Fashion Apparel Classification and Image Search

[1]Riddhi Pawar
Department of Information Technology
Vishwakarma Institute of Information Technology
Pune, India
Riddhi.22020020@viit.ac.in

[2]Anand Shirole
Department of Information Technology
Vishwakarma Institute of Information Technology
Pune, India
Anand.21910161@viit.ac.in

[3]Namrata Thakur
Department of Information Technology
Vishwakarma Institute of Information Technology
Pune, India
namrata.22020010@viit.ac.in

[4]Ishank Sharma
Department of Information Technology
Vishwakarma Institute of Information Technology
Pune, India
ishank.21910221@viit.ac.in

[5] Ekta Mulkalwar
Department of Information Technology
Vishwakarma Institute of Information Technology
Pune, India
ekta.21910986@viit.ac.in

[6]Pravin Futane
Department of Information Technology
Vishwakarma Institute of Information Technology
Pune, India
pravin.futane@viit.ac.in

***Abstract*- In this research, we have attempted to solve the challenge that the e-commerce fashion business is facing. The issue is that the customer may not always know the correct keywords to use when searching for or describing the item he is looking for. To address this problem, a Convolutional Neural Network (CNN) model based on deep learning was created to classify fashion apparel images. The model was trained on the Fashion Product Images (Small) dataset which consists 44k-images. The model was able to reach an accuracy of 85 percent. After that, we deployed this model into a fastapi web application that can categorize various apparel images uploaded by users. We have also trained an autoencoder model to retrieve images that are fairly similar to the user-uploaded image.**

Keywords- neural networks, multiclass classification, convolutional neural network, autoencoder, fashion image similarity, fashion image classification

I. INTRODUCTION

Deep learning models are frequently used to classify problems like image classification, object detection, and many more. Because it is good at dealing with image classification and recognizing difficulties and has improved the accuracy of many machine learning tasks. The convolution neural network (CNN) has indeed evolved into a powerful and widely used deep learning model. It is frequently used in deep learning to analyze visual pictures. The CNN model is made up of several convolutional layers.

Age, social status, lifestyle, and gender are all reflected in clothing in multiple cultures. As a result, outfit plays is a significant role in identifying and judging a person. Predicting garment details in an unlabeled photograph can aid in the search of the most similar fashion products in an e-commerce database. Similarly, a user's favourite fashion photographs can be classified to drive an automated intelligent fashion stylist who can make personalized clothing recommendations on the user's expected style. The most important problems to tackle will vary depending on the requirements of fashion classification. Photographs are divided into two groups. The first having a basic background to display products. The second image having a human or parts of a person wearing items like pants, t-shirts, tie, shoes. As a person wearing multiple things introduces noise, placing objects in front of a plain background reduces semantic noise in images and makes it easier to assign a single label.

The goal of this study is to solve a potentially difficult problem that the e-commerce fashion business is facing. When a consumer wants to buy something, he usually uses text-based search to find it. However, the consumer may not be aware of the appropriate keywords to use while searching for or describing the item. Customers' ability to search for a certain product is hampered as a result, and demand falls. Visual search, which allows shoppers to search for a specific product using its image rather than keywords, can overcome this problem. Customers can use the visual search engine to upload an image of a product and retrieve information about it as well as similar products. Customers can use this to help them place orders online or offline (i.e., shopping from nearby stores) when they don't know the proper keywords but have a visual impression of the items they want to buy. The

customers can easily mimic their favorite influencer/celebrity styles from social media, movies, etc. This project focuses on the classification of fashion and apparel based on its types and attributes as well as retrieving similar images. Among different techniques for image classification ranging from image processing to machine learning approaches, this project uses Convolutional Neural Network (CNN) model. Then the trained model is deployed on fastapi based web application. The distinct feature of this project is that, after uploading the image it displays the exact name and related keywords of the product. This helps users to search for the product using its name or related keywords on multiple online platforms including those not having visual search feature as well as nearby shops.

II. Literature Survey

Among different machine learning techniques that are previously used for the purpose of image classification includes custom neural network models and transfer learning. The Inception Architecture[1], GoogLeNet [1] and Deep Residual Networks (ResNets) [2] were among the architectures and methodologies discussed. Deep learning and CNN are included in the survey, which is covered in detail in [3]. In image classification, many CNN designs have been employed, including LeNet [4], Alex Net [5], Google Net [6], VGGNet [7] and ResNet [8].

K. Chatfield *et. al.* published a study in which they evaluated CNN based methods and traditional shallow feature encoding methods for image classification. The PASCAL VOC-2007 dataset was used to compare the performance. It demonstrated that deep architectures outperform the traditional shallow feature encoding methods [9].

Fengzi Li *et. al.* published a paper proposing neural network models for image classification and image search. They used the Fashion Product Images (Small) dataset having 44k product images. For image classification transfer learning technique was used with pre-trained models. The VGG-19 model performed best for classification with accuracy of 87.1% for gender & master category, 95.7% for sub-category and 84.1% for article type. CNN-based and ResNet-50 based autoencoders and cosine similarity was used to search similar images [10].

Alexander Schindler *et. al.* presented a study of methods to classify fashion and apparel images using different CNN architectures. The proposed model for detecting persons provided 91.07% accuracy. The pre-trained and fine-tuned model produce 88% accuracy for gender prediction and performed better than clean models by 5.9% for VGG-16, 7.9% for InceptionV3 model [11].

Alex Krizhevsky *et. al.* suggested a deep convolutional neural network to classify 1.2 million images into 1000 classes for the ImageNet LSVRC-2010 challenge in a paper. They had error rates of 37.5 percent for the top-1 and 17.0 percent for the top-5 [12].

A. S. Henrique *et. al.* published a paper which deals with classifying outfit images from Fashion-MNIST dataset and proposes four CNN models. Comparing the classification results, highest accuracy is 99.1% using dropout technique [13].

Greeshma K V *et. al.* published a paper that explores the effect of regularization techniques and hyper-parameter tuning on neural networks trained on the Fashion-MNIST dataset. They got maximum 93.99% accuracy by regularization techniques like image augmentation, dropout and tuning hyperparameters including batch size, optimizers, epochs. [14].

S. Bhatnagar *et. al.* published a paper that proposes three CNN models for fashion article images classification which are trained on Fashion-MNIST dataset. They have used residual skip connections and batch normalization for the sake of convenience and speeding up the learning process. The model reports 2% improved accuracy over the other literary systems [15].

Mohammed Kayed *et. al.* published a paper that proposed convolutional neural network based on the LeNet-5 architecture for image classification. The Fashion-MNIST dataset having 70k images of 10 categories was used. The LeNet-5 model was able to achieve accuracy of 98% [16].

Shu Shen published a paper which uses LSTM model to classify Fashion-MNIST dataset images. The Long Short-

Term Memory Networks model results in 88.26 % best precision [17].

Yue Zhang published a paper which evaluates the performance of four convolutional neural networks namely ResNet, AlexNet, LeNet-5 and VGG-16. Among these ResNet was best suited for classifying Fashion-MNIST dataset [18].

Chao Duana *et. al*. published a paper in which they used VGG-11 network with batch normalization layer after the pooling layer for classifying fashion images in fashion-MNIST dataset. The classification accuracy is 91.5% [19].

Table 1. Literature Survey Consolidation

Sr. No.	Title	Algorithm	Accuracy
1	Neural Networks for Fashion Image Classification and Visual Search	VGG-19	95.7%
2	Fashion and Apparel Classification using Convolutional Neural Networks	CNN	88%
3	Classification of Garments from Fashion MNIST Dataset Using CNN LeNet-5 Architecture	LeNet-5	98%
4	Image Classification of Fashion-MNIST Dataset Using Long Short-Term Memory Networks	LSTM	88.26%
5	Image Classification of Fashion-mnist Data Set Based on VGG Network	VGG-11	91.5%

III. METHODOLOGY

A. *Dataset*

For collecting data to train the model for multiclass classification, three main options are available. First is to use readymade data from Kaggle or buy it from third party vendors. Second option involves collecting and annotating data manually. Third one is to write web scraping scripts to collect data i.e. images from internet.

This paper uses Fashion Product Images (Small) dataset [20] which can be easily found on Kaggle. This dataset consists of 44k high resolution color images as opposed to Fashion MNIST dataset which has greyscale images. This makes the Fashion Product Images dataset suitable for real-life business problems. This dataset contains 44,441 jpg pictures with a resolution of 80 × 60 pixels in three colour channels, all of which are identified by numeric-id. The many label attributes describing the product in the styles.csv file are matched to this numeric-id. Master Category, Sub Category, Gender, Article Type, Base Color, Season, Year, Usage, Product Display Name, and so on are some of these attributes.

B. *Data Preprocessing*

The Fashion Product Images (Small) dataset is extremely imbalanced. As a result, data pre-processing is required to overcome the dataset's data imbalance. For training purpose, among all the available attributes of images, Sub Category attribute is used to classify the fashion product images. To avoid biased results by machine learning algorithms while classifying minority classes and false impression of high accuracy of the model, the classes with lesser images are removed.

For further balancing the dataset, Data Augmentation techniques are used to leverage the amount of data available by combining slightly modified copies of current data alongside of newly created synthetic data. When training a machine learning model, it functions as a regularizer and helps to reduce overfitting. [21]. It includes cropping, rotation, zoom, horizontal or vertical flipping and shifting images. ImageDataGenerator class [22] in Keras is extremely helpful for creating tensor image data in batches with real-time data augmentation.

Table 2. Data after pre-processing

Sub-categories	**No. of labels**
Kurti	153
Saree	426
Bottomwear	409
Topwear	416
Loungewear and Nightwear	443
One Piece Dress	460
Total	**2307**

C. Convolutional Neural Network

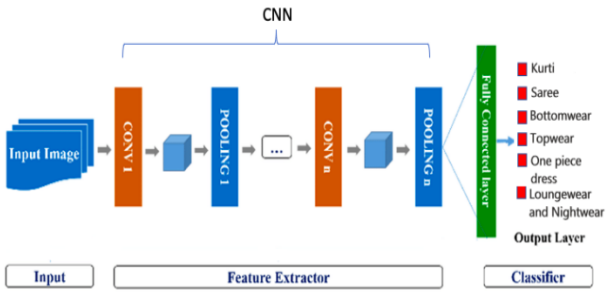

Figure 1. Convolutional Neural Network Working [23]

CNN stands for Convolutional Neural Network. It is a deep learning algorithm that identifies feature set in an input image using kernels. The CNN model depicted in Figure 1, is used to classify fashion apparel. The key advantage of using CNN is that it automatically learns and extracts features from the input data. The Softmax regression allows to distribute the probability of apparel categories among 6 different categories.

Convolutional layer. It is also called as feature extractor layer. It extracts the features of an image received from the input layer. The dot product between a particular region of the input image and the filter is calculated during the convolution operation. This is repeated for the whole image. Mostly, ReLU activation is used in this layer.

Pooling layer. It is used to reduce spatial volume of images while preserving the important characteristics. This layer is mainly used between two convolutional layers and helps to lower the computational power required for data processing. Spatial pooling is also known as down sampling as it reduces size of features extracted by convolution layers. There are three types of pooling available. The first is Max pooling, which chooses the highest possible value. Average pooling, on the other hand, selects the average of all values in the feature map. Sum pooling is the third option, which sums all of the feature map's elements.

Fully Connected Layer. In a neural network, that will always be the last layers. It is not the characteristic of CNN. The image is classified using fully connected layers. It returns a vector as output. The vector consists of probability of the input image to belong to a specific class. The class with high probability value is the predicted class of the image.

D. Convolutional Autoencoder

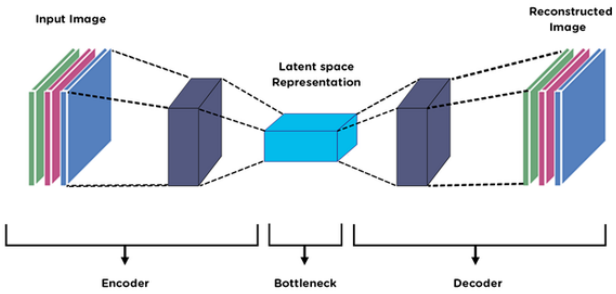

Figure 2. Convolutional Autoencoder Working [24]

Autoencoder is a particular form of feedforward neural network in which the input and output are the same. As shown in Figure 2, it compresses the input data into a lower-dimensional representation afterwards reconstruct the output from it. The input is compressed in the lower-dimensional representation, also known as the latent-space representation. Autoencoders don't require explicit labels to train on, so its considered as an unsupervised learning technique. The autoencoder's output will be close to the input but a bit degraded. Autoencoders are mainly a dimensionality reduction (or compression) algorithm. They can only compress data that is similar to what they were trained on meaningfully. They learn features that are unique to the training data.

Autoencoders are made up of four major components,

Encoder. The model learns to compress the input data into an encoded representation by reducing the input dimensions.

Bottleneck. The input data has been reduced to its smallest possible dimensions. The compressed representation of the input data is stored in this layer.

Decoder. The model learns to reconstruct the data from the encoded representation that is closest to the input data.

Reconstruction Loss. This method evaluates the decoder's performance and compares the output to the original input.

IV. Model Architecture

A. CNN Model

The proposed CNN model's first layer is the convolutional layer, Conv2D which receives the input of shape 60x60x3. This layer has 32 filters, kernel size as 3 with ReLU activation. Its output is received by MaxPooling2D layer, which down samples it. Again, a convolutional layer with configuration similar to previous one and a pooling layer are applied. Then the output is flattened and supplied to the dense layers. Finally, the model contains six dense layers. The output dimensions of Dense layers are 32, 64, 128, 256, 256 respectively with a ReLU activation function. The last dense layer has 6 as output dimension and applies softmax activation to predict multinomial probability distribution.

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 58, 58, 32)	896
max_pooling2d (MaxPooling2D)	(None, 19, 19, 32)	0
conv2d_1 (Conv2D)	(None, 17, 17, 32)	9248
max_pooling2d_1 (MaxPooling2	(None, 5, 5, 32)	0
flatten (Flatten)	(None, 800)	0
dense (Dense)	(None, 32)	25632
dense_1 (Dense)	(None, 64)	2112
dense_2 (Dense)	(None, 128)	8320
dense_3 (Dense)	(None, 256)	33024
dense_4 (Dense)	(None, 256)	65792
dense_5 (Dense)	(None, 6)	1542

Total params: 146,566
Trainable params: 146,566
Non-trainable params: 0

Figure 3. Convolutional Neural Network Model Architecture

B. Convolutional Autoencoder

The latent features of images can be generated using popular visual search techniques like autoencoders. Autoencoders are extremely helpful to identify similar images from bulk data, making them useful for product recommendation as well as image search. We have utilised the Keras functional API to build a flexible model that can handle shared layers as well as many inputs and outputs.

The proposed Convolutional Autoencoder model has 19 layers. It's first layer is Input layer, which receives input of shape 60x60x3. Convolutional layers with filters as 32, 64, 64 respectively, along with kernel size as 3, a LeakyReLU activation layer and BatchNormalization Layer are used three times in the encoder part. After flattening, a Dense layer is used to reduce the embedding feature vector size to 16 which is significantly denser than the original image. After that, there's the decoder part. As illustrated in Figure 4, the encoder and decoder are symmetric. Instead of Conv2D layers in the encoder, the decoder uses Conv2DTranspose layers. Finally, there is a Conv2DTranspose output layer with 3 filters and kernel size of 3, as well as a sigmoid activation layer.

The encoder part of the trained autoencoder model provides the low dimensional representation of each image. This representation is used to identify similar images, by computing the euclidean distance between input image and other available images. The lower the value, the more similar the images will be. In addition, the SequenceMatcher class is used to match the input image's apparel category to that of the relatively similar images in order to verify that they belong to the same category.

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 60, 60, 3)]	0
conv2d_2 (Conv2D)	(None, 60, 60, 64)	1792
leaky_re_lu_2 (LeakyReLU)	(None, 60, 60, 64)	0
batch_normalization_2 (Batch	(None, 60, 60, 64)	256
flatten (Flatten)	(None, 230400)	0
latent_space (Dense)	(None, 16)	3686416
dense (Dense)	(None, 230400)	3916800
reshape (Reshape)	(None, 60, 60, 64)	0
conv2d_transpose (Conv2DTran	(None, 60, 60, 64)	36928
leaky_re_lu_3 (LeakyReLU)	(None, 60, 60, 64)	0
batch_normalization_3 (Batch	(None, 60, 60, 64)	256
conv2d_transpose_1 (Conv2DTr	(None, 60, 60, 64)	36928
leaky_re_lu_4 (LeakyReLU)	(None, 60, 60, 64)	0
batch_normalization_4 (Batch	(None, 60, 60, 64)	256
conv2d_transpose_2 (Conv2DTr	(None, 60, 60, 32)	18464
leaky_re_lu_5 (LeakyReLU)	(None, 60, 60, 32)	0
batch_normalization_5 (Batch	(None, 60, 60, 32)	128
conv2d_transpose_3 (Conv2DTr	(None, 60, 60, 3)	867
decoder (Activation)	(None, 60, 60, 3)	0

Total params: 7,699,091
Trainable params: 7,698,643
Non-trainable params: 448

Figure 4. Convolutional Autoencoder Model Architecture

V. EXPERIMENTATION

A. *Hyperparameter Tuning*

Optimizers. Optimizers are algorithms that change the weights and learning rate of the neural network to minimise losses and generate the most accurate results possible. The Adam optimizer is used to train the CNN model and Convolutional Autoencoder model in this paper.

Batch size And Number of Epochs. The batch size determines how many samples must be processed before the internal model parameters are updated. It is the number of samples from the training dataset that are used to estimate the error gradient. Minibatch Gradient Descent is used in this work, with batch sizes more than one but fewer than the number of samples in the training dataset.

The number of epochs refers to how many times the learning algorithm loops over the entire training dataset. A single loop across the entire training dataset is referred as as an epoch. In an epoch, there are one or more batches. The CNN model and Convolutional Autoencoder model are trained for 100 epochs with callbacks and a batch size of 32. ReduceLROnPlateau, EarlyStopping and ModelCheckpoint are among the callbacks applied to these models to avoid model overtraining.

Activation Function. The activation function determines whether or not a neuron should be activated by calculating a weighted sum and then adding bias to it. An activation function converts the weighted sum of the input into an output from a node or nodes in a layer of the network. In this paper ReLU activation is applied to the hidden layers, and Softmax to the output layer of CNN model. Whereas LeakyReLU activation is applied to the hidden layers, and Sigmoid to the output layer of Convolutional Autoencoder model.

B. *Performance Metrics*

The CNN model's performance across all classes is measured using the accuracy metric. The ratio of the number of right predictions to the total number of predictions made by the model is used to compute its accuracy. It is useful when all of the data classes are roughly balanced.

The best approach to assess an Autoencoder's performance is to measure its reconstruction loss, which evaluates its efficiency to reconstruct the input from its low dimensional embedding.

VI. RESULTS

The data is split into 80-20 for training and validation of both models. With hyperparameter optimization and regularization techniques used, the CNN model was capable of attaining accuracy of 85% during testing the model. The accuracy and loss during the training and validation of the CNN model is as shown in Figure 5 And Figure 6 respectively.

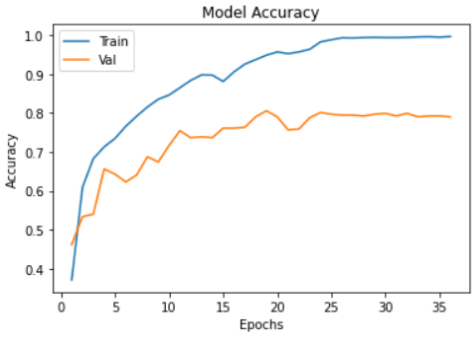

Figure 5. Training and Validation Accuracy

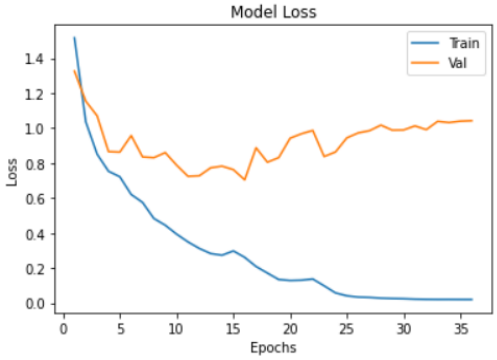

Figure 6. Training and Validation Loss

For the training and validation of Convolutional Autoencoder model the data split is same as CNN model. After training the Convolutional autoencoder model, Figure 7 and Figure 8 shows Top-5 matching images identified during the visual search process. The upper image is the image provided as input, and the five images below it are images that are relatively similar to the original input image based on the computed Euclidean distance and category similarity.

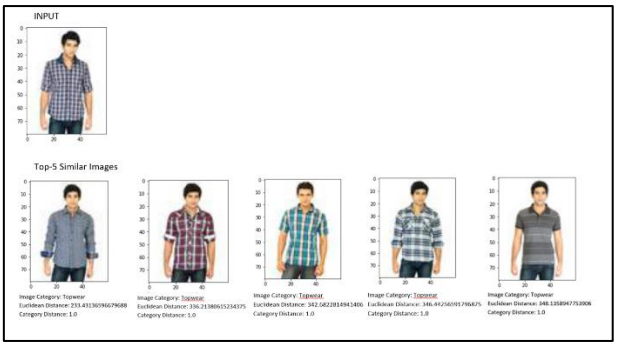

Figure 7. CNN Autoencoder Visual Search Result-1

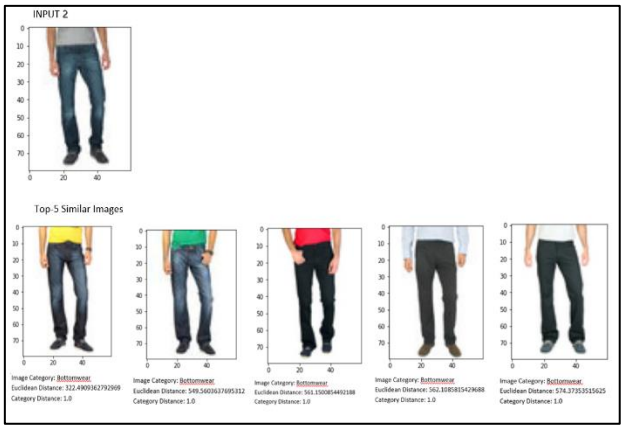

Figure 8. CNN Autoencoder Visual Search Result-2

VII. Conclusion and Future Scope

In this paper, we have used convolutional neural network for image classification and an autoencoder for image search on the Fashion Product Images (Small) dataset. The advantage of utilising CNNs is their ability to capture a two-dimensional image's internal representation. We used the Keras library in Python to implement all of the neural network-based algorithms. With an accuracy of 85% for multiclass classification, the convolutional neural network model performed well. The convolutional autoencoder model, on the other hand, performs well in finding the Top-5 comparable images from the dataset based on Euclidean distance. In future, these models can be tested on larger datasets. Along with it, we may also use natural language techniques to create a model that allows for automated image labelling.

Visual search, which solves the challenge of searching for fashion products when the shopper doesn't know the relevant terms, is a particularly fascinating use case for machine learning models. The image classification feature allows you to classify a given image and then display the apparel's specific name and relevant keywords. It enhances the buying experience by allowing customers to find out the label and category of clothing goods by simply uploading the clothing item's photo.

Machine learning models can also aid merchants in making the most of their time on the site while listing products. Sellers can upload photographs of their fashion items, and image-to-text machine learning algorithms will categorise them automatically. This can aid in the elimination of product labelling errors, which can have a detrimental influence on demand if products aren't displayed correctly in search results. To predict text content from visual data and attributes, CNN models must be combined with natural language processing (NLP) techniques such as Word2vec.

References

[1] Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, 2015
[2] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, 2016
[3] Alom, M. Z., Taha, T. M., Yakopcic, C., Westberg, S., Sidike, P., Nasrin, M. S., Asari, V. K., A state-of-the-art survey on deep learning theory and architectures. Electronics, 8(3), 292. , 2019
[4] LeCun, Y., Bottou, L., Bengio, Y., Haffner P., Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), 2278-2324. , 1998
[5] Krizhevsky, A., Sutskever, I., Hinton, G. E., Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems (pp. 1097-1105), 2012
[6] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Rabinovich, A., Going deeper with convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1-9), 2015
[7] Simonyan, K., Zisserman, A., Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556. ,2014
[8] Li, X., & Cui, Z., Deep residual networks for plankton classification. In OCEANS 2016 MTS/IEEE Monterey (pp. 1-4). IEEE, 2016
[9] K. Chatfield, K. Simonyan, A. Vedaldi, A. Zisserman, Return of the Devil in the Details:Delving Deep into Convolutional Nets, 2014
[10] Fengzi Li, Shashi Kant, Shunichi Araki, Sumer Bangera, Swapna Samir Shukla, Neural Networks for Fashion Image Classification and Visual Search, 2020
[11] Alexander Schindler, Thomas Lidy, Stephan Karner, Matthias Hecker, Fashion and Apparel Classification using Convolutional Neural Networks, 2018
[12] Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, 2012
[13] S. Henrique, A. Fernandes, R. Lyra, V. Leithardt, S. D. Correia, P. Crocker, R. Dazzi, Classifying Garments from Fashion-MNIST Dataset Through CNNs, 2021
[14] Greeshma K V, Sreekumar K, Hyperparameter Optimization and Regularization on Fashion-MNIST Classification, 2019
[15] S. Bhatnagar, D. Ghosal, M. H. Kolekar, Classification of fashion article images using convolutional neural networks, 2017
[16] Mohammed Kayed, Ahmed Anter, Hadeer Mohamed, Classification of Garments from Fashion MNIST Dataset Using CNN LeNet-5 Architecture, 2020

[17] Shu Shen, Image Classification of Fashion-MNIST Dataset Using Long Short-Term Memory Networks, 2018
[18] Yue Zhang, Evaluation of CNN Models with Fashion MNIST Data, 2019
[19] Chao Duana, Panpan Yinb, Yan Zhic, Xingxing Li, Image Classification of Fashion-mnist Data Set Based on VGG Network, 2019
[20] Fashion Product Images (Small), https://www.kaggle.com/paramaggarwal/fashion-product-images-small
[21] Data Augmentation, https://en.wikipedia.org/wiki/Data_augmentation
[22] ImageDataGenerator in Keras, https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/image/ImageDataGenerator
[23] Convolutional Neural Network Layers Working, https://towardsdatascience.com/convolutional-autoencoders-for-image-noise-reduction
[24] Convolutional Autoencoder Working, https://www.i2tutorials.com/explain-about-auto-encoder-details-about-encoder-decoder-and-bottleneck/