# FashionMNIST Classification using Convolutional Neural Network and K-Nearest Neighbors

Rithvik Srinivasaiya
MS in Data Science
Texas A&M University
s.rithvik@tamu.edu

Atharva Agashe
MS in Computer Engineering
Texas A&M University
atharvagashe22@tamu.edu

Sriram Balasubramanian
MS in Data Science
Texas A&M University
srib21@tamu.edu

*Abstract*— This research delves into using Convolutional Neural Networks (CNN) and k-Nearest Neighbors (KNN) for FashionMNIST dataset classification. FashionMNIST, a benchmark in computer vision, has 60,000 training and 10,000 testing images across ten fashion categories. The study aims to compare CNN and KNN efficacy in accurately categorizing diverse fashion items. Outcomes reveal strengths and weaknesses, assessed through performance metrics like accuracy, precision, recall, and F1 score. Findings provide valuable insights into algorithm suitability for image classification tasks, particularly within the clothing domain, contributing to the understanding of their potential applications in real-world scenarios.

**Keywords—FashionMNIST, CNN, KNN, Image Classification, Machine Learning)**

## I. Introduction

The realm of computer vision has undergone remarkable transformations in image classification, primarily propelled by the revolutionary impact of deep learning models [1]. In the context of deep learning, meticulous attention is devoted to hyperparameters—external elements influencing model behavior—demanding careful selection for optimal performance.

This research paper intricately elucidates the construction of a CNN model, enriched with a diverse array of regularization methods and hyperparameter optimization techniques. Importantly, the study broadens its scope to include a comparative analysis featuring the KNN model, representing traditional machine learning approaches in image classification. The research objective is to recognize fashion objects through the utilization of the FashionMNIST dataset, comprising 60,000 training set samples and 10,000 test set samples. The F-MNIST dataset encompasses grayscale images of fashion products categorized into 10 distinct classes, each with a resolution of 28x28 pixels. This comprehensive approach explores the capabilities of deep learning while also providing a comparative examination with the KNN model. The dual-model strategy contributes to a nuanced understanding of the strengths and limitations of both methodologies in the context of image classification tasks.

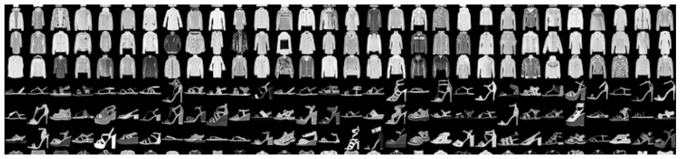
**Fig.1. A comprehensive visual representation of few images from the Fashion-MNIST Dataset.**

The paper scrutinizes the KNN algorithm and its relevance in the FashionMNIST dataset, a curated collection of fashion-related images. Through meticulous examination, the nuances of KNN are dissected, exploring its efficacy in pattern recognition, data mining, and fashion classification [2]. Additionally, the investigation involves the integration of k-fold cross-validation, a robust technique employed to enhance the reliability and generalizability of KNN [3] models tailored for fashion-related image datasets.

Shifting focus to CNNs in the second part of the study, their transformative role in fashion image processing is explored. In a seminal study by Krizhevsky et al. (2012) [5], the effectiveness of deep neural networks in achieving remarkable results on complex datasets through supervised learning was examined. Recent research endeavors have extensively employed CNNs for image classification tasks, with studies incorporating features such as residual skip connections and batch normalization (BN) to enhance efficiency and speed [7].

Within the realm of CNNs, the challenge of overfitting remains significant, particularly in the context of fashion image classification. This research addresses overfitting and introduces the concept of dropout as a regularization method [8]. Dropout, proposed by Srivastava et al. (2014)[9], involves randomly ignoring or "dropping out" selected neurons during training. This acts as a regularizing mechanism, preventing the model from becoming overly specialized and fostering improved generalization tailored for the nuances of the FashionMNIST dataset. Moreover, the adoption of the Rectified Linear Unit (ReLU) as an activation function [10] has effectively addressed the gradient vanishing problem, enabling faster training.

Expanding on the discussion of overfitting, the paper delves into the specific challenges posed by this phenomenon in the context of fashion image classification. Understanding the nuances of overfitting in fashion-related data is essential for developing robust models that can discern unique patterns and features within the FashionMNIST dataset, ensuring accurate and reliable categorization of fashion items.

The final segment introduces the concept of data augmentation [11] as a key strategy to address challenges associated with limited labeled data in the fashion domain. By augmenting the training dataset with diverse transformations, the model's ability to generalize to a broader spectrum of real-world fashion scenarios is enhanced, thereby bolstering overall performance in the FashionMNIST context.The field of computer vision has undergone significant advancements in image classification tasks, largely driven by the transformative impact of deep learning models[1]. Within the context of deep learning, careful attention is given to hyperparameters, discrete elements external to the model architecture that exert substantial influence on its behavior, requiring meticulous selection for optimal performance.

## II. METHOD

### A. *Exploratory Data Analysis (EDA)*

Before delving into the intricacies of model training and evaluation, we performed EDA . The primary objective of EDA was to gain deeper insights into the Fashion MNIST dataset, uncover underlying patterns, and identify any anomalies or inconsistencies that might affect the subsequent modelling phase.

- Size and Shape Analysis: The training dataset contains a total of 60000 samples, while the testing dataset comprises 10000 samples. The dimension of each image in the dataset was 28 x 28.

- Class Distribution Analysis: To understand the distribution of data across different classes, we combined the labels from both training and testing datasets and performed a bin count and created a data frame. Each class had 700 images which showed that the dataset has a uniform number of samples in all the classes.

- Sample Visualization: We visually examined samples from each class by selecting and displaying the first image of every class from the training dataset.

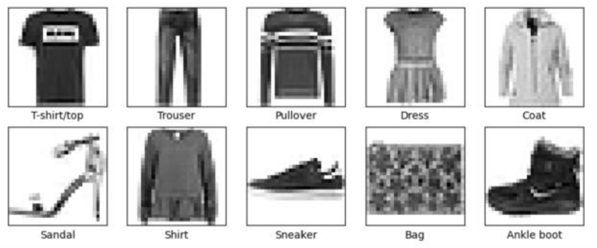

**Fig.2. Data samples and their class labels**

- Pixel Intensity Exploration: For a more granular analysis, the pixel values of images were flattened to understand the range and distribution of pixel values. This was represented in a data frame, which included both the numeric and textual class labels for each sample.

### *B.* Preprocessing for KNN

The preprocessing steps include the crucial transformation of the Fashion MNIST dataset for optimal application of the KNN classification model. Specifically, the focus lies on reshaping the 28 x 28 pixel images representing diverse fashion items.

- The initial matrix structure of each image is flattened into a one-dimensional array comprising 784 pixels. This essential reshaping process is fundamental for compatibility with the KNN model, as it facilitates a more nuanced analysis and classification of intricate fashion features.

- Normalization was applied to ensure that pixel intensity values, representing image features, were scaled to a uniform range between 0 and 1. Normalizing pixel values to a standardized range mitigates the influence of feature magnitudes, ensuring that the algorithm's decisions are not biased towards features with larger scales.

### *C.* Motivation for selecting KNN Model

KNN algorithm demonstrates remarkable adaptability across diverse data types and scenarios due to its non-assumptive approach toward underlying data distributions. This quality proves especially valuable when handling complex or non-linear data structures. The algorithm's adjustable hyperparameters, such as values for the number of neighbours, allow for fine-tuning through methods like cross-validation.

### *D.* Building the KNN Model

In the model development process, a systematic exploration of hyperparameters such as the number of neighbors, was done. Euclidean distance was used to calculate the distance between the neighbors.

- Hyperparameter Tuning: Throughout hyperparameter exploration, multiple iterations of the KNN algorithm were employed with number of neighbours (K) ranging from 3 to 11. This iterative process involved data segmentation into subsets for training and validation, allowing for an accurate assessment of each parameter's impact on model performance. The selection of the optimal number of neighbors value was determined based on value of K achieving the highest average accuracy.
- K-Fold Cross Validation: For validation, K-Fold cross-validation was employed to ensure the model's adaptability and performance across varying parameter settings. The training data was prepared for 5-fold cross-validation, to ensure robust model training.

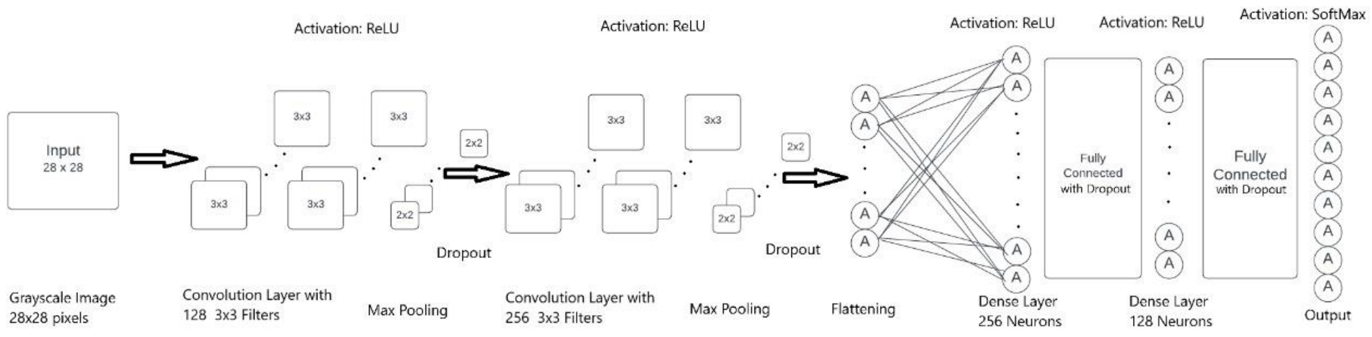

**Fig.3.Architecture of CNN Model**

### *E. Motivation for Selecting CNN Model*

CNN was chosen as the algorithm of choice because CNNs are effective in capturing local patterns in images, making them suitable for tasks like image classification where the relationship between pixels is informative of the content. For tasks like detecting and classifying objects in photos, CNNs are able to automatically and adaptively learn the spatial hierarchies of characteristics from input images. This contrasts with algorithms like decision trees or KNN, which generally do not perform as well on high-dimensional data like images.

### *F. Preprocessing for CNN*

The preprocessing steps include null value checking, normalization, reshaping, data splitting, and one-hot encoding. This collectively prepares the Fashion MNIST dataset for effective training and evaluation within a CNN framework. Each step plays a vital role in ensuring that the data is in the appropriate format and quality for the subsequent modeling phase.

- Reshaping for CNN Compatibility: To prepare the data for input into a CNN, we reshaped the images to include a channel dimension. The Fashion MNIST images, originally in a shape suitable for a dense network, were reshaped to (60000, 28, 28, 1) to accommodate the CNN's requirement for a 4D input tensor.
- Data Splitting: The dataset was divided into training, validation, and testing subsets. This was achieved by splitting the original training set, reserving 20% of its data for validation. The validation set is crucial for tuning the model's hyperparameters and evaluating its performance during training,This step ensures a robust evaluation of the model's generalization capabilities.
- One-Hot Encoding: The class labels were one-hot encoded to transform them into a binary matrix representation where each integer label is converted into a 10-element binary vector with a '1' in the position of the respective class and '0's elsewhere. This transformation is important for classification tasks with neural networks as it aligns the label format with the output layer's softmax activation in a multi-class setting.

### *G. Building the CNN Model*

- Convolutional Block 1: The initial convolutional block of our model comprises two 2D convolutional layers, each with 128 filters of size 3x3 and employing ReLU activation. This is followed by a Max Pooling layer with a 2x2 pool size, reducing spatial dimensions by half and aiding in overfitting prevention. A Dropout layer with a 0.25 rate concludes this block, further contributing to model regularization.
- The dimensions after the convolution layer and the max pooling layers are given by the following formulae

$$Output\ Height = \frac{Input\ Height - Filter\ Height}{stride} + 1$$

$$Output\ Width = \frac{Input\ Width - Filter\ Height}{stride} + 1$$

$$Output\ Height = \frac{Input\ Height - Pool\ Height}{stride} + 1$$

$$Output\ Width = \frac{Input\ Width - Pool\ Width}{stride} + 1$$

- Convolutional Block 2: The subsequent block contains two Conv2D layers, each with 256 filters and a 3x3 kernel, utilizing ReLU activation. This configuration intensifies feature extraction as the network depth increases. Similar to the first block, a MaxPooling2D layer reduces feature map dimensions, followed by a Dropout layer with a 0.25 rate to maintain robustness against overfitting.
- Flatten Layer: This layer transforms the 3D output from the preceding convolutional blocks into a 1D vector.
- Fully Connected Block: This block consists of 2 dense layers. The first Dense layer comprises of 256 neurons, utilizing ReLU activation. This layer plays a pivotal role in interpreting the high-level features extracted by the convolutional blocks. To enhance the model's resistance to overfitting, a Dropout layer is employed with an increased rate of 0.5. Another Dense layer is introduced with 128 neurons and ReLU activation. This layer furthers the network's capacity to understand

.

and process the complex features derived from the input data.

- Output Layer: The final component of the model is a Dense output layer, comprising 10 neurons corresponding to the ten classes of the Fashion MNIST dataset. This layer adopts the softmax activation function, which is integral for multi-class classification tasks.

The Adam algorithm with a learning rate set to 0.0005 was used as an Optimizer. As for the loss function, the categorical cross-entropy was used due to its suitability for multi-class classification tasks. The model's performance was evaluated using a comprehensive suite of metrics: accuracy, precision, recall, and the F1 score as shown in the confusion matrix below.

## III. Experiments

Series of trials were performed to adjust the model's hyperparameters optimally. The hyperparameter settings that yielded the most favorable outcomes were then integrated into the final model configuration.

### *A. Finding Optimum Number of Neighours*

The evaluation of the KNN algorithm with varying neighbor counts for the Fashion MNIST dataset revealed that a hyperparameter value of 5 for nearest neighbours provided the best balance between capturing local patterns and maintaining model generalization. While lower neighbor counts like 3 were sensitive to overfitting, and higher counts such as 11 oversimplified the data, a count of 5 neighbors achieved optimal performance, as demonstrated by superiority in accuracy.

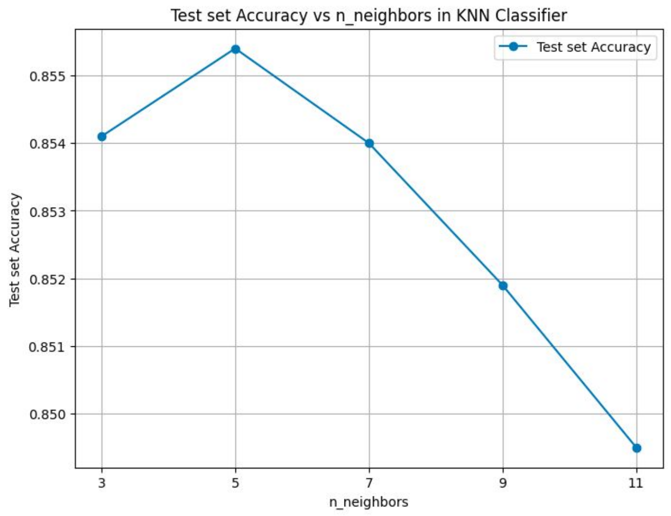

**Fig.4. Test Accuracy VS Number of Nearest Neighbours**

### *B. Finding the Optimum Number of Filters*

This experiment aimed to analyze the effect of different convolutional filter sizes on a CNN classification performance for the Fashion MNIST dataset. A set of predetermined filter sizes [16, 32, 64, 128, 256] was used to adjust the number of filters in the initial two convolutional layers of the network. The experiment's approach involved incrementally increasing the number of filters from 16 in the first layer to 256 in the final configuration, with each model's second convolutional layer utilizing the subsequent higher filter count from the set. Each model iteration was compiled with the Adam optimizer and categorical cross-entropy loss, and trained on the dataset for a consistent number of epochs. The performance was gauged using test set accuracy, and the results were plotted to determine the optimal filter configuration.

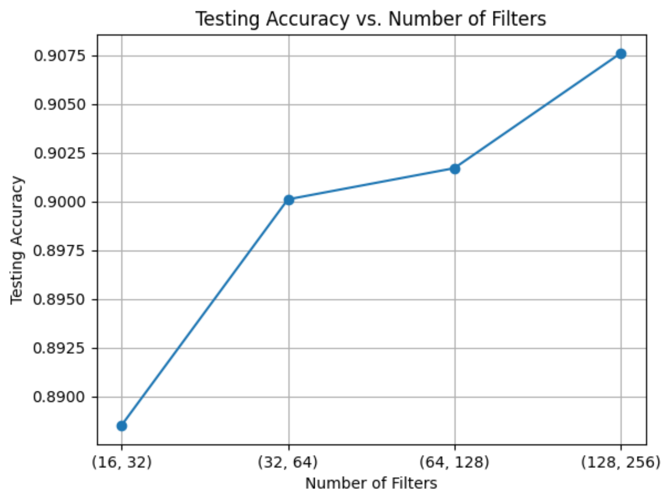

**Fig.5. Test Accuracy VS Number of Filters**

The plotted results indicate a positive correlation between the number of convolutional filters and model accuracy, with a combination of (128, 256) filters achieving the highest test accuracy. This suggests that increasing the complexity of the CNN model can lead to better performance on the Fashion MNIST dataset.

### *C. Finding the Optimum Value of Learning Rate*

This experiment was performed to evaluate the effect of varying learning rates on the training of a CNN for the Fashion MNIST dataset. Five different learning rates were tested: 0.0001, 0.0005, 0.001, 0.005, and 0.01. The same CNN model was used across all trials, featuring two sets of convolutional and max-pooling layers followed by dropout for regularization, and concluding with a fully connected network leading to a softmax output layer. The training process for each learning rate was monitored over 10 epochs, recording both loss and accuracy metrics.

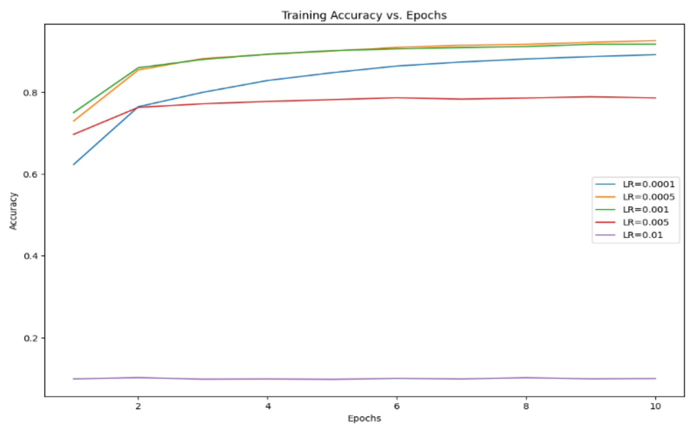

**Fig.6.Training Accuracy vs Epochs**

The results, as visualized in the graphs, indicate the model's loss decreased and accuracy increased with epochs for all learning rates. Lower learning rates showed a more gradual improvement, whereas higher rates exhibited faster convergence but with potential signs of instability or overfitting as indicated by the trends in loss and accuracy. However, learning rate of 0.0005 showed a consistent decrease in loss and increase in accuracy without any erratic behavior.

### *D. KNN vs CNN*

The two models were compared based on precision, recall, accuracy, and F1 scores. The results contribute valuable insights into the strengths and weaknesses of each model, guiding practitioners in making informed decisions about model selection for specific applications.

| **Model** | **Evaluation Metrics** | | |
|---|---|---|---|
| | **Accuracy** | **Precision** | **Recall** |
| **KNN** | 0.853 | 0.857 | 0.854 |
| **CNN** | 0.925 | 0.934 | 0.918 |

**Table.1. Comparison of Evaluation Metrices (Accuracy, Precision, Recall)**

| **Class wise F1 Score** | | |
|---|---|---|
| **Class** | **KNN** | **CNN** |
| **T-Shirt/top** | 0.818 | 0.878 |
| **Trouser** | 0.975 | 0.990 |
| **Pullover** | 0.764 | 0.889 |
| **Dress** | 0.881 | 0.926 |
| **Coat** | 0.765 | 0.880 |
| **Scandal** | 0.885 | 0.986 |
| **Shirt** | 0.609 | 0.776 |
| **Sneaker** | 0.899 | 0.971 |
| **Bag** | 0.957 | 0.983 |
| **Ankle boot** | 0.924 | 0.972 |

**Table.2. Comparison of F1 Scores**

## IV. CONCLUSION

In comparing K-Nearest Neighbors (KNN) and Convolutional Neural Network (CNN) models on the Fashion MNIST dataset, the CNN outperforms with accuracy, precision, and recall at 0.925, 0.934, and 0.918, respectively. In contrast, KNN scores lower at 0.853, 0.857, and 0.854. The CNN's superiority extends to classwise F1 scores, excelling in categories like 'Shirt,' 'Sneaker,' and 'Bag.' Its adeptness at extracting complex hierarchical features, facilitated by convolutional layers, contributes to this improvement. While KNN remains competitive in 'Trouser' and 'Coat,' its deficiency in the 'Shirt' category indicates a struggle with intricate patterns similar to other classes. In conclusion, the CNN model demonstrates a clear advantage in Fashion MNIST image classification, showcasing significantly superior metrics due to advanced feature extraction capabilities.

In comparing the K-Nearest Neighbors (KNN) and Convolutional Neural Network (CNN) models using the Fashion MNIST dataset, the CNN exhibits superior performance with overall accuracy, precision, and recall scores of 0.925, 0.934, and 0.918, respectively, outperforming the KNN's scores of 0.853, 0.857, and 0.854. This superiority extends to classwise F1 scores, where the CNN consistently achieves higher values across various classes, particularly excelling in categories like 'Shirt,' 'Sneaker,' and 'Bag.' The CNN's adeptness at extracting complex hierarchical features within images, facilitated by its convolutional layers, contributes to this marked improvement. Although the KNN remains competitive, particularly in the 'Trouser' and 'Coat' categories, its notable deficiency in the 'Shirt' category indicates a struggle with more intricate patterns that share similarities with other classes. In conclusion, the CNN model demonstrates a clear advantage in image classification on the Fashion MNIST dataset, showcasing significantly superior performance metrics, primarily attributed to its advanced feature extraction capabilities.

## V. REFERENCES

[1] K. Fu, D. Cheng, Y. Tu, L. Zhang, Credit card fraud detection using convolutional neural networks, Lecture Notes in Computer Science, 9949, 483–490, 2016, doi:10.1007/978-3-319-46675-0_53.

[2] Amato, G. and Falchi, F., 2010, September. kNN based image classification relying on local feature similarity. In *Proceedings of the Third International Conference on Similarity Search and Applications* (pp. 101-108

[3] Wayahdi, M.R., Syahputra, D. and Ginting, S.H.N., 2020. Evaluation of the K-Nearest Neighbor Model With K-Fold Cross Validation on Image Classification. *INFOKUM*, *9*(1, Desember), pp.1-6.

[4] Guo, Tianmei, et al. "Simple convolutional neural network on image classification." *2017 IEEE 2nd International Conference on Big Data Analysis (ICBDA)*. IEEE, 2017.

[5] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems (pp. 1097-1105).

[6] Xiao, H., Rasul, K., & Vollgraf, R. (2017). Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. arXiv preprint arXiv:1708.07747.

[7] Bhatnagar, S., Ghosal, D., & Kolekar, M. H. (2017, December). Classification of fashion article images using convolutional neural networks. In Image Information Processing (ICIIP), 2017 Fourth International Conference on (pp. 1-6). IEEE.

[8] Lemley, J., Bazrafkan, S., & Corcoran, P. (2017). Smart Augmentation Learning an Optimal Data Augmentation Strategy. IEEE Access, 5, 5858-5869

[9] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research, 15(1), 1929-1958.

[10] Agarap, Abien Fred. "Deep learning using rectified linear units (relu)." *arXiv preprint arXiv:1803.08375* (2018).

[11] Shijie, J., Ping, W., Peiyi, J., & Siping, H. (2017, October). Research on data augmentation for image classification based on convolution neural networks. In Chinese Automation Congress (CAC), 2017 (pp. 4165-4170). IEEE.