Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 7549397, 13 pages
https://doi.org/10.1155/2022/7549397

Research Article

Deep Learning Image Classification for Fashion Design

A. Vijayaraj,[1] P. T. Vasanth Raj,[2] R. Jebakumar,[3] P. Gururama Senthilvel,[4] N. Kumar,[5] R. Suresh Kumar,[2] and R. Dhanagopal[2]

[1]*Department of Information Technology, Vignan's Foundation for Science, Technology and Research, Guntur, India*
[2]*Centre for System Design, Chennai Institute of Technology, Chennai, Tamil Nadu, India*
[3]*School of Computing, Department of Computer Science and Engineering, SRM Institute of Science and Technology, Kattankulathur, India*
[4]*Computing Sciences and Engineering, Galgotias University, Greater Noida, Gautam Budh Nagar, Uttar Pradesh, India*
[5]*Department of Biomedical Engineering Technology, Jimma University, Ethiopia*

Correspondence should be addressed to N. Kumar; kumar.nachimuthu@ju.edu.et

Received 13 April 2022; Revised 17 May 2022; Accepted 27 May 2022; Published 14 June 2022

Academic Editor: Mohammad Farukh Hashmi

Fashion has always been an essential feature in our daily routine. It also plays a significant role in everyone's lives. In this research, convolutional neural networks (CNN) were used to train images of different fashion styles, which were attempted to be predicted with a high success rate. Deep learning has been widely applied in a variety of fields recently. A CNN is a deep neural network that delivers the most accurate answers when tackling real-world situations. Apparel manufacturers have employed CNN to tackle various difficulties on their e-commerce sites, including clothing recognition, search, and suggestion. A set of photos from the Fashion-MNIST dataset is used to train a series of CNN-based deep learning architectures to distinguish between photographs. CNN design, batch normalization, and residual skip connections reduce the time it takes to learn. The CNN model's findings are evaluated using the Fashion-MNIST datasets. In this paper, classification is done with a convolutional layer, filter size, and ultimately connected layers. Experiments are run with different activation functions, optimizers, learning rates, dropout rates, and batch sizes. The results showed that the choice of activation function, optimizer, and dropout rate impacts the correctness of the results.

1. Introduction

As the largest category in the e-commerce industry, the clothing category demands this technology more urgently. At present, business-to-consumer (B2C) e-commerce is on the rise; websites such as Amazon, Flipkart, and Myntra for clothing shopping online have become a common occurrence. Consumers' perceptions of product comprehension are influenced directly by online photos, which are frequently employed as visual signals to grab their attention and influence their purchasing decisions in recent years; the question of how to effectively depict garment pieces from web photos has gained interest in the study. One of the main reasons is that it allows you to look at fashion from a different angle and investigate more intelligence in fashion clothes, such as obtaining comparable or identical fashion goods from an e-commerce website. On the other hand, clothing images come in a variety of styles, and various clients have a deeper grasp of the approach. On B2C e-commerce platforms, different clothing brands may show different types of images of clothes [1].

Its most heavily used deep neural network is a convolutional neural network (CNN). Convolutional neural networks (CNNs) use self-optimizing artificial neurons that work similarly to convolutional neural networks (ANNs). CNN has three layers: a convolutional layer, a pooling layer, and a fully connected layer. Convolutional layers have as their main purpose the generation of features for an image by sliding a smaller matrix (a filter or kernel) over the entire image and generating feature maps. Reducing the feature maps kept the most critical features of the data. To continue to the output layer, which will output the prediction, we

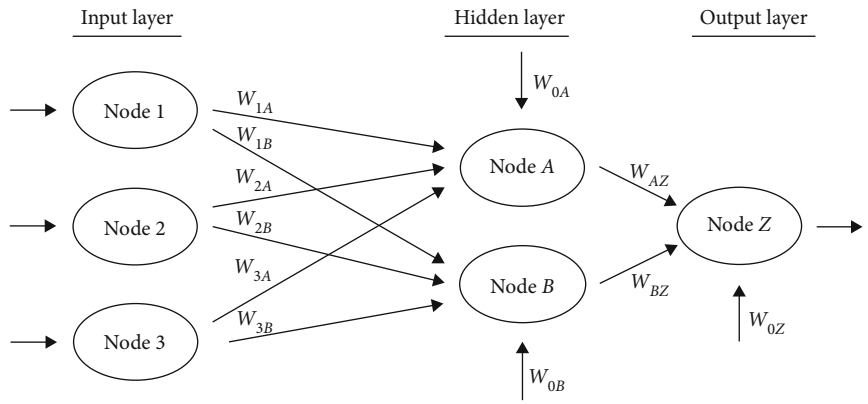


Figure 1: Architecture of ANN.

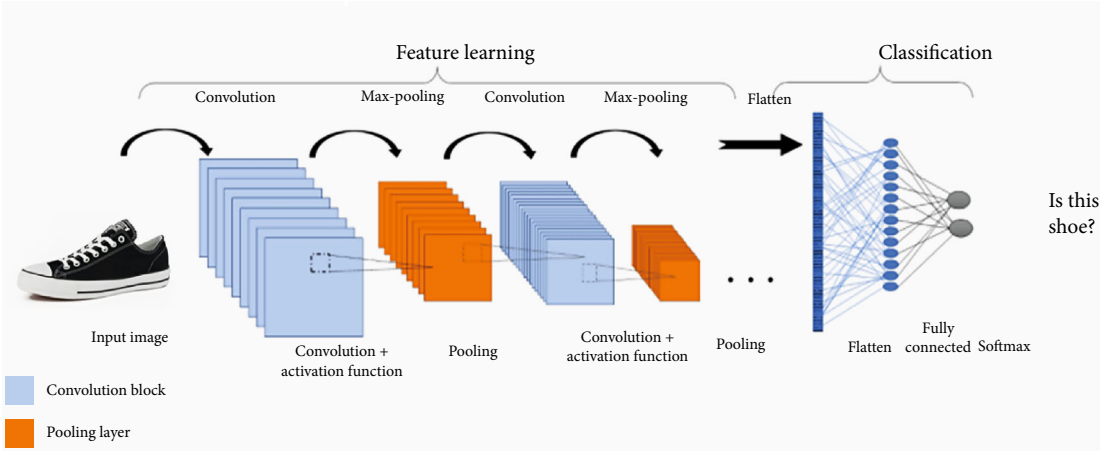


Figure 2: CNN for image classification.

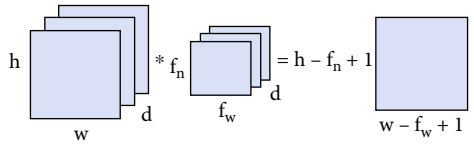


Figure 3: Multiplies the image kernel or filter kernel.

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$$

5 × 5 - Image matrix 3 × 3 - Filter matrix

Figure 4: Image Matrix multiplied by Filter matrix.

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 4 & 3 & 4 \\ 2 & 4 & 3 \\ 2 & 3 & 4 \end{bmatrix}$$

Convolved feature

Figure 5: Features Map Result.

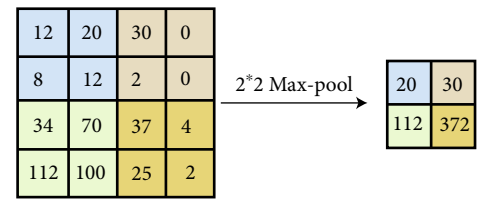


Figure 6: Max pooling.

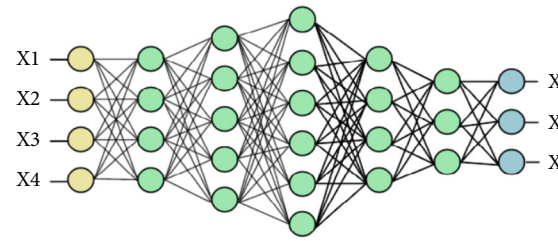


Figure 7: Fully Connected Layer.

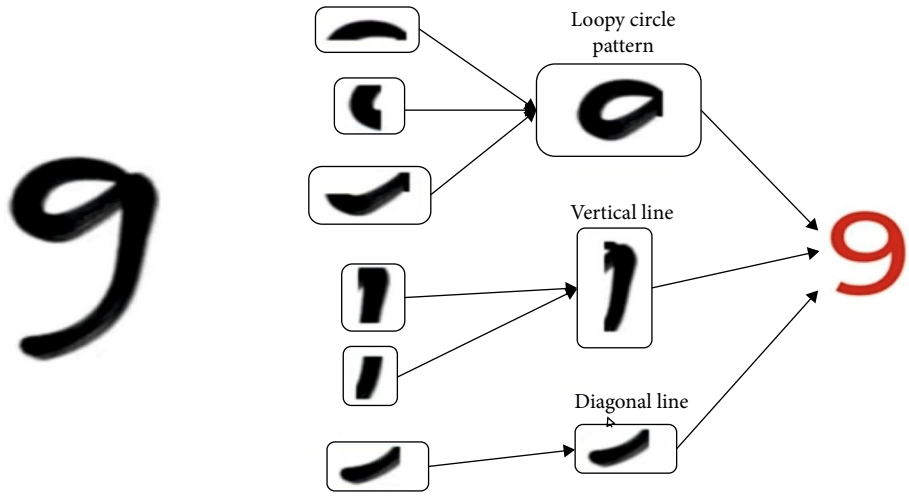


Figure 8: Handwritten Nine Symbols.

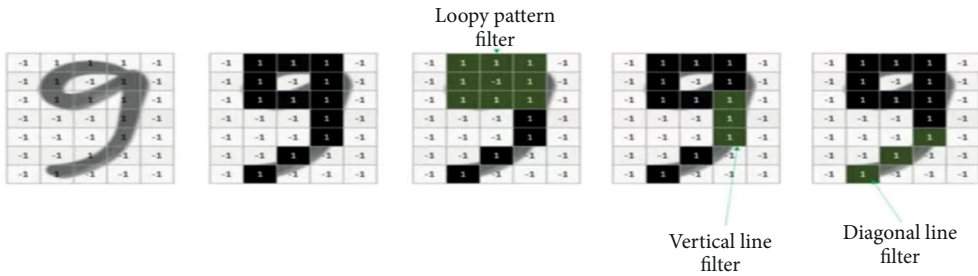


Figure 9: Filters Formation.

flatten the previous layer's input matrix by linking the bottom-most neurons in the previous layer to the top-most neurons in the next layer. The model's architecture will allow it to train using fewer datasets, which reduces the amount of parameter learning required. Time-delayed neural networks, or CNNs, utilize shared weights to speed up processing. CNN has been very good at machine learning applications, and it improves the accuracy and efficiency of applications.

In a classification query, feature extraction is a more critical task for image recognition [2]. The CNN was used to learn picture representation and reuse it on large-scale data sets for classification. The Mode classification, in which labels characterize costumes, is one of the most challenging multi-class classification tasks. The intricacy of this multi-class issue for classification stems from the breadth of the features of clothes and the depth of the categorisation. Because of the complexity of the deep, different labels/classes have comparable characteristics. This study aims to improve the Fashion-MNIST Dataset [3], which contains 70,000 images, the performance of the fashion classification problem. When it comes to fashion classification, there are a few things to keep in mind. For example, extending patterns can easily deform clothing. Second, some clothing may be considered different depending on the viewpoint, while others may be deemed the same. Third, certain garments are difficult to recover due to their tiny size. Fourth, pictures may be shot in a variety of settings, such as varied perspectives, light, and noise. Fifth, various garments, such as pants and tights, have comparable characteristics and can be fuzzy. Sixth, whether a clothing image is just a snapshot of the garment or a photograph of the model wearing it differs. As a result, an algorithm that can perform well in multi-class fashion categorization is essential. This research gives a quick summary of the various CNN models for fashion-MNIST categorization. Nonetheless, CNN's significant contribution will be to tackle the multi-class fashion categorization problem [4].

The remaining portions of the paper are structured as follows: In Section 2, will discuss the related work. Following this, Section 3 proposed work of CNN. Section 4, discusses the Material & Method of the proposed work. Section 5, discussed the Data Set used. Section 6, conferred experimental results and discussion of the work. Section 7, discussed Conclusion & Future work of the proposed work.

2. Related Work

Deep learning and CNN have also been thoroughly scrutinized [5]. Two CNN architectures utilised in image recognition are VGGNet [6] and ResNet [7]. All of these architectures are competing to interpret and identify

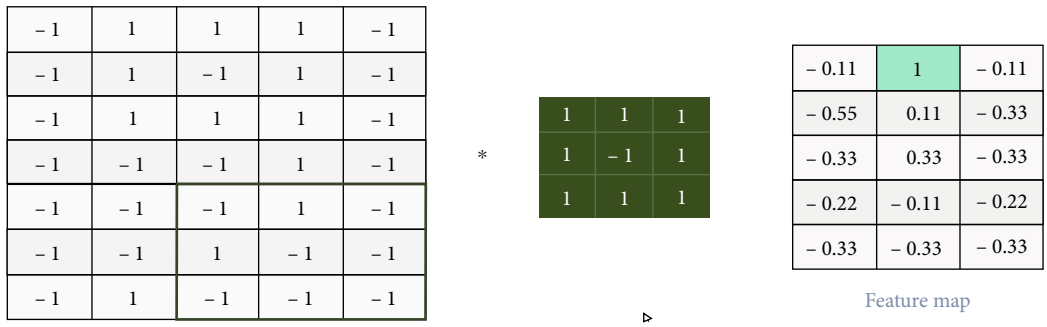


Figure 10: Feature Map.

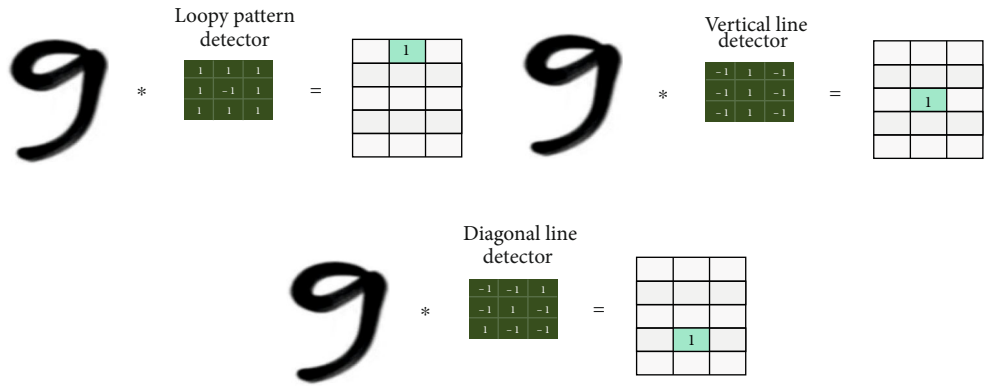


Figure 11: Feature Map Detection.

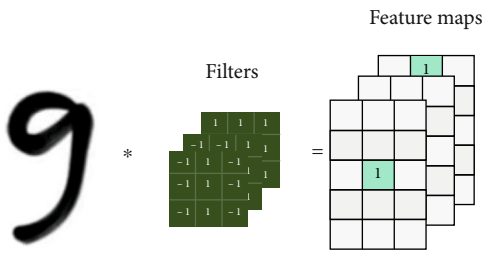


Figure 12: Combining Different Feature Maps.

pictures accurately. The use of neural networks in metric learning has also helped with image similarity estimates and visual search. The Fashion-MNIST datasets contain 700,000 annotated real-life pictures for image classification [5]. We'll go through some of the work that's been done with the Fashion-MNIST dataset and propose a model for categorisation in this session of fashion article images; pictures from the fashion-MNIST dataset can be identified using convolutionary neural network-based deep study architectures. In order to speed up the learning process and enhance the categorization of fashion articles for the Fashion-MNIST dataset, three alternative, batch normalisation, and residual skip connections, CNN architectures were developed. Two-layer CNN, batch normalization, and skip connections were used to attain 92.54% accuracy. The CNN-SVM and CNN-Softmax models on the Fashion-MNIST dataset achieved roughly 90.72% and 91.86%, respectively. In Zhong's study [8, 9], the random erasing technique is used to train a CNN. Zhong's study randomly selects a rectangular region in a picture and then erases its pixels with random values. This method provides training pictures with various degrees of occlusion, reducing the risk of overfitting and making the model occlusion-resistant. On the Fashion-MNIST dataset, test alternative topologies with favorable results on various recognition tasks. In a deep neural network, Agarap [8] utilizes ReLU instead of activation as a classification function. He used two classification functions, softmax and ReLU, in conjunction with convolutional models and feed-forward neural networks (FFNNs). DL-ReLU models are tested against DL-Softmax models on the MNIST, Fashion-MNIST, and Wisconsin Diagnostic Breast Cancer datasets to see which model yields better predictive performance. On a team level, CNN-ReLU had the most accurate forecasts. The faster CNN-Softmax converged, the more accurate class predictions it produced on average. On the MNIST dataset, CNN with softmax activation function achieves 95.36 percent accuracy. A hierarchical convolutional neural network was built utilizing the Fashion-MNIST dataset using VGGNet [10]. A 128-batch SGD model's findings are displayed, using learning rates of 0.001, 0.0002, and 0.00005. The acquired training and testing are both 100% and 93.52 percent accurate. CNN, with the relu activation function, on the other hand, has an accuracy of 91.74 percent.

An approach to adding and comparing Sigmoid features, ELUs, and ReLUs for missing benchmarks in the Fashion-MNIST dataset was provided by Michael McKenna [11]. A

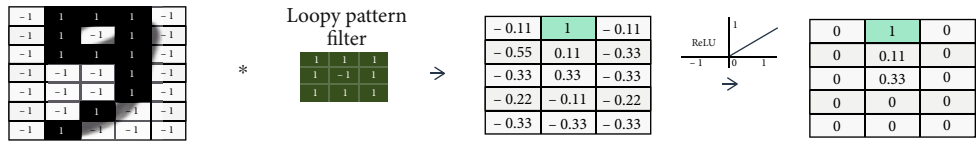
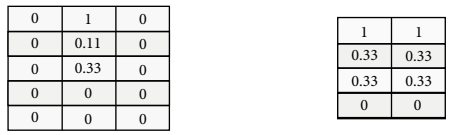


Figure 13: Applying Activation Function.



2 By 2 filter with stride = 1

Figure 14: Pooling.

benchmark is constructed to test whether the Fashion-MNIST missing multi-layer non-convolutional neural feed-forward networks function as claimed. Consequently, it is necessary to uncover the capability of current activation features (compared with ELU, ReLU, and sigmoid). Fashion-MNIST has a few notable features because it provides various benchmarks. However, they all use shallow architectures and only marginally support ELUs. When the network was partially trained, the output was considerably less insufficient than convolutional benchmarks, highlighting some of the benefits of ELUs and ReLUs. Shunning Shen [12] utilized Short-Term Memory Networks to create a model for picture categorization that leverages the Fashion-MNIST dataset, saving time and enhancing accuracy. Han determined that the LSTM model had the highest accuracy (88.26 percent). [13] constructed a good benchmark dataset that considers all of MNIST's accessible features, such as its short size and straightforward encoding. The Fashion-MNIST format is converted to the format of the MNIST dataset.

As a consequence, they may work with the original MNIST dataset using any machine learning system. Even though MNIST has been trained in over 99.7% of cases, Fashion-MNIST is more complicated than just digitizing data from MNIST. Jmour et al. [14] trained the CNN for a traffic sign categorization system using the ImageNet dataset. CNN is used to learn features and categorize RGB-D pictures. A lot of factors influence the accuracy of training results. Hu et al. [15] recommended testing a five-layer CNN architecture on a large number of hyper spectral image datasets to categorize hyper spectral pictures directly in the spectral domain, which yielded better results. In [16, 17] propose two approaches for learning various combinations of fundamental activation functions: identity function, ReLU, and tanh. [18] study tests how successfully CNNs recognize real-time video objects. Alex Nets, GoogLeNet, and ResNet50 are convolution neural networks for object recognition. CNNs may be tested using many image datasets. Convolutional neural networks use ImageNet, CIFAR10, CIFAR100, and MNIST. AlexNet, GoogleNet, and ResNet50 are compared. Utilized ImageNet, CIFAR10, and CIFAR100 since a single data set does not reveal a network's capabilities and constraints. Videos are tested, not trained. GoogLeNet and ResNet50 are more accurate than Alex Net, according to our analysis. Analyse because CNNs function differently across object types. [19] Spam contains malicious links, programs, counterfeit accounts, false news, reviews, and rumors. Spam text detection and management increase social media security. This study looks at how to identify and categorize spam text on social media. Machine Learning, Deep Learning, and text-based spam detection and classification algorithms are discussed in this article [19].

2.1. An Artificial Neural Network (ANN). An Artificial Neural Network (ANN) is a type of computer designed to replicate how the human brain analyses and processes information. Throughout the science of artificial intelligence, an ANN attempts to replicate the network of neurons that make up a human brain so that computers can comprehend information and make decisions in a human-like manner. Computers are programmed to act like interconnected brain cells to create an artificial neural network. The human brain contains approximately 1000 billion neurons. Each neuron has several association points ranging from 1,000 to 100,000. Data is stored in the human brain in such a way that it may be spread, and we may pull multiple pieces of this data from our memory at the same time if needed. The human brain, we may say, is made up of incredible parallel processors [20].

From Figure 1. The ANN architecture is made up of three layers: input, hidden, and output. In the input layer, it accepts different types of input. Between the input and output layers, there is a hidden layer. It does all of the calculations necessary to uncover hidden features and patterns; input passes through a series of changes in the hidden layer, resulting in an output delivered through this layer. The artificial neural network, which includes a bias, computes the weighted sum of the inputs. This computation is expressed using a transfer function.

$$\sum = \mathrm{wi} * \mathrm{Xi} + \mathrm{b} \tag{1}$$

The weighted total is used as an input to an activation function to generate the output. Activation functions determine if a node should fire or not. The only ones who make it to the output layer are those who are fired. Depending on the work at hand, there are a variety of activation functions that can be used.

3. Proposed Work

In image processing, classification, and segmentation, deep feedback mechanisms of an artificial neural network, such as convolutional neural networks, are used. Artificial neural

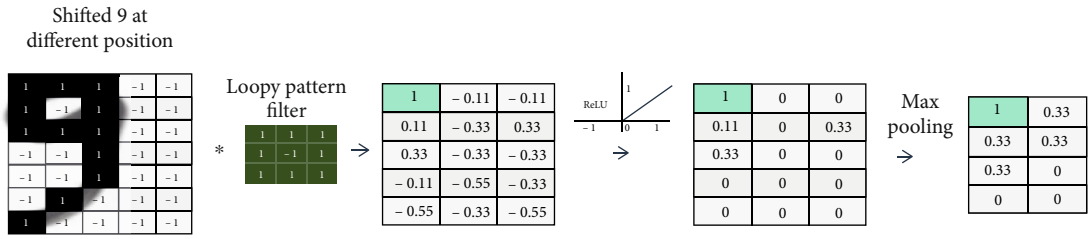


Figure 15: WorkFlow of Feature Extraction.

Label	Description	Examples
0	T-shirt/top	
1	Trouser	
2	Pullover	
3	Dress	
4	Coat	
5	Sandals	
6	Shirt	
7	Sneaker	
8	Bag	
9	Ankle boots	

Figure 16: Fashion-MNIST Dataset.

Figure 17: Confusion Matrix.

networks (ANNs) imitate human brain activities in ways similar to the human brain. ANNs are structures made up of a network of computer nodes that collaborate to process complicated data inputs and optimise the final output in a distributed manner. The construction of CNNs is comparable to conventional ANNs since they are made up of neurons

	Precision	Recall	F1-score	Support
T-shirt/top	0.85	0.84	0.85	1000
Trouser	0.98	0.98	0.98	1000
Pullover	0.84	0.83	0.84	1000
Dress	0.93	0.86	0.89	1000
Coat	0.81	0.87	0.84	1000
Sandal	0.98	0.98	0.98	1000
Shirt	0.72	0.73	0.72	1000
Sneaker	0.94	0.97	0.95	1000
Bag	0.97	0.98	0.97	1000
Ankle boot	0.98	0.95	0.96	1000
Accuracy			0.90	10000
Macro avg	0.90	0.90	0.90	10000
Weighted avg	0.90	0.90	0.90	10000

Figure 18: CNN Classification Report.

Table 1: Test Accuracy.

Model(method)	Test accuracy
CNN using Adam	94.52
ANN using Adam	90.12
CNN using RMS	88.72
ANN using RMS	87.51
CNN using SGD	86.43
ANN using SGD	84.21
CNN using ADAGRAD	81.80
ANN using ADAGRAD	79.56
CNN using SGDM	79.93
ANN using SGDM	77.93

organized into optimizing neurons and obtain inputs and apply linear functions to produce a single scalar product with a non-linear function. Multi-layer perceptron networks are CNNs (convolutional neural networks) that have been normalized, resulting in regularized versions with all neurons in one layer connected to all neurons in the next layer and with a class-specific loss function added to the end (the weight). All of the typical ANN-specific tips and tactics are passed down to CNN. Convolutional, pooling, and completely connected layers are among the three types of layers used in CNN. Each of these levels performs a separate function on the input data. To pull out features from the input image, filters are utilized. The pooling layer has several functions, including maximum pooling and average pooling. The procedures of the pooling layer have max and average extraction processes. The procedures of the pooling layer have max pooling and average pooling. The maximum value is extracted in the filter region as well as the average value in the filter region. The completely linked layer collects data from feature maps and applies it to a final categorization. Figure 2. CNN for image classification One of the labels in the Fashion Mnist dataset is classified as a Convolutional Neural Network. This has to be done in feature extraction and classification, which has to be gone through filters, Activation Functions Pooling, etc. All of these are explained in detail below.

3.1. Convolution Layer. The first to extract features from an input image is the convolution layer. The convolutional layer preserves the relationship between pixels by using only a tiny input square for image information. It is a two-step process: First, there needs to be an image matrix. Second, this image matrix is processed with a kernel or Filter in Figure 3.

(i) The image matrix has a width, height, and depth of $h \times w \times d$

(ii) The Filter's size is $fh \times fw \times d$

(iii) The output's length, width, and height are $(h\text{-}fh+1) \times (w\text{-}fw+1) \times 1$

A 3x3 filter matrix is used to make a 5x5 image whose pixel values are 0, 1 in Figure 4.

The output of the convolution of a $5*5$ image matrix multiplied by a $3*3$ filter matrix is called "Features Map" shown in Figure 5.

To blur a picture, apply a convolution and sharpen it, apply a convolution. And to detect edges, apply a convolution. Convolutions feature extraction benefits from strides and padding.

3.1.1. Strides. How the filter convolves around the input volume is determined by its stride. In the first scenario, the filter convolved around the input volume by moving one unit at a time. The stride refers to how much the filter shifts. Astride is commonly chosen to create an integer rather than a fractional volume. The stride is the number of pixels that are shifted across the input matrix. The filters are shifted one pixel at a time when the stride is set to 1, and two pixels at a time when the stride is set to 2.

3.1.2. Padding. Padding is a term used in convolutional neural networks to define how many pixels are added to an image by the CNN kernel when it processes it. If the padding in a CNN is set to 0, the value of every pixel added will be zero. Padding is essential in the construction of a convolutional neural network. If we shrink the image and use a neural network with hundreds of layers, we will end up with a small image that has been filtered.

3.2. Pooling Layer. In image preprocessing, the pooling layer is very critical. Pooling reduces the number of parameters when the photos are too huge. The act of "downsizing" the picture generated by the preceding layers is known as "pooling." Reducing the density of an image by making it smaller is equivalent to reducing the image's pixel density. By reducing the number of distinct features of each map, spatial pooling, also known as downsampling or subsampling, reduces dimensionality without losing valued data. Spatial pooling includes:

3.2.1. Max Pooling. Maximum pooling is a discretization procedure that samples the value. One of the critical functions of the binning sub-region minimization is to constrain assumptions about features discovered in the binned region. By applying a max filter to the sub-regions of the initial representation that are not overlapping, it is possible to obtain

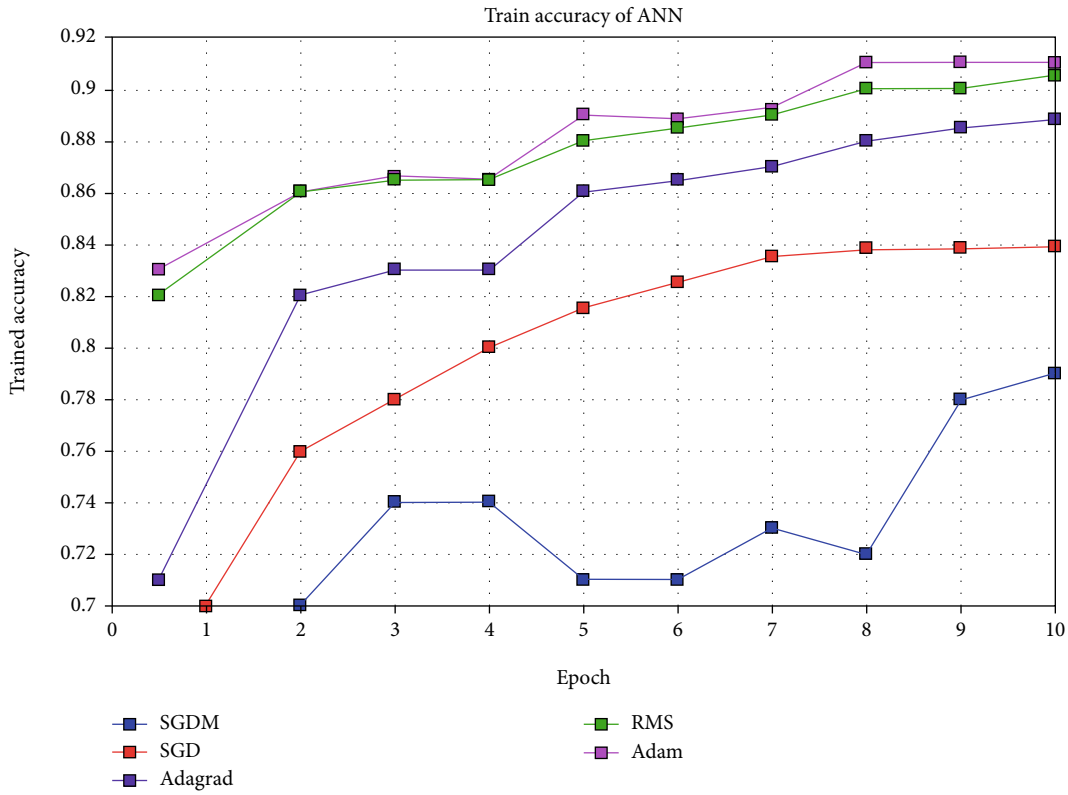


Figure 19: Train Accuracy of ANN.

the greatest possible pooling. Figure 6 shows the procedure of Max pooling.

3.3. Average Pooling. Average pooling is a down sampling process for feature maps that first determines the average value for patches of a feature map and then uses it to generate a down-sampled feature map. Once a convolutional layer is applied, it's commonly used. The input will be split into rectangular regions and the average values of each sector will be calculated. Calculating the average for each patch of the feature map is what average pooling entails. This means that each of the feature map's 2x2 squares gets down sampled to its average value.

3.4. Sum Pooling. Feature map down sampling, which reduces the pixel size of images, is made possible by the Sum Pooling. In other words, sum pooling is a form of a max function, but instead of returning the maximum value, it takes the sum of all the input values. Although they use the same sub-region as their max function, mean and sum pooling choose to use their sub-max regions instead of using the max function.

3.5. Fully Connected Layer. Every input from one layer is connected to each activation unit of the following layer, and the layers are completely coupled. A standard machine learning model almost always incorporates fully connected layers at the end that take in all of the information presented by earlier layers and ultimately deliver the final result. At the output of the convolutional layers, the data features are represented at a high level. You can use a fully connected layer to learn non-linear combinations of these features while keeping the output layer flattened and connected to the output layer. The fully connected layer gets input from other layers and converts it to a vector before sending it. The output will be divided into the desired number of classes by the network.

Figure 7 Shows the Different Nodes Inter-Linked to each other to categorize to display the result.

4. Material and Method

To Understand Better about the model, let us consider the Below Example.

According to Figure 8, the handwritten image will be divided into convolutional filters for feature extraction; if the image is large, pixels filters will be used to divide it into different parts. In the above Fig., the image has been divided into three filters. After extracting features, it is done with Activation Functions and Pooling. The extracted features are then in 2D or 3D Arrays. With the help of the flatten layer, it should be converted into 1Dimensional Arrays, and the dense layer should help one neuron connect to another with fully connected layers. This should classify the image.

4.1. Filters Formation. The Filters Formed wherever the object is identified it filled with some value. Filters can detect the patterns such as edges in an image by detecting the values of an image. A convolutional layer's filters typically

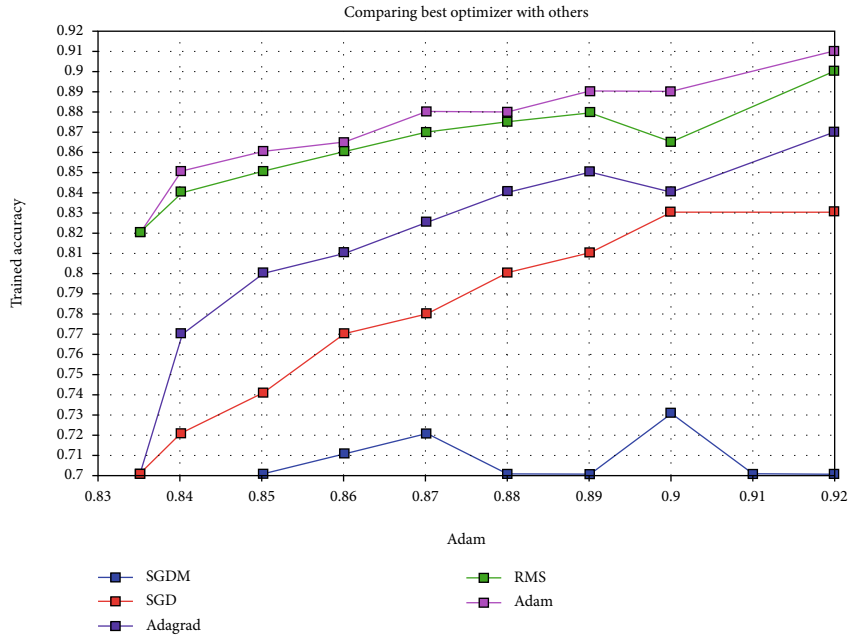


Figure 20: Comparing Best Optimzer with existing methods.

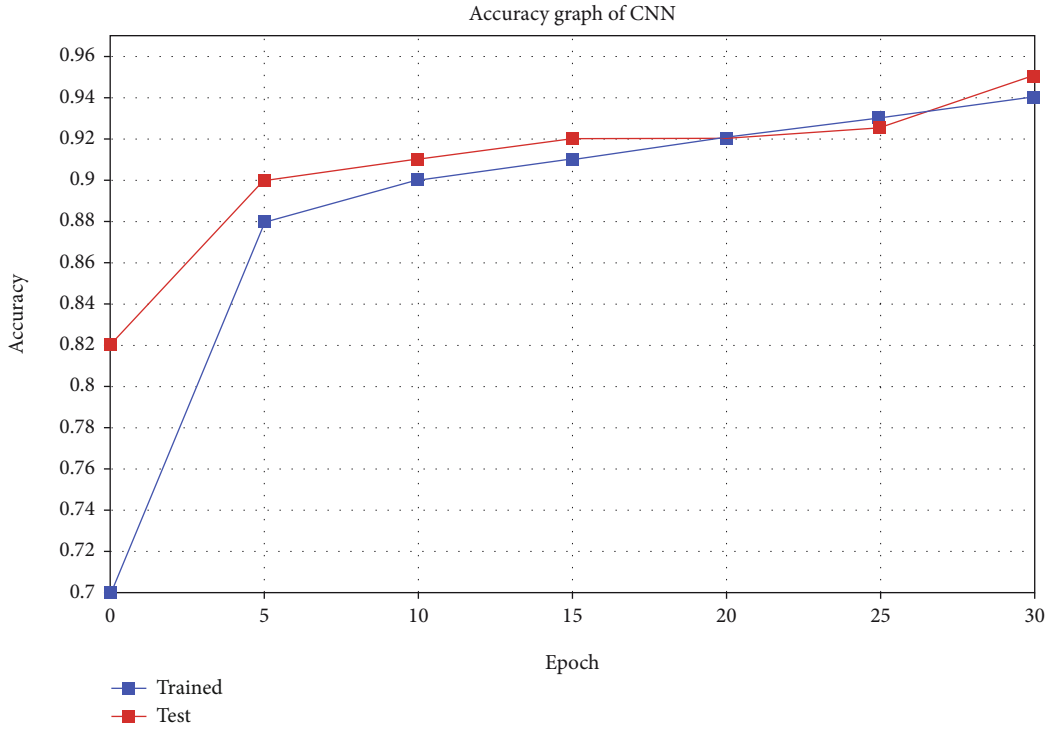


Figure 21: Accuracy Graph of CNN.

learn to recognise concepts like a person's shoulders or face borders. We may achieve a greater level of abstraction and in-depth knowledge from a CNN by stacking additional layers of convolutions on top of each other.

From Figure 9. Shows the computers transformed the image into Matrix Format. In the matrix, the cells filled with number 1 like wherever the object is identified, are filled with one remaining; all are -1. In this way, the image went divide into three filters which we discussed earlier.

4.2. Feature Map Formation. In Figure 10. Shows how the feature map extracted is shown. The matrix form of image and Filter is multiplied to get the feature map. Firstly, it gone started with the first cell and ended with the last cell. The

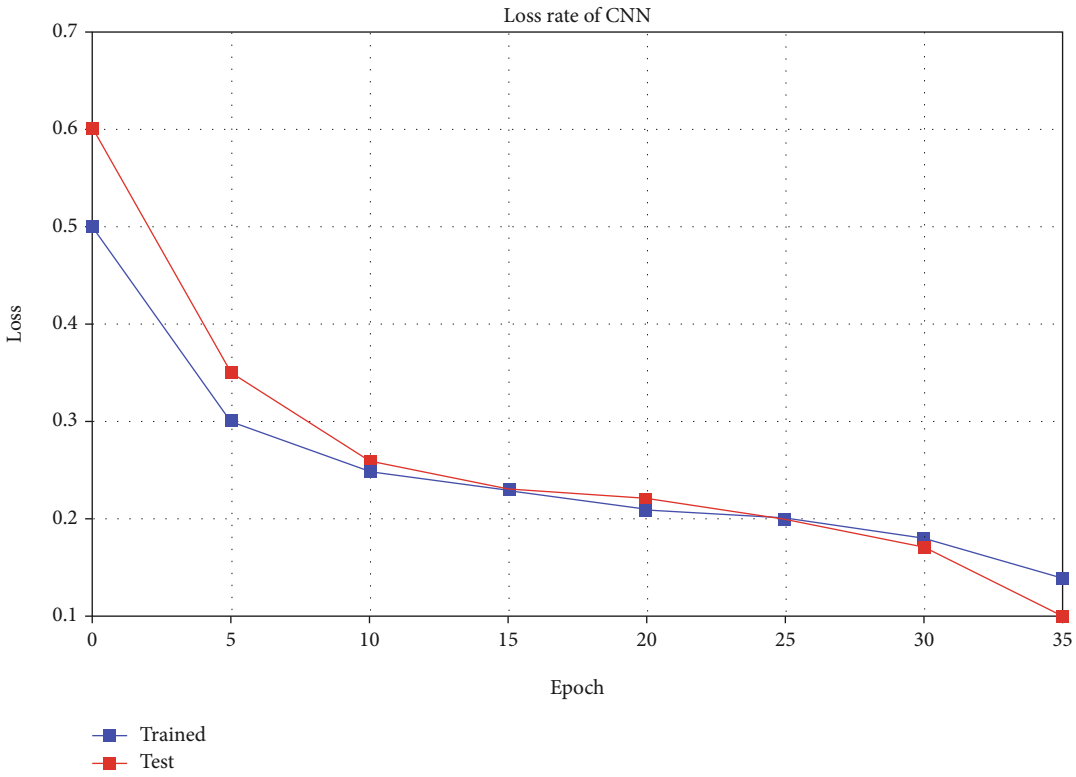


Figure 22: Loss rate of CNN.

Figure 23: Visualization of CNN.

Movement went be done with the strides. If the stride is 1 means, it is going with one cell after another. If strides are two means, it leaves one cell and is done with another from the starting. Like that, every Filter is multiplied. In the feature map, wherever the one is identified there itself the object of Filter is identified.

In Figure 11 shows the object detection identified from different features. The image matrix and feature matrix have gone multiply each other where ever the object is identified there the cell is filled with one the noticed that the feature matrix of the image is identified with some object in that cell. Like that way, the whole process will happen and then it identifies the different parts of the object in the image.

From Figure 12. After Extracting Feature maps, all the extracted features are arranged in stack order. Where ever the object of the image is identified, then the cell is filled with some number. Like that way, the image is divided in to different filters. In each Filter wherever the object is detected it and filled with a number. Like that way different filters identifies different objects all that identified objects can arrange in stack flow order to describe the whole image.

4.3. Applying Activation Function. The Activation Function is a mathematical barrier between the current node's input and the following layer's output. In order to achieve non-linearity, we use activation functions. Non-linear activation functions are employed to help the Multilayer network obtain the benefits of multi-layer networks. We would be treating this as a simple regression model if we did not use the activation function. In the Above Figure 13, the

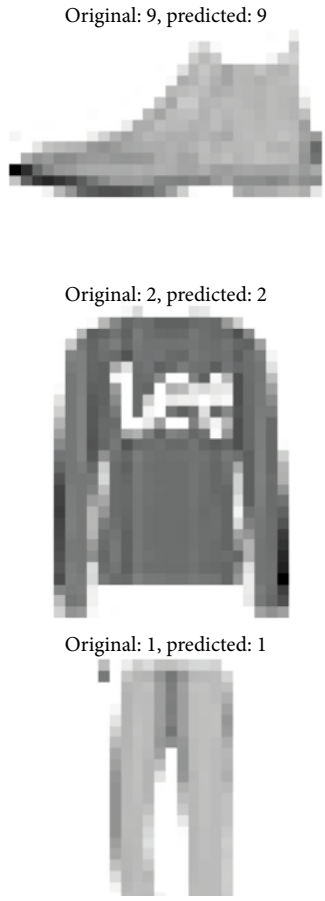


Figure 24: Visualization of ANN.

Activation Relu is applied. It transforms wherever the negative value comes; it turns to Zero and positive values remain the same.

4.4. Pooling. In the above, Figure 14. Pooling is done. The main aim of pooling is to reduce the size of an image. In the max-pooling operation done with stride 1, in this case, the top four cell values in the following matrix were used to perform a two-by-two filter pooling operation. Stride 1 means the cell is moved from one side to its right-hand side in such a way that the whole Maxpooling operation is done.

4.5. WorkFlow of Feature Extraction. In Figure 15, the total workflow of feature extraction is shown. The above diagram depicts the entire CNN operation. After that, with the help of the flattening layer, all detected objects can be filled into a one-dimensional array. After that, feature extraction and classification can be done with the help of fully connected layers.

5. Dataset

Reference: Dataset is downloaded from Kaggle

Zalando built the Fashion-MNIST dataset to replace the actual MNIST dataset of handwritten digits, powered by renewable energy. The dataset, Fashion-MNIST, contains 28x28 pixel images of fashion product photos. The images have greyscales with 70,000 images, of which 60,000 are training images and 10,000 are validation images. The 10 item class fashion item classification is done using CNN in this paper. In Figure 16, deep learning is tested using the original MNIST dataset, which Fashion-MNIST for testing replaces.

5.1. Setup Environment. This work's experimental setting was created by following a simple procedure. We have completed our process on the Windows 10 operating system. We have done it in the Google collab for better GPU performance. In this experiment, all networks were trained and tested using Tensor Flow and Keras on a computer. Not only is Google colab gone, it is done in a Jupyter notebook. But in this case, we are going to install each and every package that we need to work on this project. In Google Colab, all of these things are already set up and ready to go in the cloud environment. It will also be very useful for images with a lot of pixels.

5.2. Experiments and Results. Apply the concept to two distinct approaches, ANN and CNN. As applying different optimizers and activation functions to both of the models, the first thing to note is that in CNN, images are transformed into feature maps to classify images. In ANN, there is no feature map extraction. It takes too much computation. The results are very decent in both models.

5.3. Evaluation Metrics. In this research, we examine several models using various methods to help us decide which model is most appropriate. Three confusion metrics are used to judge our test dataset. They are precision, recall, and accuracy, as well as the F-measure. Figure 17. Shows the Classification of Class labels in Confusion Matrix. Figure 18. Shows the Classification Report of CNN Model Correctly Predicted ratios of Class Labels.

6. Experimental Results and Discussion

The experimentation is currently running on two models, ANN and CNN, each with different activation functions and optimizers. The number of epochs it takes to train networks and the learning rate, batch size, and architecture of networks must also be determined to construct the network architecture. The batch size is the number of samples set before the model is updated by instance. The number of epochs refers to the number of times an entire dataset is used. In particular, we use the following hyperparameters: lr =0.005, BatchSize =512, and number of epochs =20.

From the above parameters, our model trained with decent accuracy resulted in a 0.9572. And the tested accuracy was around 0.9452. The dataset contains low-resolution images with 28x28 pixels. The train accuracy that we got was 0.9572 and the test accuracy was 0.9452.

6.1. Performance and Comparison with Other Models. In the section, we compare our model with other models that have been tested on the Fashion Mnist Dataset, CNN with SGD, and CNN with RMS ANN, SGD, and so on. Figure 17 shows

the train accuracy of different models, and Table 1. shows the test accuracy of different models in fashion Mnist. Different Models Test accuracy and comparing the best optimizers with other models are shown below in graph and table format. Figure 19 shows that the trained accuracy of the model with different optimizer models. The accuracy of epochs is calculated.

Figure 20 Shows that the Best Optimizer Value is Compared with Other Optimizer Models that Should be Shown in Graph. As we train the model, it overfits, as we can see from the graph above (Training Accuracy: 99 percent; Test Accuracy: 92 percent). Deep learning models perform well on training data but not on test data (over fitting). The most straightforward method for avoiding overfitting is to limit the model's size, or the number of learnable parameters. The second option to remove overfitting is to use regularization techniques like L1, L2, and dropout. If the neural network contains fewer layers and neurons per layer, it may not perform well on the training dataset, resulting in an underfitting problem. Furthermore, if a neural network has a huge number of layers and neurons per layer, it may perform well on the training dataset but not so well on the test dataset, resulting in an over-fitting problem. Unfortunately, there is no exact formula for determining your model's architecture (number of layers, number of neurons in each layer). You'll have to rely on trial and error (experimentation) to figure out the best (best) architecture for your project.

Though there are other regularization strategies for neural networks, such as L1 and L2 regularization, dropout is the most successful and widely used regularization technique. When dropout is applied to a layer, it "drops out" (i.e., sets to zero) a number of the layer's output features (neurons) at random. The "dropout rate" is the percentage of features (neurons) in a layer that are zeroed out. The dropout rate is usually kept between 0.2 and 0.5 percent. Because dropout is only used during the training phase and is handled internally by the TensorFlow Keras API (evaluate() method), we do not need to remove it explicitly (manually) from the model when predicting on the test dataset. Dropout is not used in the test phase; instead, the layer's output values are lowered by a factor equal to the dropout rate, giving the impression that the same number of neurons are engaged in the test phase as in the training phase.

As a result, to encode the over-fitting complexity in our scenario, we'll use the dropout regularization technique. To use TensorFlow Keras to implement the dropout, we create a dropout layer and position it directly after the layer to which the dropout has to be applied. Figure 21 shows that the downfall of training accuracy from 99 to 95 after applying the dropout rate.

Figure 21 shows that the downfall of training accuracy from 99 to 95 after applying the dropout rate. Figure 22. Shows that the 'Accuracy" and "Loss" plots above, we see that there is no Over fitting observed now in the model. Table 1 show that the test accuracy of different models was mentioned in the tabular format. From the Figures 23 and 24 shows that the Our trained model is used to predict the images on the test images. Here in this way the images gone be predicted here we used to predict first twelve images of test data. The Attributes that are used are Actual and Predicted. The First twelve images are predicted correctly without any error.

From the Figures 23 and 24 shows that the Our trained model is used to predict the images on the test images. Here in this way the images gone be predicted here we used to predict first twelve images of test data. The Attributes that are used are Actual and Predicted. The First twelve images are predicted correctly without any error.

7. Conclusion and Future Work

In this paper, we address the problem of distinguishing clothing elements in fashion photographs using a CNN. This is accomplished using the fashion MNIST dataset, which contains many images, together with CNN and ANN algorithms for accurate and effective image classification. Image recognition is becoming increasingly crucial as deep learning algorithms progress. CNN recognition is commonly used when it comes to fashion-related applications, such as garment classification, retrieval, and computerised clothing labelling. In this study, we employ CNN architecture on the Fashion MNIST dataset. We would like to compare this with different datasets. Fashion MNIST is a dataset having low-resolution images. We would like to try these high-resolution images in the future, and we also plan to put CNN architecture to the test on a dataset of real-life apparel pictures that we amassed ourselves.

Data Availability

The data are available from the corresponding author upon reasonable request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work is not funded by any Institution or Organization.

References

[1] Y. Chun, C. Wang, and M. He, "A novel clothing attribute representation Network-Based Self-Attention Mechanism," *IEEE Access*, vol. 8, pp. 201762–201769, 2020.

[2] S. S. Kadam, A. C. Adamuthe, and A. B. Patil, "CNN model for image classification on MNIST and fashion-MNIST dataset," *Journal of Scientific Research*, vol. 64, no. 2, pp. 374–384, 2020.

[3] Kaggle, 2019, https://www.kaggle.com/zalando-research/fashionmnist.

[4] M. Kayed, A. Anter, and H. Mohamed, "Classification of Garments from Fashion MNIST Dataset Using CNN LeNet-5 Architecture," in *International Conference on Innovative Trends in Communication and Computer Engineering (ITCE)*, Aswan, Egypt, 2020.

[5] M. Z. Alom, T. M. Taha, C. Yakopcic et al., "A state-of- the-art survey on deep learning theory and architectures," *Electronics*, vol. 8, no. 3, p. 292, 2019.

[6] Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang, "Random erasing data augmentation," 2017.

[7] S. Bhatnagar, D. Ghosal, and M. H. Kolekar, "Classification of Fashion Article Images Using Convolutional Neural Networks," in *In 2017 Fourth International Conference on Image Information Processing (ICIIP)*, pp. 1–6, Shimla, India, 2017.

[8] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, http://arxiv.org/abs/1409.1556.

[9] X. Li and Z. Cui, "Deep Residual Networks for Plankton Classification," in *In OCEANS 2016 MTS/IEEE Monterey*, pp. 1–4, Monterey, CA, USA, 2016.

[10] Y. Seo and K. S. Shin, "Hierarchical convolutional neural networks for fashion image classification," *Expert Systems with Applications*, vol. 116, pp. 328–339, 2019.

[11] M. McKenna, "A comparison of activation functions for deep learning on fashion-MNIST," 2017, http://arxiv.org/abs/1708.07747.

[12] S. Shen, *Image Classification of Fashion-MNIST Dataset Using Long Short-Term Memory Networks*Research School of Computer Science.

[13] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," 2017, http://arxiv.org/abs/1708.07747.

[14] N. Jmour, S. Zayen, and A. Abdelkrim, "Convolutional Neural Networks for Image Classification," in *2018 International Conference on Advanced Systems and Electric Technologies (ICA-SET)*, pp. 397–402, Hammamet, Tunisia, 2018.

[15] W. Hu, Y. Huang, L. Wei, F. Zhang, and H. Li, "Deep convolutional neural networks for hyperspectral image classification," *Journal of Sensors*, vol. 2015, 12 pages, 2015.

[16] F. Manessi and A. Rozza, "Learning Combinations of Activation Functions," in *In 2018 24th International Conference on Pattern Recognition (ICPR)*, pp. 61–66, Beijing, China, 2018.

[17] A. Rama, A. Kumaravel, and C. Nalini, "Construction of deep convolutional neural networks for medical image classification," *International Journal of Computer Vision and Image Processing*, vol. 9, no. 2, pp. 1–15, 2019.

[18] N. Sharma, V. Jain, and A. Mishra, "An analysis of convolutional neural networks for image classification," *Procedia Computer Science*, vol. 132, pp. 377–384, 2018.

[19] S. Kaddoura, G. Chandrasekaran, D. Elena Popescu, and J. H. Duraisamy, "A systematic literature review on spam content detection and classification," *PeerJ Computer Science*, vol. 8, article e830, 2022.

[20] G. Di Franco and M. Santurro, "Machine learning, artificial neural networks and social research," *Quality and Quantity*, vol. 55, no. 3, pp. 1007–1025, 2021.