyond Eyewear have been eliminated from the data, because these classes contained fewer than 1000 records.
After removing the unwanted classes, the distribution of the data across the subCategory feature is as follows:



2.6 PCA - Dimensionality Reduction

In order to reduce the computational time and complexity of the training models, we have reduced the features in the image using Principal Component Analysis (PCA). Principal component analysis has been performed across each channel (red, blue, green) of an RGB image capturing 90 percent of the variance.

The 64X64 (considering a single channel) matrix was given as input to the PCA, and the data was projected onto the principal components capturing 90 percent of the variance. This data was then inverse projected to the same size - 64 X 64 across all three channels. Finally, the individual arrays were stacked to form the complete image.

By using this dimensionality reduction approach, we were able to reduce the computational time required by our model and capture important features for each garment class while ignoring noise and background. A sample PCA-reduced image is presented below. While the quality of the image is reduced, most of the important features for identifying a garment are still present. The background is slightly de-emphasized, and any noise present is blurred. The main object is highlighted the most as shown.

After the exploration of our data pre-processing techniques, we proceed to discuss our methodology and algorithms used.

3. Algorithm and Methodology

Convolutional neural networks (CNN) have been a popular choice for classifying image data, as they can capture significant features of an image by applying smaller filters. Over time, various CNN architectures have been developed to classify garments data with good results. Walking in the same direction, we decided to experiment with three different state-of-the-art (SOTA) CNN models for garment classification and evaluate their performance based on test accuracy. However, before delving into these advanced models, we also considered the idea of experimenting with a basic CNN model to establish fundamental concepts, but our main focus sticks to these SOTA architectures.

Three CNN networks chosen are ResNet50, VGG16 and DenseNet201 because of their established popularity and proven efficiency in classifying fashion data. The simplicity of the dataset due to the dimensionality reduction using PCA will be of high significance in running against the CNN models.

3.1 Train Test Split

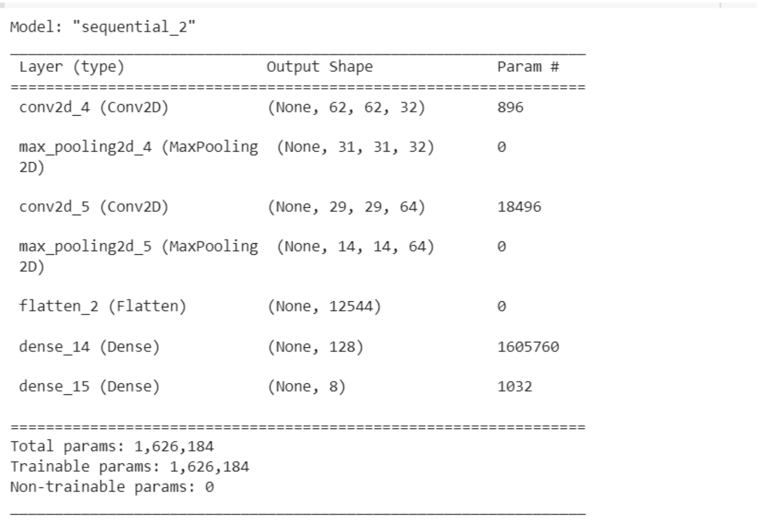
Before proceeding to train our models, we split the data into training and test sets using train test split function from the scikit-learn library. The data set was divided into 90% training set and 10% test set. Since the models used are already pretrained, splitting the data into an 8:2 ratio or any other ratio would not have a major impact on the test accuracy (only a difference by 1% or 2%). We chose the 9:1 ratio as having more data for the model training will benefit the neural network.

3.2 Architectures

Convolutional neural networks (CNNs) use filters, also known as kernels, to extract significant features from an image, such as the edge of a shirt or the shape of a shoe. A basic CNN model typically consists of three types of layers: convolutional, pooling, and fully connected layers. The weights in the convolutional layers (kernels) are trainable and responsible for feature extraction. During model training, these features are learned using forward propagation and backward propagation techniques. The learned features from the convolutional layers are downsampled to reduce dimensionality in the pooling layer. After repeating these convolutional and pooling layers a required number of times, the fully connected layer takes these features as input and generates a vector of class probabilities. An activation function can be used at this stage in the fully connected layer. This process continues until the desired accuracy is achieved or the chosen number of epochs is reached.

3.2.1 Basic CNN model

The standard CNN architecture considered consists of two convolutional layers and two pooling layers one after the other, as seen in the code snippet. The initial fully connected layer inputs the flattened layer, which thereby is passed into the final fully connected layer with units = number of classes for multiclass classification. The model uses categorical cross-entropy as loss function, compiled with adam optimizer and accuracy as evaluation metric.



```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_4 (Conv2D)           (None, 62, 62, 32)        896

 max_pooling2d_4 (MaxPooling  (None, 31, 31, 32)       0
 2D)

 conv2d_5 (Conv2D)           (None, 29, 29, 64)        18496

 max_pooling2d_5 (MaxPooling  (None, 14, 14, 64)       0
 2D)

 flatten_2 (Flatten)         (None, 12544)             0

 dense_14 (Dense)            (None, 128)               1605760

 dense_15 (Dense)            (None, 8)                 1032

=================================================================
Total params: 1,626,184
Trainable params: 1,626,184
Non-trainable params: 0
_________________________________________________________________
```

Although the basic CNN model would showcase good performance, we wanted to experiment using state-of-the-art architectures in order to build an efficient classification system. While all of the convolutional neural networks would consist of the same type of layers and share the same objective across each type of layer, they differ in the number of layers and depth of the neural network.

3.2.2 VGG16

VGG16 is known for its simpler architecture and efficiency in image classification tasks. As the name suggests, VGG16 consists of 13 layers of convolution and max pooling operations and 3 fully connected layers at the end. The number of trainable parameters in VGG16 is significantly higher because of its large fully connected layers at the end.

3.2.3 DenseNet201

DenseNet201 is known for its dense connectivity between the blocks, where each block is connected to all the preceding layers, rather than just the most recent one. This ensures that the network learns not only from the previous layer but from all the feature extraction kernels, allowing for better information flow. This particular neural network is effective when the amount of data is limited due to the large number of layers (201) and dense connections.

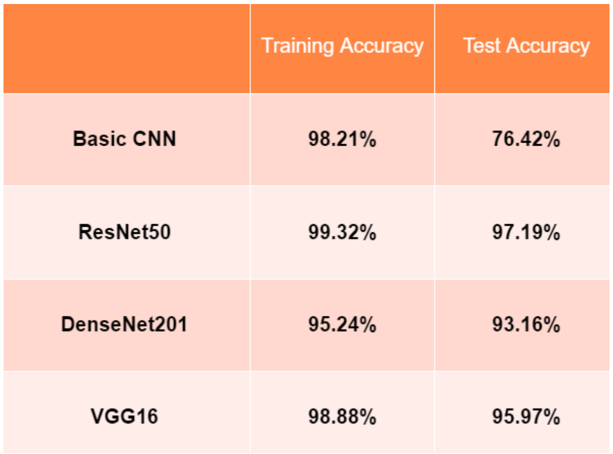
3.2.4 ResNet

Residual Network 50, abbreviated as ResNet50, is a deep CNN network with 50 layers. The unique feature of the ResNet model is the use of residual connections between layers, which enables information to skip over certain layers where no new features are detected.This allows for faster feature extraction and deeper penetration into the neural network, while avoiding unnecessary layers, thereby reducing computational costs.

Due to the complexity of deep learning architectures and our limited knowledge in this area, we have chosen to import pre-built models from the TensorFlow libraries.

4. Experiments and Results

After careful training of the image data with the three models, the training and test accuracies observed are as follows:



	Training Accuracy	Test Accuracy
Basic CNN	**98.21%**	**76.42%**
ResNet50	**99.32%**	**97.19%**
DenseNet201	**95.24%**	**93.16%**
VGG16	**98.88%**	**95.97%**

4.1 Observations

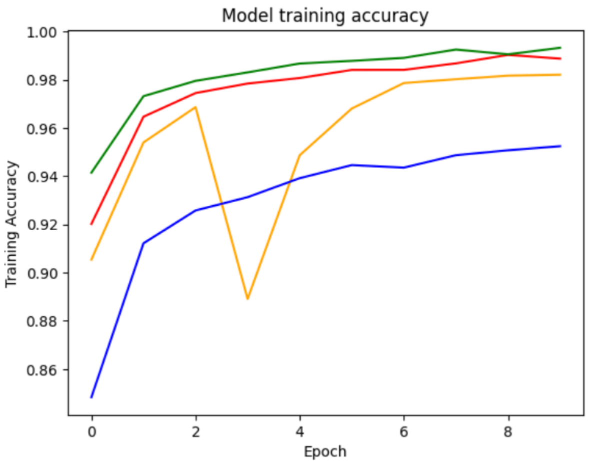
1. The CNN model performed well on the training data, but its performance on the test set was poor compared to the other models.

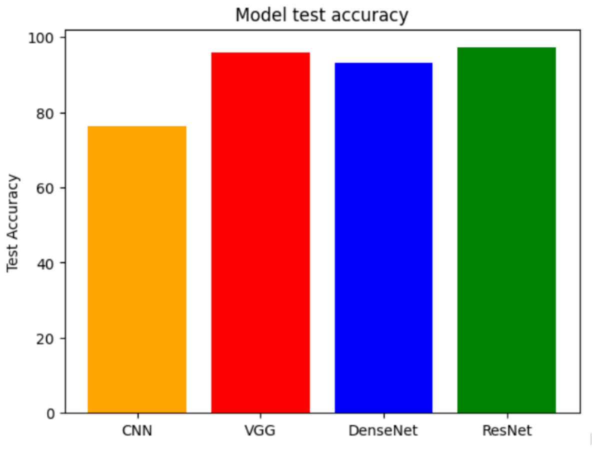
2. Discussing the performance of the three state-of-the-art models:

   (a) VGG16 has demonstrated good training accuracy (98.8%) and test accuracy (95.9%) with a variation of only 3%.

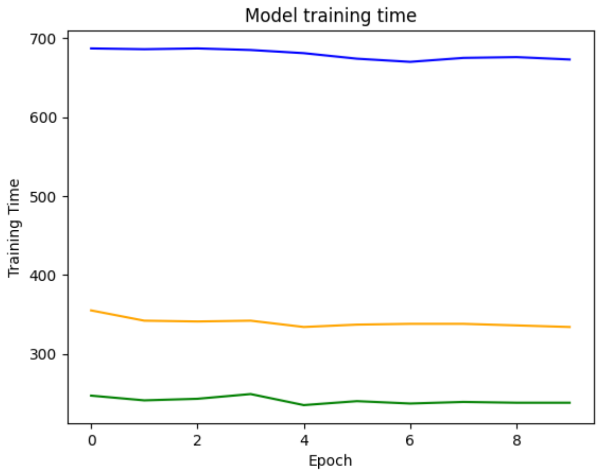
   (b) DenseNet showed less variation between training and test accuracies (2%), but its overall performance was not as good as the other models.

   (c) ResNet has showcased highest training (99.3%) and test accuracy(97.19%).





In addition to accuracy, our objective was to develop an efficient classification system that could achieve high accuracy in the least amount of time. Therefore, we have plotted graphs to compare the amount of time consumed by each model.



The green graph is the time consumed by ResNet50, orange graph represents DenseNet201 and blue graph represents

VGG16. The graph shows that ResNet consumed the least amount of time compared to the other state-of-the-art models. We have only displayed the time consumed by the state-of-the-art models, as they are our main focus.

4.2 Results and Inferences

1. Although the basic CNN model showcased good training accuracy, it could not perform well on the test set. This is due to the overfitting of the model. It could not learn the differentiating features across each class of the garment.

2. ResNet50 was able to achieve higher test accuracy than other models.

   (a) This is mainly because of the sufficient number of layers in ResNet than VGG16 and the residual connections between them. VGG16 was too simple for the data set and it could not learn all the features of the data within the 16 layers.

   (b) Although DenseNet201 had more layers than ResNet, it could not showcase adequate performance because of its speciality only to work well with limited data.

3. ResNet50 consumed the least amount of training time.

   (a) Due to the residual connections between the layers, ResNet50 was able to skip unnecessary layers, thereby reducing computational cost.

   (b) Given the large number of parameters in the fully connected layers of the VGG network, the training time was significantly high.

   (c) DenseNet had comparatively higher training time than ResNet because of the huge number of layers present in DenseNet.

5. Visual Similarity Search

To showcase the practical application of our garment classification system, we implemented a visual similarity search feature. This feature is similar to the concept of Google Lens, where users can input any garment image and obtain five products from the dataset that are visually similar to the given image.

This visual similarity search feature can be useful for fashion e-commerce websites, where customers can upload an image of a garment they like and find similar products to purchase. It can also be used by fashion designers or stylists for inspiration and trend analysis. Overall, the visual similarity search feature adds value to our garment classification system and demonstrates its practicality.

The methodology behind the visual similarity search feature involves leveraging the advantages of K-Nearest Neighbors (KNN) on the PCA-reduced garment images. PCA helps reduce noise and background in images, making the garment object as primary focus.As a result, there is easier mapping between similar images due to shared properties.

For example, if a particular image is a shirt, it will have pixel values only in the region of the shirt object. This makes it easier to differentiate the edges and shape of the object. The KNN algorithm is then applied to the PCA-reduced images to identify the five most visually similar garments in the dataset. The algorithm calculates the Euclidean distance between all the objects and the given object, and returns the five nearest neighbors in terms of distance.

5.1 Feature Creation

The major problem that arose when considering the subCategory feature as the target variable was gender. Although the recommended products were similar to the given image, the recommendation went fundamentally wrong by suggesting men's garments to women or vice versa.

To overcome this problem, we created a new feature named ”labels” by combining the subCategory and gender features. This ensures that the recommended products are similar to the given product in terms of both category and gender. When given an input image outside of the dataset, the following products are the five recommended products. The given image needs to be resized to 64x64 shape to match the original dataset before passing it through the KNN algorithm.
Input Image:

Output images:

6. Deployment and Maintenance

We utilized Google Colab for data preprocessing, exploratory data analysis, model development, and evaluation. The final notebook is attached along with the report. All the code in the notebook can be run directly without the need for the data file, as the data can be imported directly from Kaggle using the Kaggle API.

Convolutional neural networks can be time-consuming and require significant computational resources, so we implemented the trick of saving the models using the h5 extension. This allowed us to save the trained model and load it when needed, reducing training time whenever we needed to evaluate new data.

7. Summary and Conclusions

1. We have observed that ResNet50 achieved the highest accuracy with the least amount of training time, indicating its immense potential in the fashion and e-commerce industry.

2. Basic algorithm like KNN have shown reasonable performance in garment recommendation, opening up new directions of exploration and potentially changing how fashion products could be recommended and marketed to customers.

3. The efficient classification system developed can be utilized by fashion businesses to increase revenue and sales, such as targeted marketing through online advertising, managing inventory, and improving logistics by identifying bestsellers.

8. Future Scope

1. Our classification system could be improved further by using dropout and batch normalization techniques and fine-tuning hyper parameters.

2. Additionally, we could decrease the complexity of the KNN algorithm by predicting the class of the given new image and calculating distances to objects only in that class, instead of calculating distances to the whole data set.

3. While creating the new ”labels” feature, we have only considered gender. Further improvements could be made by considering all other features such as garment type (T-shirts, shirts, jeans), color (red, blue, black), product description (striped, checks, collared), and season of wear and usage (ethnic, summer, tropical). This will lead to more accurate and personalized recommendations for customers.

4. Furthermore, text-based recommendation using classification methods and NLP can also be employed based on the product description given by the customer and the product display name.

5. In addition, collaborative recommendation system can also be developed using reviews and ratings of the products given by the customers.

Acknowledgments

References