FASHION IMAGE CLASSIFICATION

ADRIANUS HANS VIARY - 56947929, City University of Hong Kong, Hong Kong
ARRIGO WILLIAMS NG - 56329943, City University of Hong Kong, Hong Kong
BENEDICT RONALDO GUSTAF - 56680002, City University of Hong Kong, Hong Kong
ENRYL EINHARD - 56731436, City University of Hong Kong, Hong Kong
JONATHAN ANDIKA WIBOWO - 56284830, City University of Hong Kong, Hong Kong
KEANE DYLAN YENNOTO - 56700945, City University of Hong Kong, Hong Kong

ABSTRACT

This paper creates a convoluted neural network model to classify images, especially in the different types of fashion. Forty-four thousand four hundred sixteen fully labeled images were used from the dataset, selecting the top fifteen articles with the highest data counts selected, and four different types of models were tested. Model 1 has the first dropout sequence layer, and Model 2 uses the second dropout sequence layer. Model 3 and 4 used the RMSProp optimizer. Model 3 has the first dropout sequence layer, and Model 4 uses the second dropout sequence. The results show model 2 as the best model, which uses a Convolutional Neural Network and Adam optimizer with a dropout sequence of 0.5, 0.25, and 0.2 on the first, second, and third hidden layer. This model can label most items correctly with a test accuracy of 91.77% and a test loss of 0.5113. Further research can be done using a larger resolution of images, more article types in fashion, and an enlarged image database to increase accuracy and decrease the loss value. In addition, results from this research can be implied to various image search engines in the fashion field to help classify fashion products on the online platform.

Additional Key Words and Phrases: CNN, Image Classification, Adam optimizer, Fashion Article types

INTRODUCTION

Towards the new digital era, the sales of e-commerce have increased over the following years. According to Khan [4], e-commerce signifies online transactions and information transfers. Statistics taken from Statista claim that the growth of e-commerce increased starting in 2019 [2]. Further research by a company also foresaw e-commerce sales would increase rapidly over the years [6]. Among all types of e-commerce that are rapidly developing, fashion develops the fastest among the others. Since 2018, more than 57% of the global internet has purchased fashion-related products, which results in a market value of over $759.5 billion. The fashion company also predicts that fashion will be able to reach 1 trillion USD in overall sales in 2025 through a 7.2% of annual growth rate [6].

However, the continuous rapid growth of the market may cause issues in online retail shops. The easy access to the online market has increased the number of sellers and customers who sometimes are not familiar with the application features. This paper addresses seller that does not label their items before uploading them to the e-commerce platform, especially in fashion retails. Thus, items can be wrongly assigned or not categorized at all due to the lack of knowledge from the sellers, which prevents customers from using the search query to find a type of fashion. The lack of features in fashion-related applications

may lead to the customer's dissatisfaction with the application and prolong the time for each customer to find a specific good.

This paper aims to create a model for image classification of fashion goods, ensuring that all items are labeled correctly and will appear on the search page when filtered by customers. With an accurate model, this algorithm will reduce human error when sellers upload fashion items for sale, and the items will appear in more searches, increasing the engagement of customers.

METHODS & PROCEDURE

A suitable dataset is needed to train and compare the convolutional neural network models. In order to do that, a dataset containing 44.416 images of fashion objects was retrieved from Kaggle [1]. The dataset provided each image with an article type and id to distinguish the fashion objects. Other information provided in the datasets is gender, master category, subcategory, article type, base colour, season, year, usage, and product display name.

For the purpose of ensuring the usability of the data and the performance of our models, the data are needed to be pre-processed.

First, the data of each article type in the data is counted. The fifteen article types with the highest data counts are selected to be classified by our model. The reasoning behind this is to ensure the quality of our model is not hindered by the variety of article types, especially with a limited dataset.

```
# Importing labels dataset
df = pd.read_csv(data_path + 'styles.csv', error_bad_lines=False)

# Choosing label 'articleType' to be focused and formating the type
df_type = pd.DataFrame(df['articleType'].value_counts())
df_type['articleType'] = df_type['articleType'].astype(int)
df_type

# Choosing the top 15 classes
types = list(df_type.head(15).index)
types
```

Figure 1: Choosing the types of fashion objects to be classified.

Second, the raw datasets provided by Kaggle are not labeled, so the data need to be labeled.

```
# Labeling the data
df_labels = df.loc[df['articleType'].isin(types)]
df_labels = df_labels[['id', 'articleType']]
df_labels = df_labels.reset_index(drop=True)
df_labels
```

Figure 2: Labeling the data.

Next, in order to ensure the usability of the dataset, the data are checked for the missing or rejected images to be removed. Also, to be used in our model, the article types of the data are converted from string to numerical data (integer).

```
# Importing Images
img_data_list = []
rejected_image = []

for i in df_labels['id']:
    input_img=cv2.imread(data_path + '/images' + '/'+ str(i) + '.jpg')
    try:
        input_img_resize = cv2.resize(input_img, dsize=(64, 64))
        img_data_list.append(input_img_resize)
    except:
        rejected_image.append(i)
        print(f"Rejected Image: {i}.jpg")

# Remove rejected image from df_label
df_labels = df_labels[~df_labels['id'].isin(rejected_image)]
df_labels = df_labels.reset_index(drop = True)

# Converting articleType(string) to numerical data
labels = df_labels['articleType'].unique().tolist()
df_labels['label'] = df_labels['articleType'].apply(lambda x: labels.index(x))
```

Figure 3: Checking for missing data or values.

Lastly, the data are formatted and normalized in order to be able to be used in the model.

```
# Formatting and Normalization of the data
X = np.array(img_data_list)
X = X.reshape(-1, 64, 64, 3)
X = X.astype('float')
X = X/255

Y = np.array(df_labels['label'])
Y = Y.reshape(len(X),)
```

Figure 4: Data formatting and normalization.

MODEL

This project adopts the Convolutional Neural Network (CNN) architecture to train four models to classify fashion images into 15 categories. It is not effective to use classical neural network architectures for image classification. This is due to the fact that they ignore the spatial structure of images. This results in input pixels being equally treated, regardless of their distance. Therefore, a convolutional neural network is used to take into account the spatial structure of an image by using 2D Convolutional layers to learn images' features. In a CNN, images are processed by extracting features, where manual feature extraction is unnecessary. Thus, CNN's work typically uses far less preprocessing than other image classification algorithms to extract features from images.

CNN is constructed by input, hidden, and output layers. The network was then trained in a number of epochs, the algorithm's number of repetitions during the training process.

This project fed the input layer with 64x64 pixel images and ran the training process with 150 epochs. We used a small size for each image because fashion images do not have many details, and the structure of the images could be easier to differentiate between different classes. The small image scale enables us to reduce the amount of data, providing more time allocation to train the models with a larger number of epochs where the network needs to learn the large dataset volume given the plenty of fashion image categories to be classified in the network.

In the three hidden layers, each layer consisted of a convolutional layer with a ReLU activation function, a pooling layer using MaxPooling, and a dropout layer to reduce overfitting.

In the convolution layer, the primary aim is to extract and detect features from the input fashion image. As a result, smaller data are able to be extracted from the original image while preserving the essential spatial relationship between the images' pixels, compiled in a convolved feature map. Then, the results from the convolution process are given an activation function ReLU to increase non-linearity in the network due to the non-linearity of the original images. The ReLU function converts negative values from the feature map to zero effectively. This activation function could reduce the time needed to train the network while maintaining a decent generalization of the accuracy.

Then, the data outputted after applying the ReLU function are progressively reduced by the MaxPooling layer. The input data for this layer are split into different sets of pixel blocks, and in this project, we applied the 2x2 pool size. By using MaxPooling, we are able to output the maximum pixel value in each pool, reducing the parameters of the input data while showing the important features only.

In the network architecture, we use two different sequences of dropout layers to find any effect on the loss and accuracy of the model result. Dropout layers are used as an alternative to reduce the overfitting of the model. In this layer, the inputs were randomly removed from the training process, and the parameters corresponding to these eliminated inputs are not updated. The first dropout layer sequence is 0.25 in all layers, meaning one out of four inputs is discarded in each layer. The second sequence applies the largest dropout in the first layers while gradually decreasing it in the next layers. There is a 0.5 dropout in the first layer, 0.25 in the second, and 0.2 in the last hidden layer and output layer.

After the three hidden layers, the output will be turned into a long vector so it could be processed in the dense layer, known as a fully connected layer, is a layer that helps in changing the dimensionality of the output from the previous layers. This results in a relationship between values of the data that is easily defined by the model. Then, the data will be outputted to 15 different fashion image categories in the output layer, using the softmax activation function.

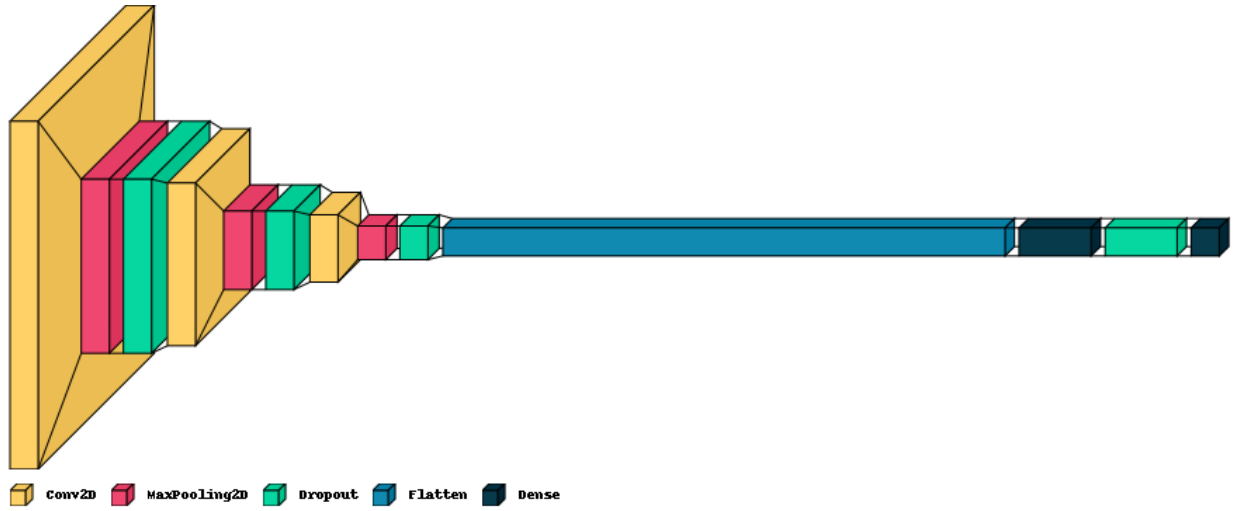

Figure 4: Visualization of the layers in the model

```
model = Sequential()
model.add(Conv2D(32, (3, 3), input_shape = (64,64,3), activation='relu'))
model.add(MaxPooling2D(pool_size = (2, 2)))
model.add(Dropout(0.25))

model.add(Conv2D(64, (3, 3), input_shape = (64,64,3), activation='relu'))
model.add(MaxPooling2D(pool_size = (2, 2)))
model.add(Dropout(0.25))

model.add(Conv2D(128, (3, 3), input_shape = (64,64,3), activation='relu'))
model.add(MaxPooling2D(pool_size = (2, 2)))
model.add(Dropout(0.25))

model.add(Flatten())
model.add(Dense(units = 512, activation = 'relu'))
model.add(Dropout(0.25))
model.add(Dense(units = 15, activation = 'softmax'))
```

```
model2 = Sequential()
model2.add(Conv2D(32, (3, 3), input_shape = (64,64,3), activation='relu'))
model2.add(MaxPooling2D(pool_size = (2, 2)))
model2.add(Dropout(0.5))

model2.add(Conv2D(64, (3, 3), input_shape = (64,64,3), activation='relu'))
model2.add(MaxPooling2D(pool_size = (2, 2)))
model2.add(Dropout(0.25))

model2.add(Conv2D(128, (3, 3), input_shape = (64,64,3), activation='relu'))
model2.add(MaxPooling2D(pool_size = (2, 2)))
model2.add(Dropout(0.2))

model2.add(Flatten())
model2.add(Dense(units = 512, activation = 'relu'))
model2.add(Dropout(0.2))
model2.add(Dense(units = 15, activation = 'softmax'))
```

Figure 5: Details of the layers in the model.

In compiling the model, we use sparse categorical cross-entropy and not categorical cross-entropy as the loss function since the nature of our classification is mutually exclusive. One sample can only belong to exactly one fashion object. There are no cases of one sample that could have multiple soft classifications or, in this case, one sample that could have multiple classes of fashion objects.

For the optimizer, we attempted to utilize two types of optimizers. Adam and RMSProp, the two most popular optimizers for image classification. Model 1 and 2 are built using the Adam optimizer, where Model 1 has the first dropout sequence layer, and Model 2 uses the second dropout sequence layer. Model 3 and 4 used the RMSProp optimizer. Model 3 has the first dropout sequence layer, and Model 4 uses the second dropout sequence.

Table 1: Model descriptions

Models	**Optimizer**	**Dropout Sequence**
Model 1	Adam	1 (0.25, 0.25, 0.25, 0.25)
Model 2	Adam	2 (0.5, 0.25, 0.2, 0.2)
Model 3	RMSProp	1 (0.25, 0.25, 0.25, 0.25)
Model 4	RMSProp	2 (0.5, 0.25, 0.2, 0.2)

RESULTS

This section will see the classification results of the four models mentioned in the previous section. We will perform a classification task for our models using a test dataset different from the training dataset. Then, we will evaluate the models using accuracy metrics. Below is the result for each model.

Model 1

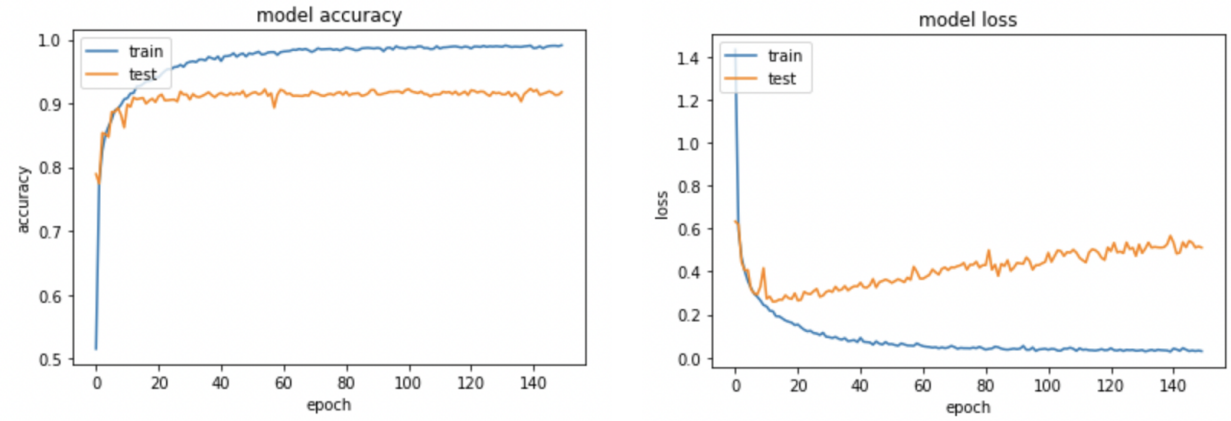

Figure 6: Graph of model 1 accuracy and loss.

Table 2: Model 1 result.

Training Accuracy	0.9879	**Training Loss**	0.0380
Test Accuracy	0.9157	**Test Loss**	0.5760

Model 2

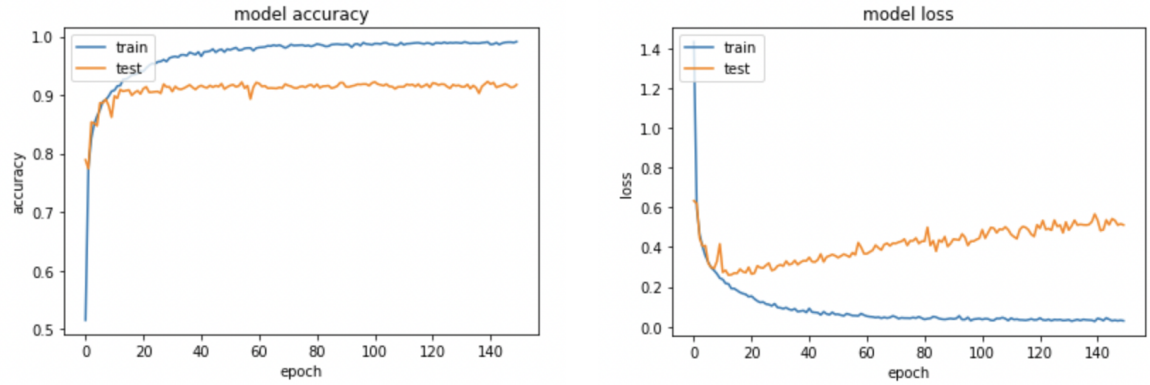

Figure 7: Graph of model 2 accuracy and loss.

Table 3: Model 2 result.

Training Accuracy	0.9911	**Training Loss**	0.0296
Test Accuracy	0.9177	**Test Loss**	0.5113

Model 3

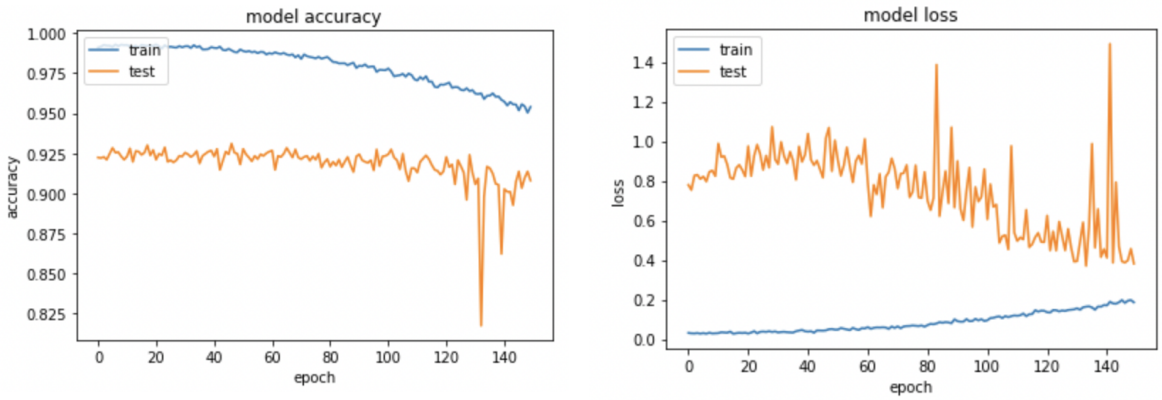

Figure 8: Graph of model 3 accuracy and loss.

Table 4: Model 3 result.

Training Accuracy	0.9540	**Training Loss**	0.1865
Test Accuracy	0.9080	**Test Loss**	0.3820

Model 4

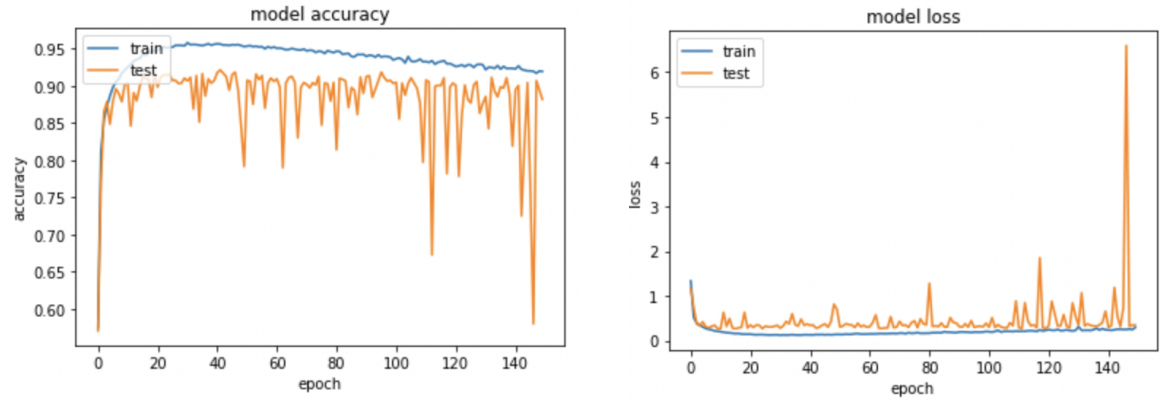

Figure 9: Graph of model 4 accuracy and loss.

Table 4: Model 4 results.

Training Accuracy	0.9197	**Training Loss**	0.2937
Test Accuracy	0.8822	**Test Loss**	0.3474

Based on the above results, we can see that model 2 has the biggest accuracy. However, model 1 and model 3 still perform well and only have a small difference in accuracy compared to model 1. Model 4 is the least accurate compared to others. Based on these results, we conclude that Model 2, which uses a CNN model and adam optimizer, performs best in this dataset.

Using model 2, we create a function to predict an image directly. Below is the implementation.

```
def predictFashionType(image_path, true_label = None):

    input_img=cv2.imread(image_path)
    plt.imshow(input_img)

    try:
        input_img_resize = cv2.resize(input_img, dsize=(64, 64))
    except:
        print(f"Invalid Image")

    image = np.array(input_img_resize)
    image = image.reshape(1,64,64,3)
    image = image.astype('float')
    image = image/255
    prediction = model.predict(image)
    label_pred = labels[np.argmax(prediction)]
    print(f"Predicted Label: {label_pred}")

    if true_label != None:
        print(f"True Label: {labels[true_label]}")
```

```
predictFashionType("1855.jpg", "Tshirts")
```

```
Predicted Label: Tshirts
True Label: Tshirts
```

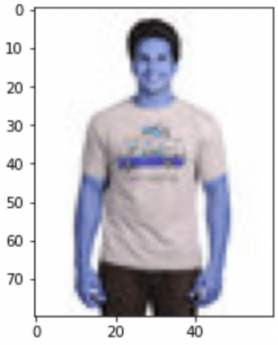

Figure 10: Prediction function and an example of predicting a fashion image.

DISCUSSION

Comparing the two groups with different optimizers applied when compiling the models, we found that models with adam optimizer have higher accuracy with higher loss. On the other hand, models with RMSProp optimizer have lower accuracy and also a smaller loss in the model evaluation.

As dropout is used to improve the model's generalization, higher dropout in the first hidden layers after the convolutional layers could improve the model performance due to more output randomly set to 0. This affects the weight updating process of backpropagation, resulting in a better model when used to test images that were not inputted as training data [5].

These optimization algorithms are intended to minimize loss, which is the difference between the predicted and expected results, by modifying the weights and learning rate of the network after each epoch.

However, the loss result from the models could be not credible since we only assign 15 categories as the labels in the output layer, whereas the image dataset includes several fashion images that do not belong to the 15 categories. These images could be classified as outliers since they will have a large difference from the predicted output, increasing the loss for the models despite the high accuracy.

As the accuracy measures the number of correct predictions to the total predictions made, the evaluation shows that the model performed well when predicting the fashion images. The adam optimizer models are better with a 0.01-0.02 difference than the RMSProp models.

Since Adam optimizers were built inheriting the features of RMSProp and another optimizer called Adagrad, Adam uses the second moment of the gradients as well, as opposed to the adapting learning rate based on the first moment in RMSProp. Additionally, the learning rate must be manually configured for several uses with RMSProp, and the suggested value does not fit every case [3]. Therefore, Adam's higher accuracy could be attributed to a feature that allows the learning rate to be modified for each network weight independently.

CONCLUSION AND FUTURE WORK

In conclusion, this paper has successfully created a model that can perform image classification with high accuracy and low loss. From our research, the best model to do it is model 2, which uses a Convolutional Neural Network and Adam optimizer with a dropout sequence of 0.5 on the first hidden layer, 0.25 on the second hidden layer, and 0.2 on the third hidden layer, and 0.2 on the output layer. With this model, we are able to label most items correctly with a test accuracy of 91.77% and a test loss of 0.5113. Although the loss on model 2 is higher than the losses of models 3 and 4, which uses RMSProp optimizer, the final results can not rely too much on loss as it is considered not credible. As a result, this experiment chose the model with the highest accuracy rate. It is hoped that this research can successfully reduce human errors from inexperienced sellers that might classify fashion items wrongly or even not classify them.

For future work, the models could be improved on several aspects. This can be done by increasing the number of categories predicted on the output layer. The current trained models can only predict 15 categories as this experiment only chose 15 with the most data. Improvements can be made by gathering more data from pictures on the internet for every type of fashion. In addition, the models could be trained and tested on images with higher resolution. As the current research trained and tested images with a 64x64 pixel resolution. In the future, the models could be trained and tested using a larger image size..

Furthermore, this research could be an innovation for an online market platform where users can search for fashion goods by uploading an image to a specific platform. Due to the abundant amount of types of fashion, the primary purpose of the image engine is to let the customers who do not know the product's type find their wanted goods quickly and accurately. This method can be preferable because labeling certain products could result in errors, while image classification can ensure the quality of the data is based on the accuracy of the future product in the online platform.

REFERENCES

[1]
Param Aggarwal. 2019. Fashion Product Images Dataset. *www.kaggle.com*. Retrieved April 15, 2022 from https://www.kaggle.com/datasets/paramaggarwal/fashion-product-images-dataset
[2]
Daniela Coppola. 2021. Global e-retail growth rate 2020, by macro-region. *Statista*. Retrieved April 12, 2022 from https://www.statista.com/statistics/1225478/global-b2c-e-commerce-growth-by-macro-region/
[3]
Ayush Gupta. 2021. A Comprehensive Guide on Deep Learning Optimizers. *Analytics Vidhya*. Retrieved from https://www.analyticsvidhya.com/blog/2021/10/a-comprehensive-guide-on-deep-learning-optimizers/
[4]
Abdul G. Khan. 2016. AutomatedQueueManagement CorporateSustainabilityFactors RolesofHumanResourcesManagers ImpactofElectronicDocuments VOLUME16ISSUE1VERSION1.0. *Global Journal of Management and Business Research* 16, 1 (2016). DOI:https://doi.org/10.17406/GJMBR
[5]
Cory Maklin. 2019. Dropout Neural Network Layer In Keras Explained. *Medium*. Retrieved from https://towardsdatascience.com/machine-learning-part-20-dropout-keras-layers-explained-8c9f6dc4c9ab
[6]
Aaron Orendorff. 2021. 10 Trends Styling 2021's Ecommerce Fashion Industry: Growth + Data in Online Apparel & Accessories Market. *Common Thread Collective*. Retrieved from https://commonthreadco.com/blogs/coachs-corner/fashion-ecommerce-industry-trends#:~:text=Ecommerce%20Fashion%20Industry%20Grows%20to%20%241%20Trillion%20by%202025