PDAN8412 POE PAR 3

Makabongwe Lwethu Sibisi

ST10145439

Table of Contents

QUSETION 1

Data Needed for CNN Image Classification

Convolutional Neural Networks (CNNs) are specialized deep learning architectures designed for image classification tasks that automatically extract hierarchical spatial features from visual data. Unlike traditional machine learning algorithms, CNNs utilize convolutional layers with learnable filters that detect patterns such as edges, textures, and complex objects, combined with pooling layers that reduce dimensionality while preserving critical features. This approach allows the model to learn spatial hierarchies directly from raw pixel data, making it particularly effective for classifying images by processing them through multiple layers of feature extraction before making final predictions through fully connected layers (Ultralytics, 2019) (GeeksforGeeks, 2025).

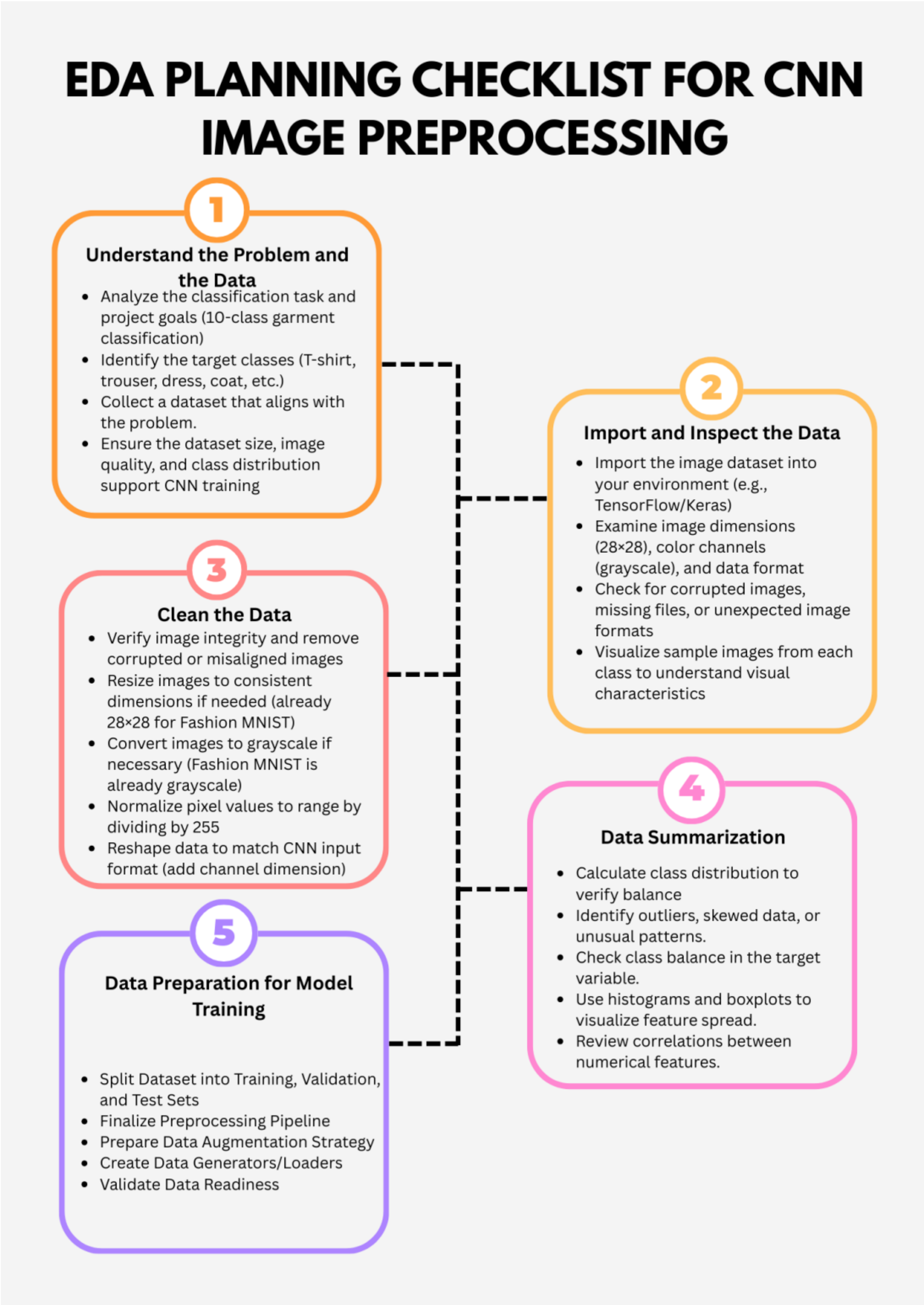
CNNs require data containing clean, well-structured, labelled images with consistent dimensions and adequate representation across all target classes. The Fashion MNIST dataset by Zalando Research from Kaggle is ideal, as it includes 70,000 grayscale images (60,000 for training and 10,000 for testing) of fashion articles across 10 balanced classes including T-shirt/top, trouser, pullover, dress, coat, sandal, shirt, sneaker, bag, and ankle boot. These 28×28 pixel images are perfect for the model because they maintain standardized dimensions with pixel values ranging from 0 to 255, representing darkness intensity, which allows the CNN to learn discriminative visual patterns associated with each garment category. Furthermore, the dataset's structured format, moderate size, balanced class distribution, and meaningful visual features between predictors and the classification outcome make it a strong foundation for implementation (Kayed, et al., 2020) (Zalando & Chris, 2017).

Data Quality Considerations & Common Pitfalls

Common pitfalls like inconsistent image dimensions, poor lighting conditions, class imbalance, and insufficient training samples distort the CNN's ability to extract generalizable features and accurately classify unseen images. High-quality, labeled datasets are crucial for accurate classification, as poor data quality can lead to inaccurate predictions and overfitting where the model memorizes training examples rather than learning robust features. Additionally, inadequate preprocessing such as improper normalization or lack of data augmentation may cause the model to perform poorly on test data, while imbalanced datasets can bias the network toward dominant classes. The Fashion MNIST dataset addresses these issues with its standardized structure, consistent 28×28 pixel dimensions, balanced class distribution with 7,000 samples per category, and pre-split training and testing sets. However, careful preprocessing is still required, including pixel normalization (scaling values to 0-1 range), proper data reshaping for CNN input layers, and potentially implementing data augmentation techniques such as rotation or shifting to enhance model robustness. Paying close attention to these factors ensures the resulting CNN model is robust, generalizable, and achieves high classification accuracy across all fashion categories (Ultralytics, 2019).

QUESTION 2

EDA

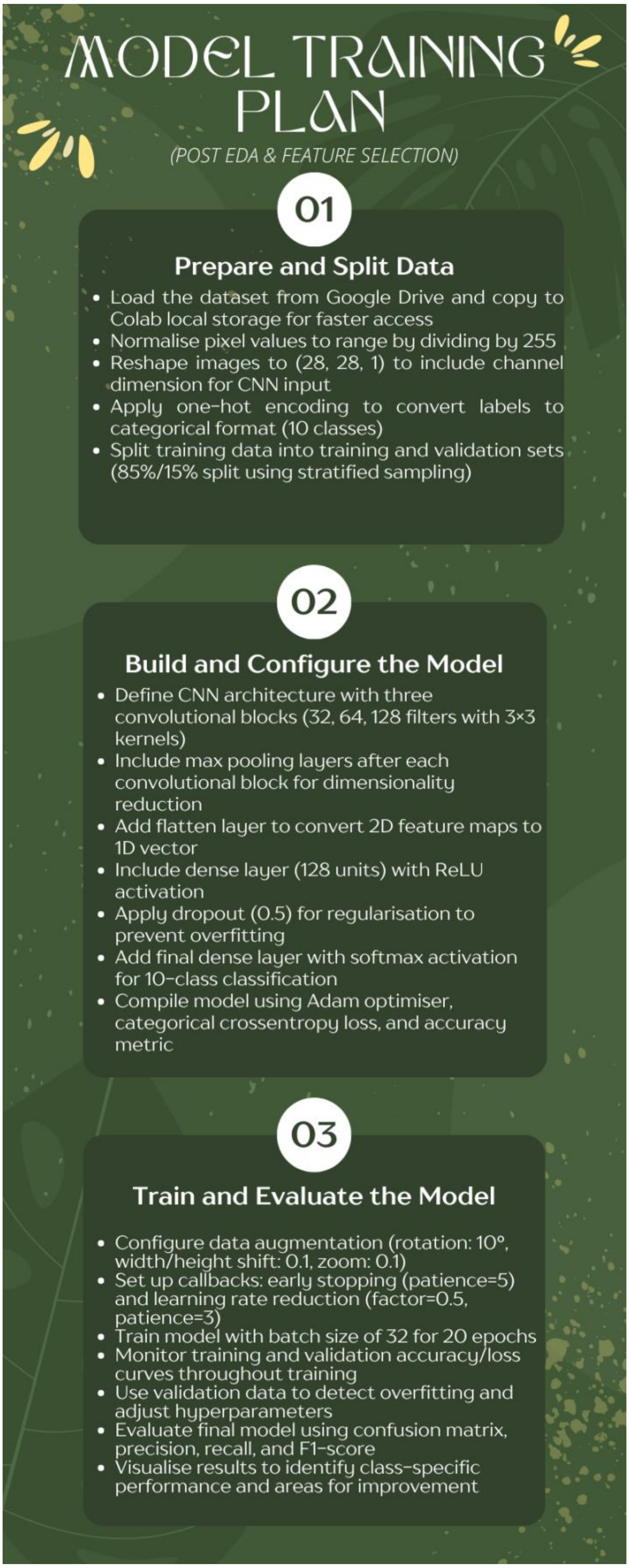


(GeeksforGeeks, 2025) (Byeon, 2020)

Feature Selection

Unlike traditional machine learning algorithms that require manual feature engineering, Convolutional Neural Networks automatically extract hierarchical features through their convolutional layers, learning progressively complex patterns from edges and textures to complete object representations. However, feature selection for CNNs involves critical preprocessing decisions that directly impact model performance. For the Fashion MNIST dataset, the process begins with determining optimal input representations, specifically maintaining the greyscale single-channel format (28×28×1) rather than converting to RGB, as this preserves computational efficiency whilst retaining all relevant visual information for garment classification. Next, I will implement strategic data normalisation by scaling pixel values to the range, which standardises feature magnitudes and accelerates convergence during training. To enhance the model's ability to extract robust features, I will apply data augmentation techniques such as rotation, width/height shifts, and zoom transformations, effectively expanding the feature space without additional data collection. Throughout this process, I will employ regularisation methods like dropout and L2 penalties to prevent overfitting and ensure the learnt features generalise well to unseen images. Finally, the selected preprocessing pipeline and augmentation strategy will be validated through cross-validation by comparing performance metrics like accuracy, precision, recall, and F1-score, ensuring the final approach maintains optimal feature extraction capability for accurate garment classification across all 10 Fashion MNIST categories (GeeksforGeeks, 2025).

Train model


MODEL TRAINING PLAN
(POST EDA & FEATURE SELECTION)
01
Prepare and Split Data
• Load the dataset from Google Drive and copy to Colab local storage for faster access
• Normalise pixel values to range by dividing by 255
• Reshape images to (28, 28, 1) to include channel dimension for CNN input
• Apply one-hot encoding to convert labels to categorical format (10 classes)
• Split training data into training and validation sets (85%/15% split using stratified sampling)
02
Build and Configure the Model
• Define CNN architecture with three convolutional blocks (32, 64, 128 filters with 3×3 kernels)
• Include max pooling layers after each convolutional block for dimensionality reduction
• Add flatten layer to convert 2D feature maps to 1D vector
• Include dense layer (128 units) with ReLU activation
• Apply dropout (0.5) for regularisation to prevent overfitting
• Add final dense layer with softmax activation for 10-class classification
• Compile model using Adam optimiser, categorical crossentropy loss, and accuracy metric
03
Train and Evaluate the Model
• Configure data augmentation (rotation: 10°, width/height shift: 0.1, zoom: 0.1)
• Set up callbacks: early stopping (patience=5) and learning rate reduction (factor=0.5, patience=3)
• Train model with batch size of 32 for 20 epochs
• Monitor training and validation accuracy/loss curves throughout training
• Use validation data to detect overfitting and adjust hyperparameters
• Evaluate final model using confusion matrix, precision, recall, and F1-score
• Visualise results to identify class-specific performance and areas for improvement

Interpret and evaluate model

I will use the following evaluation metrics to interpret the results of my CNN model:

Accuracy:
Measures the proportion of correctly predicted garment classifications compared to the total number of predictions, providing an overall indication of model performance across all 10 Fashion MNIST categories

Precision, Recall, and F1-score:
These metrics will be obtained from a classification report.

- Precision evaluates how many of the predicted garment categories are actually correct for each class.
- Recall assesses how well the model identifies all true instances of each garment type.
- F1-score combines precision and recall into a single measure to provide a balanced evaluation for each clothing category.

Confusion Matrix:
A confusion matrix will be generated to visualise model predictions versus actual garment labels. This helps to identify which classes are being correctly predicted and where misclassifications occur, particularly between visually similar items such as shirts and T-shirts.

I will classify the model as successful if it achieves 90% or higher validation accuracy whilst maintaining balanced performance across all 10 garment categories, with no individual class falling below 70% precision or recall. This ensures the model generalises well and is not biased towards specific garment types.

Report

Introduction
Explain the goal of the project and outline the CNN model developed to classify Fashion MNIST garment images into 10 categories. Summarise the dataset used (60,000 training images, 10 classes) and
Business Impact and Application
Break down practical applications of the CNN model, such as automated fashion cataloguing, e-commerce product tagging, or retail inventory management. Highlight how accurate garment classification can improve customer experience and operational efficiency in the fashion industry.
Data Cleaning
Describe the preprocessing steps, including normalisation of pixel values to range, reshaping images to (28, 28, 1) for CNN input, and one-hot encoding of labels. Explain how these steps ensured the dataset was ready for model training.
Model Results and Interpretation
Summarise the key performance metrics: 90% validation accuracy, precision/recall/F1-scores per class, and confusion matrix findings. Interpret what these results indicate about the model's effectiveness in distinguishing garments, noting strong performance for most classes and challenges with similar items (Shirt vs T-shirt/top).
Report Mind Map
EDA
Break down the key features of the Fashion MNIST dataset and highlight balanced class distribution (6,000 samples per category), image dimensions (28×28 greyscale), and visual characteristics identified through sample visualisations and statistical summaries.
Model Development and Training
List the CNN architecture (3 convolutional blocks with 32, 64, 128 filters, max pooling, dropout 0.5, dense layers) and training setup, including Adam optimiser, batch size 32, 20 epochs, early stopping, and learning rate reduction callbacks. Summarise how these choices improved training stability.
Feature Engineering and Preprocessing
Explain how pixel normalisation and data reshaping prepared images for CNN layers. Describe the train-validation split (85%/15% stratified) and how data augmentation (rotation, shifting, zoom) was planned to enhance model generalisation.

Question 3

Please find Reponses within the github repo.
https://github.com/just-makab/PDAN8412_POE_PART_3.git

Question 4

https://github.com/just-makab/PDAN8412_POE_PART_3.git

Question 5

Please refer to the Report Document.

Bibliography

Ultralytics, 2019. *Fashion-MNIST Dataset*. [Online]
Available at: https://docs.ultralytics.com/datasets/classify/fashion-mnist/
[Accessed 18 November 2025].

GeeksforGeeks, 2025. *Image Classification using CNN*. [Online]
Available at: https://www.geeksforgeeks.org/machine-learning/image-classifier-using-cnn/
[Accessed 15 November 2025].

Kayed, M., Anter, A. & Mohamed, H., 2020. *Classification of Garments from Fashion MNIST Dataset Using CNN LeNet-5 Architecture*. [Online]
Available at: https://2024.sci-hub.se/8156/a865d1e4094b84179e3354c97f2cb0ce/kayed2020.pdf
[Accessed 15 November 2025].

Zalando, R. & Chris, C., 2017. *Fashion MNIST*. [Online]
Available at: https://www.kaggle.com/datasets/zalando-research/fashionmnist
[Accessed 17 November 2025].

Byeon, E., 2020. *Exploratory Data Analysis Ideas for Image Classification*. [Online]
Available at: https://towardsdatascience.com/exploratory-data-analysis-ideas-for-image-classification-d3fc6bbfb2d2/
[Accessed 16 November 2025].