Project Report On

Product Description Generation Using Image And Text Analysis

Submitted in partial fulfillment of the requirements for the award of the degree of

Bachelor of Technology

in

Computer Science And Engineering

By

Nayana V Reji (U2103151)

Nikita Alex (U2103156)

Niranjan G Das (U2103157)

Praveen Raj C S (U2103166)

Under the guidance of

Ms. Janani K

Assistant Professor

Computer Science And Engineering
Rajagiri School of Engineering & Technology (Autonomous)
(Parent University: APJ Abdul Kalam Technological University)
Rajagiri Valley, Kakkanad, Kochi, 682039
April 2025

CERTIFICATE

This is to certify that the project report entitled ***”Product Description Generation using Image and Text Analysis”*** *is a bonafide record of the work done by* ***Nayana V Reji (U2103151), Nikita Alex (U2103156), Niranjan G Das (U2103157), Praveen Raj C S (U2103166)****, submitted to the Rajagiri School of Engineering & Technology (RSET) (Autonomous) in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology (B. Tech.) in Computer Science and Engineering during the academic year 2024-25.*

Ms. Janani K
Project Guide
Assistant Professor
Dept. of CSE
RSET

Ms. Sangeetha Jamal
Project Coordinator
Assistant Professor
Dept. of CSE
RSET

Dr. Preetha K G
Professor and HoD
Dept. of CSE
RSET

ACKNOWLEDGMENT

We wish to express our sincere gratitude towards **Rev. Dr. Jaison Paul Mulerikkal CMI**, Principal of RSET, and **Dr. Preetha K G**, Professor, Head of the Department of Computer Science and Engineering for providing us with the opportunity to undertake our project, **Product Description Generation using Image and Text Analysis**.

We are highly indebted to our project coordinator, **Ms. Sangeetha Jamal**, Assistant Professor, Computer Science and Engineering, for her valuable support.

It is indeed our pleasure and a moment of satisfaction for us to express our sincere gratitude to our project guide **Ms. Janani K**, Assistant Professor, Computer Science and Engineering for her patience and all the priceless advice and wisdom she has shared with us.

Last but not the least, we would like to express our sincere gratitude towards all other teachers and friends for their continuous support and constructive ideas.

Nayana V Reji

Nikita Alex

Niranjan G Das

Praveen Raj C S

Abstract

This project proposes to create a system that assists e-commerce vendors in producing clear and consistent product descriptions. Through the analysis of product images and correlation with limited textual data, the system produces elaborate descriptions that enhance product visibility and minimize errors. It corrects the widespread phenomenon of unclear, incomplete, or inconsistent product descriptions, which in most cases result in unhappy buyers and poor selling performance. Focusing on buyer experience and seller convenience, the system ensures key product features are captured and listed distinctly. The site is created to be simple and easy to keep, avoiding complexity that could require technical knowledge. Such automation not only saves time but also helps with consistency across the listings. At last, the project aims at simplifying listing a product, enhancing discoverability, and supporting better customer decision-making through proper and persuasive product descriptions.

Contents

List of Abbreviations

NLP - Natural Language Processing

SEO - Search Engine Optimization

CNN - Convolutional Neural Network

LSTM - Long Short-Term Memory

OCR - Optical Character Recognition

BLEU - Bilingual Evaluation Understudy

METEOR - Metric for Evaluation of Translation with Explicit ORdering

ROUGE - Recall-Oriented Understudy for Gisting Evaluation

CIDEr - Consensus-based Image Description Evaluation

RNN - Recurrent Neural Network

IMF - Image Multi-Feature

SDCM - Siamese Difference Captioning Model

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Product descriptions hold a significant position in e-commerce competition to convert potential buyers, improve search engine ranking, and increase the overall experience of the user. With accurate and detailed product descriptions, customers are able to make wise buying decisions, thus minimizing the number of product returns and dissatisfaction. But manually creating these descriptions is a tedious job that requires much time and effort from the sellers, particularly for mass online stores who keep introducing new products in their inventory.

Today, most sellers find it difficult to generate rich and SEO-friendly descriptions, and as a result, their product listings end up being ambiguous, inconsistent, or incomplete. This adversely affects sales and customer satisfaction. Manually writing descriptions also brings in human errors, which makes it hard to have consistency in product listings.

In response to these challenges, this project suggests an intelligent system that generates product descriptions automatically through the combination of image and text analysis.The automation of this process will decrease substantially the effort and time that sellers need to spend in order to list products and enhance the quality and consistency of product information in online stores.

1.2 Problem Definition

In the competitive online marketplace, precise product descriptions are crucial for Search Engine Optimization as well as customer satisfaction. Most sellers find it difficult to develop detailed, uniform descriptions, resulting in poor visibility, ambiguity, and bad customer experiences. Writing these descriptions manually is also time-consuming, particularly for sellers dealing with large inventories. The market requires a system that assists sellers in creating accurate descriptions to make them more searchable and minimize errors.

1.3 Scope and Motivation

1.3.1 Scope

This project will create an automatic product description generation system through image and text analysis to generate precise, structured, and SEO-optimized product descriptions for e-commerce products. The system will extract critical product attributes from images through deep learning-based feature extraction and inspect text provided by sellers to include further information.It will generate properly structured descriptions to provide clarity, consistency, and completeness. The system will also maximize the search engine descriptions to make the products visible across all the e-commerce platforms. By making the effort and time used to write descriptions seamless through automation, the system will minimize sellers' effort and time invested in business. Last but not least, it will optimize the experience of the customer by delivering the product information in the correct and credible way, reducing confusion and return rates.

1.3.2 Motivation

With the fast-growing e-commerce sector, product descriptions play a crucial role in influencing purchasing decisions, search engine optimization, and customer satisfaction. Manually generating comprehensive and structured descriptions is time-consuming and labor-intensive, especially for sellers who have to manage huge inventories of products.Most sellers find it difficult to maintain consistency, lack of product attributes, and poor search optimization, resulting in decreased visibility, lower sales, and increased return rates due to incorrect or unclear descriptions.

This project is inspired by the importance of automating and optimizing the description creation process to be quicker, more efficient, and highly precise. By utilizing image and text analysis, vendors are able to create stable and well-formed descriptions easily to have improved search visibility, enhance user experience, and achieve greater conversions. Besides, the system will assist in eliminating human error, achieving consistency, and achieving greater e-commerce efficiency, bringing benefits to sellers and customers.

1.4 Objectives

The overall goal of the project is to create an automated product description generation system that enhances e-commerce efficiency and precision. The precise goals are:

- Automate Description Generation – Reduce human involvement by developing a system that can produce structured and correct product descriptions autonomously.
- Extract Product Features from Images – Gather useful product features from images using deep learning methods.
- Add Seller-Supplied Information – Handle text input from sellers to incorporate extra product information that cannot be ascertained from images.
- Optimize for SEO – Generate descriptions optimized for search visibility on online shops and search engines.
- Improve Consistency and Accuracy – Reduce human errors, inconsistencies, and omitted product attributes in descriptions.
- Enhance Customer Experience – Present detailed, concise, and trustworthy descriptions to inform customer buying decisions and minimize return rates and confusion.
- Enable Scalability – Allow the system to accommodate large inventory quantities easily and thus be easily deployable on both small and big e-commerce sites.

By attaining these goals, the project seeks to automate the listing process of the products to better benefit the customers and sellers, as well as maximize overall efficiency in e-commerce.

1.5 Challenges

The main challenges in this project are the proper transformation of product attributes into SEO-optimized descriptions with consistency and clarity. It adds complexity to the system, as the descriptions must meet the varied branding needs and handle diverse product types. Moreover, the project has to address the issues related to processing large volumes of data efficiently to support a seamless seller experience.

1.6 Assumptions

- The system assumes that the provided product images are clear and of good quality.
- The text entered by the seller are accurate to the best of their knowledge with the brand name mentioned first, followed by the clothing type, to enable effective structured text extraction using positional mapping.
- Pre-trained models are effective in feature extraction from images and text.

1.7 Societal / Industrial Relevance

This project is relevant in the e-commerce industry, especially for online marketplace and individual retailers, to increase product exposure and consumer experience. It gets well-crafted, SEO-friendly descriptions that help facilitate product discoverability, potentially leading to higher sales and lower return rates.This offers small business people an accessible tool by which they may compete with large retailers, making the digital marketplace more inclusive. Clearer descriptions of products avoid consumer confusion and ultimately lead to a better purchasing decision and satisfaction from customers.

1.8 Organization of the Report

The prime intention of this report is to provide an overall idea about the development and the implication of an automated product description generation system for electronic commerce platforms. It starts with an introduction which asserts the background, problem state- scope, objectives, challenges, and societal relevance of the project.A literature survey follows, describing the existing models and methodologies, plus pointing out where those gaps are, and how the proposed system aims to fill them in. The requirements and system architecture describes the architecture framework, tools, and methods used and committed to image and text analysis. This is followed by the system implementation chapter where the process of system development, its dataset, and experiment setup are explained which then leads to the results and discussion, assessing the performance of the system.There is a discussion on challenges and limitations as a means of portraying the obstacles experienced and findings arrived at. Finally, the report possesses a conclusion and future work to summarize key findings, contributions, future developments and a list of references.

Chapter 2

Literature Survey

2.1 CaptionNet: Automatic End-to-End Siamese Difference Captioning Model With Attention

2.1.1 Introduction

The purpose of the paper [1] is to create a model which can produce automatically captions specifying what two photographs share in common. Common picture caption-describing a single image in isolation, but this task is aimed at the project of difference captioning where the objective is to compare two images and produce a text output that highlights their distinct features. The motivation behind this approach is to to address scenarios demanding comparative analysis, say comparing product variations spotting changes in medical imaging, or monitoring changes in surveillance footage. The authors suggest an end-to-end model architecture that can learn to generate accurate, difference-focused captions, thus widening the scope of automatic captioning to applications involving multiple images

2.1.2 Methodology

The methodology is based on a Siamese network architecture that feeds two images and then identifies what is different. This network essentially consists of two identical subnetworks that have shared parameters across them, implying that they obtain similar feature representations from each input image. Finally, the attention mechanism is applied by focusing on which regions of both images are probably going to exhibit differences, then the network employs these to formulate captions that make more emphasis in those areas.The architecture is end-to-end, taking in two images directly and outputting a difference-focused caption, thus allowing feature extraction and text generation to happen within the same model pipeline. The attention mechanism improves the accuracy and relevance of the generated captions by focusing on pertinent image regions.

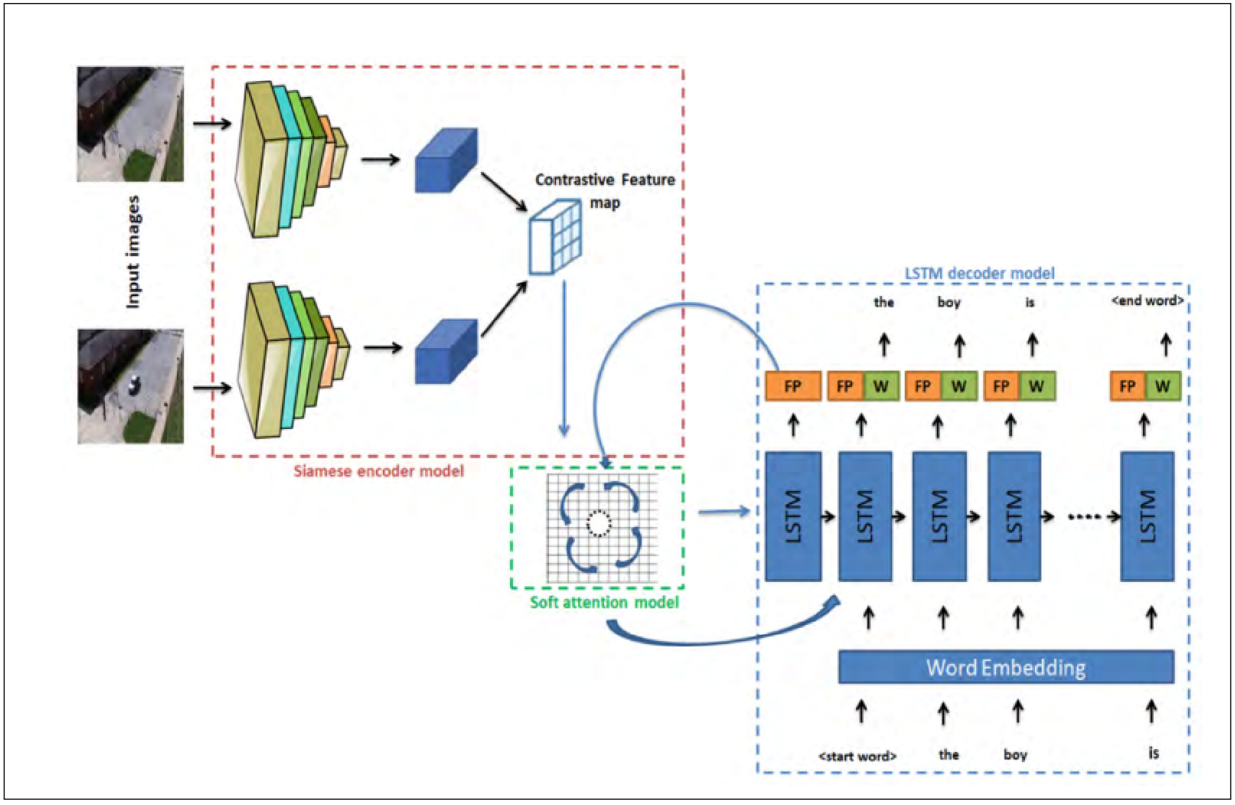


Figure 2.1: SDCM framework

2.1.3 Results

The CaptionNet model demonstrated promising results for the accurate capturing and description of differences between image pairs. The attention mechanism effectively enhanced the model's capacity to identify differences and highlight meaningful distinctions between images. Evaluating the model using standard metrics for captioning, such as BLEU and METEOR scores, CaptionNet showed competency performance, especially in comparative analysis contexts. The results indicate that the model could be a useful tool for applications requiring automatic difference captioning, with fine-tuning potential to specific domains like product comparison or change detection.

2.2 Image and Video Captioning for Apparels Using Deep Learning

2.2.1 Introduction

This paper [2], develops a deep learning model for the generation of captions for apparel images and videos to automate the description generation process for fashion products. This work will be used to support e-commerce and retail industries, which rely on accurate and detailed descriptions for the enhancement of customer experience and increased online sales. The motivation here is to lessen the effort towards the manual description of products. In addition, the quality of descriptions will increase, and they would be more standardized. By enabling caption generation and thereby standardizing descriptions, making products searchable with ease, and achieving a better alignment between customer inquiries and relevant items, the model will be quite useful. This paper introduces an all-inclusive deep learning methodology specific to the apparel domain and addresses challenges unique to the visual nature of fashion products.

2.2.2 Methodology

The approach involves a CNN-LSTM framework. CNNs retrieve rich visual details from images of clothing and movie frames of video. LSTM networks produce text descriptions from such visual details. Transfer learning is adopted, including pre-trained CNN models being available for use, and fine-tuned for fashion images to pick up on minute detail, e.g., texture of fabrics, hue, and motif. The pipeline is also flexible and can handle single images or sequences in video frames. For videos, the model uses temporal attention mechanisms such that it emphasizes consistent attributes across frames, thereby producing coherent and contextually correct captions.

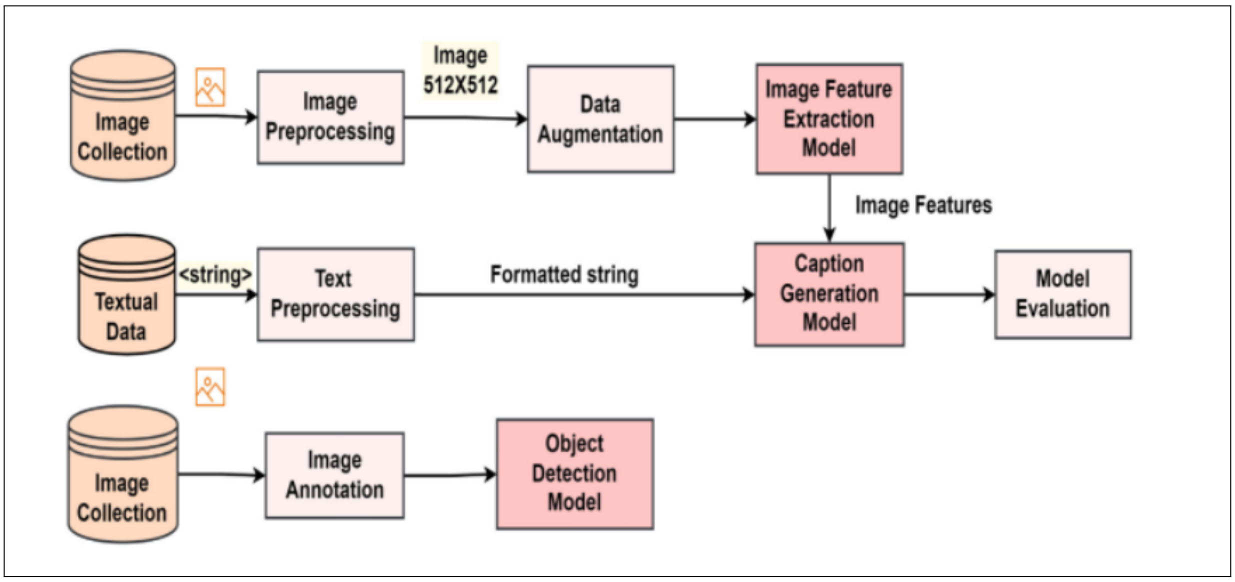


Figure 2.2: Architecture for apparel based Image and Video Caption Generation system

2.2.3 Results

The model performs very well on clothing datasets with high caption accuracy for describing style, color, and fit. The captions were evaluated using standard metrics BLEU, ROUGE, and CIDEr. The resulting captions are descriptive and pertinent. On the video captioning side, the proposed temporal attention mechanism was effective by generating smooth and comprehensive descriptions, even in scenarios with multiple items or movement. The results confirm this model's capability to automate and improve the quality of product descriptions for apparel in online marketplaces, hence demonstrating applicability for image-based as well as video-based product listings.

2.3 Image Multi-Feature Fusion for Clothing Style Classification

2.3.1 Introduction

This paper [3] attempts to classify clothing style by integrating multiple visual features from images, with the aim of improving the accuracy and robustness for fashion categorization. The objective of this research is to enhance the performance of classification models for style classification, especially when individual features like color or texture alone might not fully represent complex styles. This paper is motivated by the rapidly growing demand for e-commerce applications with the ability to classify with precision since one piece of clothing may convey more than one style attribute; for example, it can be vintage, casual, or formal. Combining several visual cues to achieve an exhaustive representation of every clothing style will make this approach more valid for real-world applications like fashion recommendation systems.

2.3.2 Methodology

The methodology involves using a multi-feature fusion approach on features extracted using different neural networks. First, a Convolutional Neural Network (CNN) can be used in order to be able to take core visual elements, involving texture and shapes. In this regard, use of a color histogram and features of edges enables specific information based on color distributions and garment outlines. All of these features are then combined to a unified feature vector, used for the last model, most of the times fully connected or an ensemble classifier. It is able to learn the interdependency between multiple visual attributes as feature fusion takes place, therefore enabling the system to distinguish subtler style changes more effectively. It also uses feature selection techniques that only retain the most relevant features for the system, optimizing the model's accuracy and efficiency.

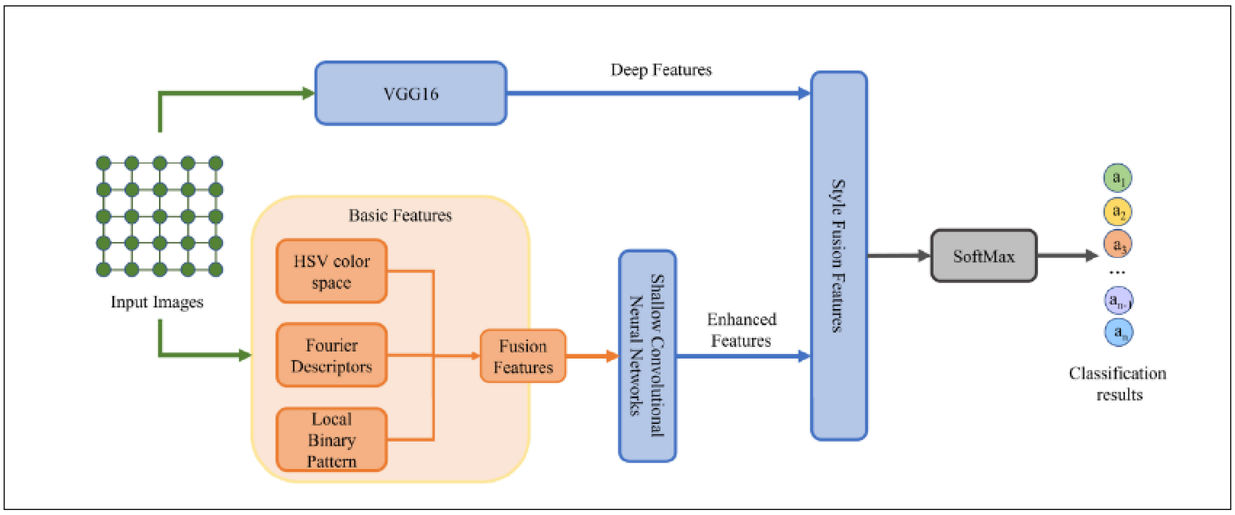


Figure 2.3: Schematic diagram of IMF model

2.3.3 Results

This multi-feature fusion approach increased the accuracy of classification much better than that achieved in single-feature models. The results on the experiments conducted on the fashion datasets are good in terms of precision and recall, especially between visually similar styles.It is seen that texture, color, and edge features fuse better to bring about a finer distinction between styles' characteristics. This approach shows promise for application in e-commerce and fashion retail, as the accurate classification of styles is necessary for delivering better search and recommendation experiences to the customer.

2.4 Switching Text-Based Image Encoders for Captioning Images With Text

2.4.1 Introduction

This paper [4] presents a novel approach to generating captions for images containing embedded text, such as signs, product labels, and documents. This paper addresses the task of understanding image captions in scenes where text richness is present within the images that require understanding visual content and, more importantly, reading the text as well to derive meaningful captions for them. Improved captioning models for images arise from this direction of work mainly in scenarios with embedded text within images, so as to contextualize understanding about educational materials, advertisements, or other retail catalogs. The proposed system introduces a specialized encoder that dynamically switches between visual and text-based feature extraction, thereby providing a more accurate depiction of text-bearing images.

2.4.2 Methodology

The methodology involves the switching mechanism of traditional image encoders and text-based image encoders to capture the visual and textual elements. In this model, a CNN-RNN architecture was used to process visual features. Optical Character Recognition is used for extracting text. The switching mechanism controls the importance of the detected text and chooses between the visual encoder and the text-based encoder depending on its significance to the context of the overall image. The relevant features are then passed into a language model to produce captions that combine the visual as well as textual content. This adaptive encoding enables the model to rank text when it is most important to the meaning of the image, resulting in richer and more contextually appropriate captions.

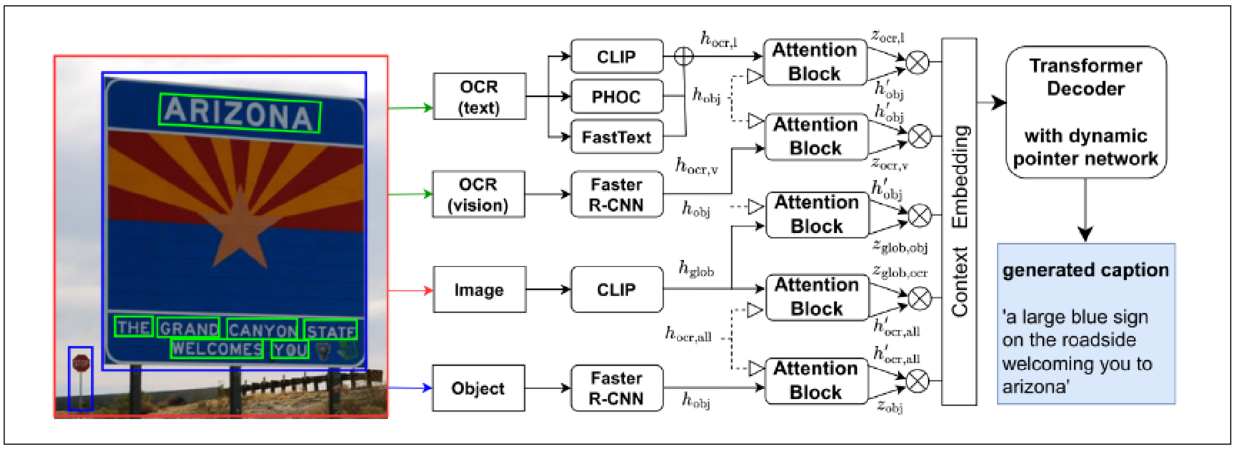


Figure 2.4: Architectural diagram of Multimodal Transformer based model

2.4.3 Results

Switching between text-based and encoder-based techniques improved the performance of the image captioning tasks on image-text datasets by considerably higher percentages of accuracy as opposed to earlier baselines, including more accurate representations in captions especially if text is visible and content-dependent. Performance metrics like BLEU, METEOR, and CIDEr amply demonstrate that this technique surpasses conventional image captioning models significantly on text-image data. This adaptive technique holds highly promising applications in retail and advertising industries, where text contained within images is significant for the communication of product information and brand messages.

2.5 Summary and Gaps Identified

Title	Advantages	Disadvantages
CaptionNet:Automatic End-to-End Siamese Difference Captioning Model With Attention	- Leverages Siamese architecture for comparing image pairs. - End-to-end model simplifies pipeline. - Effective attention mechanism focuses on relevant differences.	- Limited to difference captioning. - May struggle with ambiguous or subtle differences between images. - Needs domain-specific fine-tuning for best results.
Image and Video Captioning for Apparels Using Deep Learning	- Employs CNN-LSTM for both image and video captioning. - Effective transfer learning for fashion images. - Temporal attention improves video captioning.	- Model complexity could increase for large video datasets. - Performance may degrade on less structured or inconsistent fashion datasets.
f**Image Multi-Feature Fusion for Clothing Style Classification**	- Combines multiple visual features (texture, color, edges). - Improves accuracy in fashion style classification. - Effective in distinguishing subtle style variations.	- Feature fusion can increase model complexity. - May require large datasets for optimal feature learning. - May struggle with highly diverse or complex styles.
Switching Text-Based Image Encoders for Captioning Images With Text	- Dynamic encoder switching optimizes feature extraction based on image context. - Improves caption accuracy for text-heavy images. - Integrates OCR and CNN-RNN.	- Relies heavily on accurate OCR for performance. - May not handle images with noisy or unclear text well. - Computational overhead due to switching mechanism.

Table 2.1: Summary of approaches in Image Captioning and Classification models

2.5.1 Gaps Identified

1. **Limited Generalization:** The majority of models, including CaptionNet, are task-specific (e.g., difference captioning, which limits their ability to generalize to more general image types and uses.

2. **Challenges in Video Context:** Existing models struggle to maintain coherence in changing and complex video images, e.g., more than one moving object or changing lighting.

3. **Feature Fusion Overhead:** Multi-feature fusion methods add computational overhead and need large datasets, which makes them resource-hungry and hard to scale.

4. **OCR Dependence:** OCR-dependent models are much dependent on text extraction accuracy, but noisy or low-quality OCR errors can appreciably decrease captioning quality.

5. **Scalability and Real-Time Performance:** Scalability and real-time performance have been difficult for most image and video captioning models to achieve, especially for large-scale applications such as e-commerce or surveillance.

Chapter 3

Requirements

3.1 Software and Hardware Requirements

3.1.1 Software Requirements

1. **Development Environment:** VS Code is used for writing, debugging, executing code efficiently and for structured development and deployment.
2. **Libraries:** The following libraries are used to build and train machine learning models:
    - **TensorFlow:** Provides deep learning capabilities for training and deploying neural networks.
    - **Torch:** A deep learning library used for building and training neural networks, known for its flexibility and dynamic computation graph.
    - **Hugging Face Transformers:** Supports pre-trained transformer models for NLP and vision tasks.
    - **OpenCV:** Facilitates image processing operations, including feature extraction and data augmentation.

3.1.2 Hardware Requirements

1. **Cloud Platforms:**
    - **Kaggle:** Used for training models on publicly available datasets with free GPU/TPU access.
    - **Google Colab:** Provides cloud-based GPU acceleration for training deep learning models.

Chapter 4

System Design

4.1 System Architecture

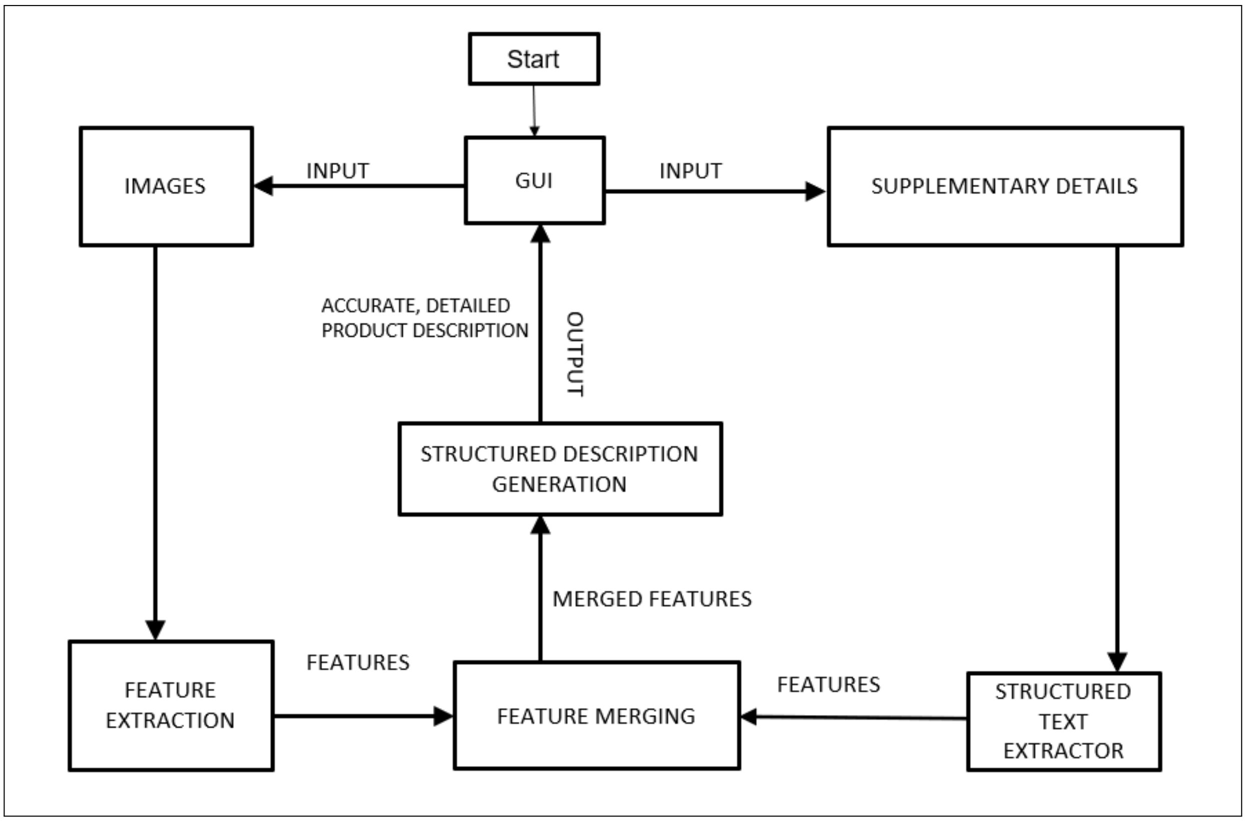


Figure 4.1: System Architecture

The system is designed to produce structured and accurate product descriptions by integrating image-based feature extraction and natural language processing (NLP). The workflow begins with two main inputs: product images and other textual data. These inputs are captured through a graphical user interface (GUI), which is an interactive platform for users to enter product-related data. The images are feature-extracted where important visual descriptors like style, material, and design patterns are detected. At the same time, the supporting details are processed with NLP so that suitable textual attributes are extracted from the given descriptions.

Once features are extracted, they are fed into the feature merging module that fuses visual features from images with textual features extracted from accompanying details. Here, all accessible information is processed before creating the final product description. The fusion process is central to solving disagreements between extracted features, favoring precision, and ensuring the description is comprehensive yet structured.

Lastly, the combined features are input to the structured description generation module, where a properly formatted, detailed product description is generated. This is then presented via the GUI, giving the user a true representation of the product, resulting in improved search optimization and enhanced product listings for e-commerce sites.

4.2 Component Design

4.2.1 Image Feature Extraction Module

Objective of the image feature extraction module is to extract characteristic features such as shape, fabric and pattern from clothing images. Dataset used in this module is DeepFashion-MultiModal dataset is a large-scale high-quality human dataset with rich multi-modal annotations. It contains 44096 images and its corresponding shape, fabric and pattern labels. The CNN Model used in this module is a pretrained ConvNeXt model fine-tuned for attribute classification. Input to the model is the apparel image from which features are extracted and output from the model is the extracted shape, fabric and pattern attributes.

4.2.2 Structured Text Extractor

The objective of the project is to extract structured product information—type of garment and brand names—solely from unstructured product descriptions using the help of a Structured Text Extractor. This is achieved by training a tailored model with spaCy in such a way that it is able to properly identify and extract these entities. The training material is manually annotated product descriptions where each brand name and garment type is annotated within the description. A blank spaCy English model is initialized, and Structured Text Extractor is incorporated and trained from scratch with the labeled dataset. The model is trained in an iterative process using stochastic gradient descent, and a dropout rate is introduced for better generalization and prevention of overfitting. After training, the model accepts raw seller-input descriptions and gives the identified brand and type as output.

4.2.3 Feature Merging Module

The system combines two models to produce structured product descriptions. The Structured Text Extractor model identifies important textual information, including brand name and clothing type, from the seller-provided product description. At the same time, a ConvNext (CNN) model inspects the product image to predict clothing attributes such as sleeve length, fabric type, pattern, neckline, and clothing length. The results from both models are then merged and arranged in a formatted text prompt with all the details to generate an appropriate and complete product description.

4.2.4 Structured Description Generation Module

The framework tries to produce structured product descriptions by integrating different apparel features. A labelled dataset is used to fine-tune a Transformer-based FLAN-T5 model to generate the structured descriptions efficiently. The dataset contains features like brand, type, sleeve length, neckline, fabric, and pattern along with their respective descriptions. The input to the model is a text prompt containing the combined features achieved from image feature extraction and text extractor module, and the output is a highly structured and informative product description that best represents the apparel.

4.3 Expected Output

The user experience on e-commerce platforms can be greatly improved by a system that creates logical and structured product descriptions based on both textual and visual inputs. This method enhances the efficiency of searches by delivering optimal and correct descriptions, which enable users to locate pertinent products with ease. By addressing the inconsistencies and gaps that are normally present in seller-supplied information, the system ensures a higher level of accuracy and clarity in product listings.

4.4 Use-Case Diagram

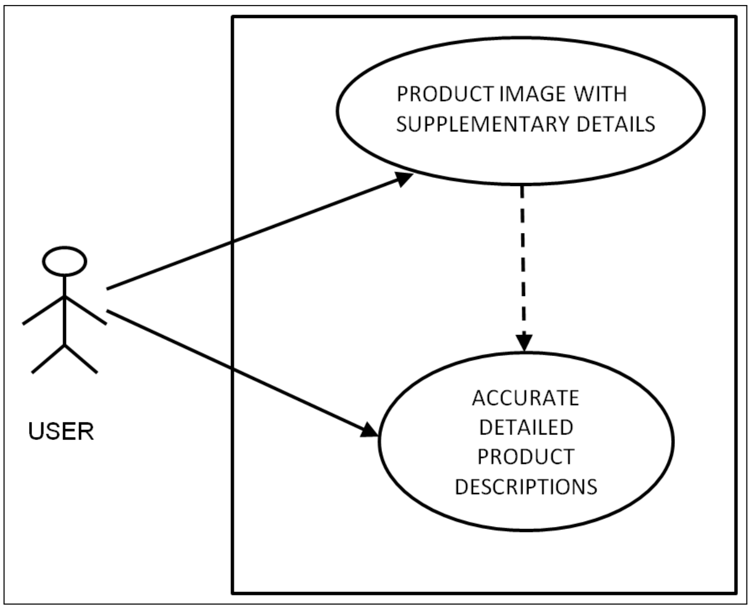

PRODUCT IMAGE WITH SUPPLEMENTARY DETAILS
USER
ACCURATE DETAILED PRODUCT DESCRIPTIONS

Figure 4.2: Use-Case Diagram

4.5 Sequence Diagram

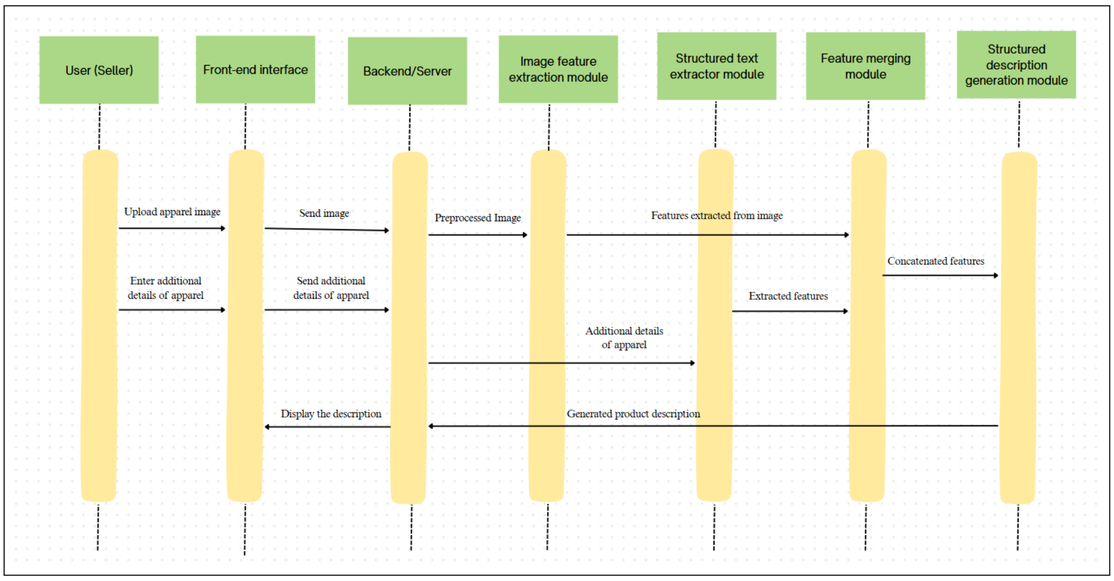

User (Seller)
Front-end interface
Backend/Server
Image feature extraction module
Structured text extractor module
Feature merging module
Structured description generation module
Upload apparel image
Send image
Preprocessed Image
Features extracted from image
Concatenated features
Enter additional details of apparel
Send additional details of apparel
Extracted features
Additional details of apparel
Display the description
Generated product description

Figure 4.3: Sequence Diagram

4.6 Gantt Chart

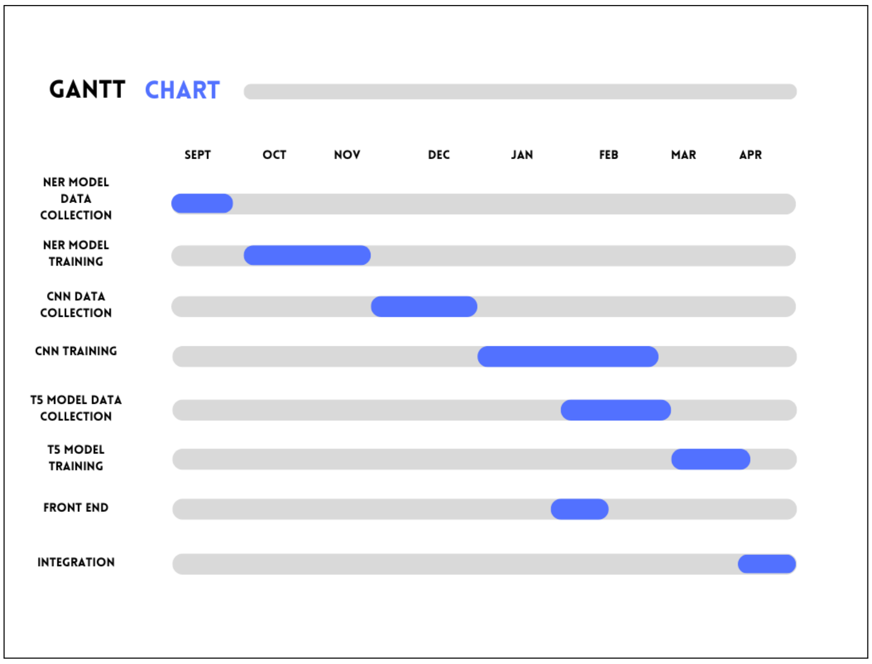


Figure 4.4: Gantt Chart

Chapter 5

System Implementation

This chapter describes in detail the methodology used in this project and how it was implemented. It details the working of the prototype developed as part of this project.

The entire application was built using Python. Flask was used for the backend server, while the User Interface was served using HTML, CSS, and JavaScript. Flask is a lightweight framework in Python that aids in web application development, while HTML provides the structure, CSS enhances the styling, and JavaScript adds interactivity to the user interface.

5.1 Proposed Methodology

5.1.1 Image Feature Extraction Module

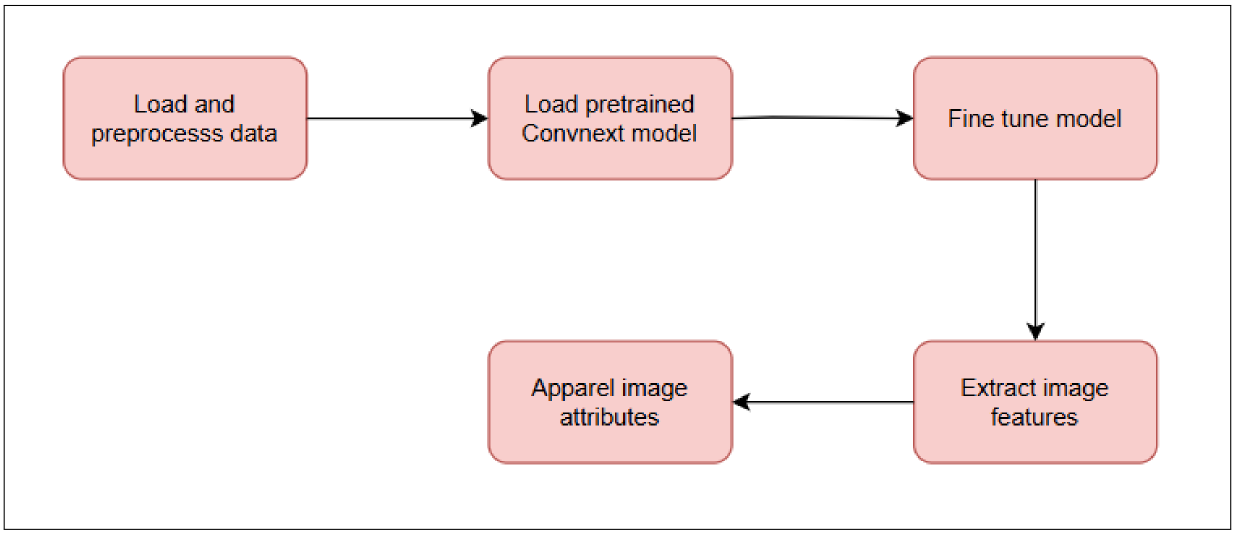


Figure 5.1: Image feature extraction module

This module is designed for fashion clothing attribute extraction using a ConvNeXt-based deep learning model. It extracts significant visual attributes from fashion product images, including shape, fabric, and pattern, by fine-tuning a pretrained ConvNeXtBase model on a labeled fashion dataset. These features can be employed for product cataloging improvement, search optimization, and availability enhancement in e-commerce platforms.

It begins with loading and preprocessing the data in which the system loads images and label files of annotation with labels including shape, fabric, and pattern.It then goes on mapping images to their labels to form a labeled dataset. For correct training, the dataset is split between training and test datasets in the ratio of 80-20. Data augmentation techniques using random flipping, rotation, zoom, and changing contrast are also employed to improve the robustness and generalizability of the model. The labels are transformed into one-hot encoded format for enabling multi-label classification.

The model is constructed using ConvNeXtBase, an advanced CNN structure pre-trained on ImageNet. For adapting it to fashion attribute extraction, the model is altered to add a multi-output classification head, which makes predictions of shape, fabric, and pattern attributes. The ConvNeXt layers are first frozen, with only the new classification layers trainable. Following the first stage, selectively stacks of ConvNeXt are fine-tuned for enhancing domain adaptation. Training is done under Categorical Crossentropy loss with label smoothing, optimizing the model with the Adam optimizer and initializing learning rate to 0.0005, which was subsequently modified to 0.0001 for fine-tuning.It first uses the output of the base model in the form of high-dimensional feature maps that reflect important image patterns. Since the feature maps are kept in spatial shape, a Global Average Pooling (GAP) layer is applied to flatten them into a 1D vector by shrinking dimensions but not valuable information. Following this comes a densely connected (Dense) layer consisting of 512 neurons with ReLU activation, bringing in non-linearity and enabling the model to learn subtle patterns in the data. Batch Normalization (BatchNorm) is employed to stabilize training, normalizing activations and preventing exploding or vanishing gradients. Finally, a Dropout layer (rate=0.3) is added, dropping 30 percentage of neurons during training to prevent overfitting and improve model generalization. This efficiently structured pipeline transforms raw feature maps into an important, summarized representation for classification. In order to prevent overfitting, early stopping is used.

It then makes use of the output of the base model in the form of spatial high-dimensional feature maps depicting salient image patterns. Since the feature maps are still spatial in nature, a Global Average Pooling (GAP) layer is adopted for converting the same into a 1D vector by losing dimensions but not losing relevant information. This is followed by a densely connected (Dense) layer of 512 neurons using a ReLU activation, which brings in the non-linearity and enables the model to learn sophisticated relationships in the data. Batch Normalization (BatchNorm) is used to stabilize the training, which normalizes the activations and keeps exploding or vanishing gradients in check. Lastly, a Dropout layer (rate=0.3) is appended, dropping out 30 percentage of neurons at training time randomly to avoid overfitting and enhance model generalization. This well-structured pipeline converts raw feature maps into a significant, summarized representation for classification.

After the feature maps are fine-tuned through the dense layers, the model predicts a few fashion attributes (shape, fabric, and pattern) through separate classification heads. All of these features are modeled as distinct multi-label classification tasks, hence the model simultaneously predicts multiple independent categories.

For each attribute class (pattern, material, and shape), the model constructs a Dense layer with softmax activation. Softmax activation is used to get the model to output a probability distribution over the set of available classes for each attribute. The most probable class is taken as the output class.

5.1.2 Structured Text Extracting Module

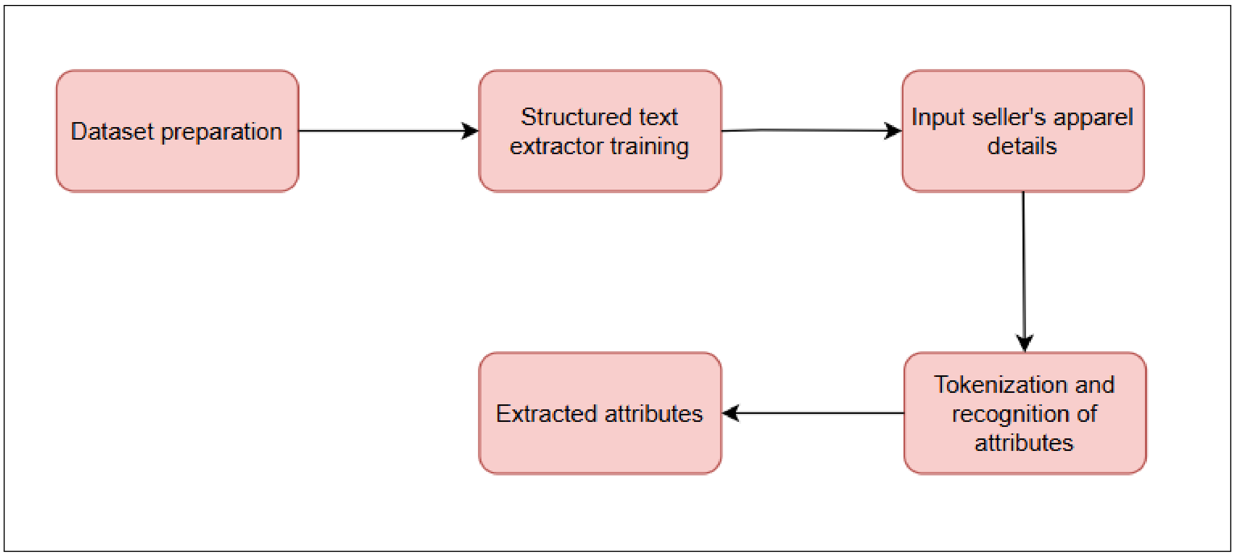

Dataset preparation
Structured text extractor training
Input seller's apparel details
Tokenization and recognition of attributes
Extracted attributes

Figure 5.2: Structured text extracting module

This module focuses on the Structured Text Extracting Module for seller-provided apparel descriptions, aiming to extract crucial product-related features such as brand names and clothing types. The objective is to transform raw, unstructured text into uniform and structured product information that can be directly used to enhance online shopping platforms, improve search relevance, and maintain consistency in product catalogs.

The process begins with dataset preparation, where a labeled collection of training data is curated. Each entry in this dataset contains a text description of a clothing item, along with annotated entities highlighting fashion relevant features such as the brand and type of the item. These annotations are represented using character offsets and are stored in JSON format, making them compatible with spaCy's training structure. These labeled examples serve as the foundational knowledge for the model to learn the association between words and their corresponding fashion related roles.

A blank English model is initialized using spaCy. The module is configured with specific entity labels such as BRAND and TYPE, enabling it to focus on extracting terms that are directly useful in the context of fashion product standardization.

Training is carried out iteratively using the provided labeled data. The dataset is shuffled before each iteration to avoid learning any sequence-specific patterns. The model is trained using stochastic gradient descent (SGD), a popular optimization method that updates the model's internal weights gradually over several passes. A dropout rate of 0.35 is introduced during training to minimize overfitting. Dropout works by randomly omitting a portion of the neural network's features during each pass, encouraging the model to generalize better to unseen examples. This makes the model robust and less sensitive to noisy or redundant patterns in the training data.

Throughout the training process, the model learns to associate patterns of text with the target entity labels. It improves progressively in recognizing consistent patterns in varied sentence structures, even when the same concept is expressed in different ways. For instance, it learns to tag "Nike T-shirt", "Adidas track pants", or "Fabindia kurta" by identifying the correct brand and clothing type in each case.

By the end of training, the Structured Text Extracting Module is capable of automatically identifying and labeling key fashion-related entities in seller descriptions. This removes the need for manual annotation and provides a streamlined mechanism to convert seller-provided content into structured, searchable data. The final model is then ready to be integrated into applications that require real-time or batch extraction of brand and type information from textual product descriptions.

5.1.3 Feature merging module

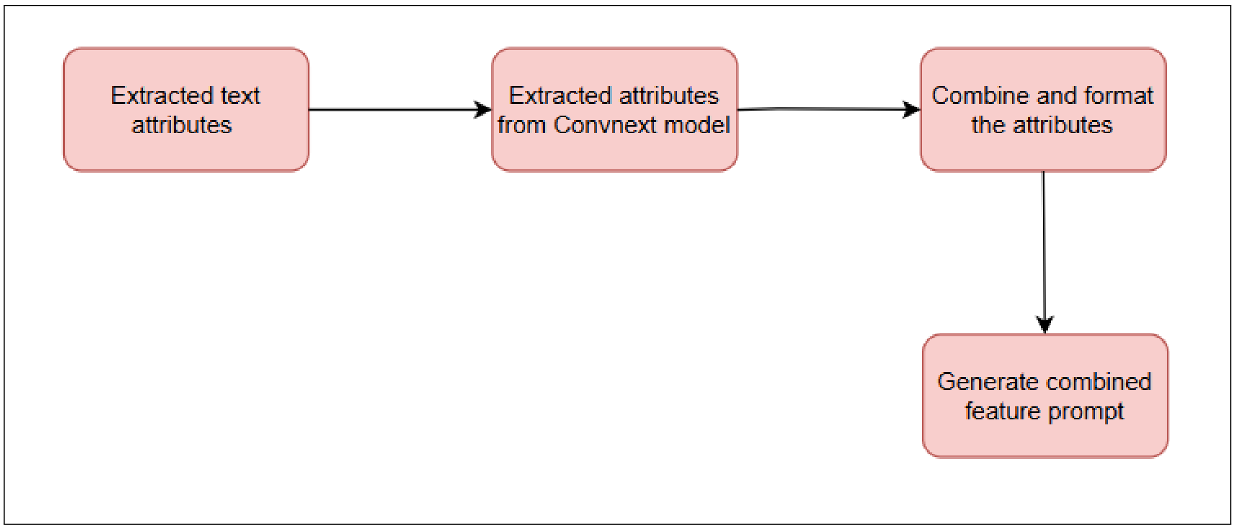

Extracted text attributes
Extracted attributes from Convnext model
Combine and format the attributes
Generate combined feature prompt

Figure 5.3: Feature Merging Module

The framework combines two of the primary models to pull out rich product features. To begin with, a Structured text extractor is used to process descriptions given by sellers for extracting vital textual features like the brand (e.g., Adidas, Nike) and products (e.g., T-shirt, jeans). This helps to capture the structured information from seller descriptions accurately. Simultaneously, a ConvNeXt model, which has been trained with image-based analysis, derives visual features such as sleeve length, neckline, material type, and pattern from the images of the products. This helps in detecting certain clothing attributes that are not explicitly mentioned in the text description.

After the two models obtain their individual attributes, the system transforms them into a structured input prompt that the T5 model uses to generate a detailed product description. The structured input prompt contains information such as brand, category, sleeve length, clothing length, material, and patterns, resulting in a full and rich description. Product listability and searchability are enhanced on online marketplaces using text and image data.

5.1.4 Structured description generation module

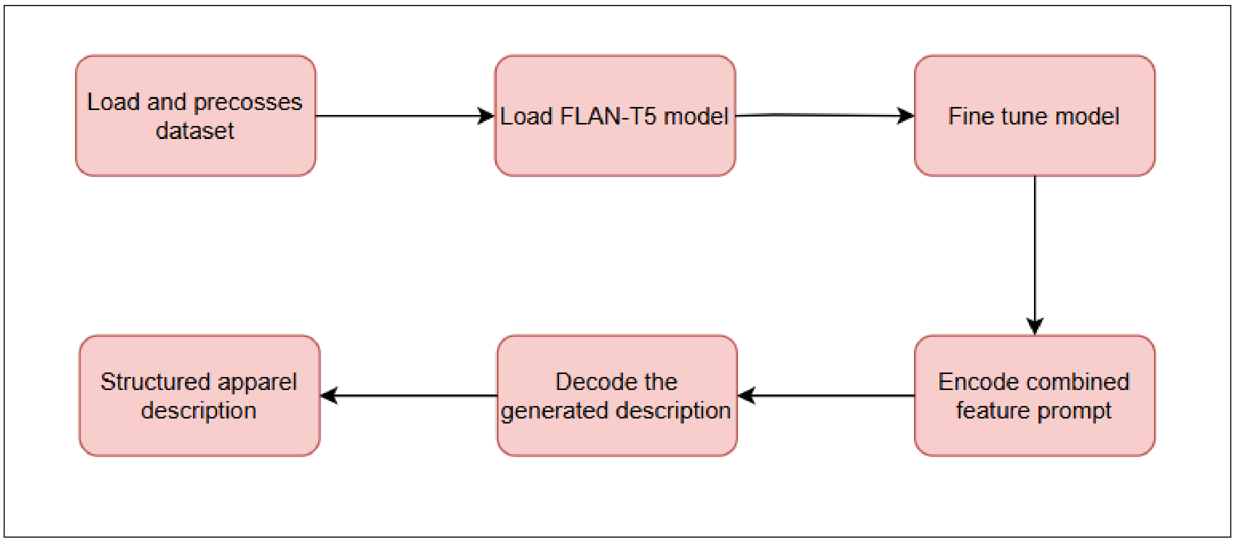

Load and precosses dataset
Load FLAN-T5 model
Fine tune model
Encode combined feature prompt
Decode the generated description
Structured apparel description

Figure 5.4: Structured Description Generation Module

The FLAN-T5 Small is an optimized version of Google's T5 (Text-to-Text Transfer Transformer) that is applied to a wide range of natural language processing (NLP) tasks, e.g., text summarization, translation, question answering, and text generation. Compared to standard T5 models, FLAN-T5 (Fine-tuned LAnguage Net) is trained to tackle instruction-based tasks better, i.e., generate answers from structured input prompts. This makes it particularly well-suited to fashion product description generation, where extracted attributes need to be converted into natural language descriptions.

Fine-tuning is achieved by training FLAN-T5 Small on a labeled dataset containing fashion product attributes and corresponding description. The dataset is preprocessed beforehand to ensure that all required attributes, such as brand, type, fabric, pattern, sleeve length, neckline, etc., are formulated into input prompts.The fine-tuning procedure of FLAN-T5 Small is carried out by a clearly defined step-by-step training methodology to generate structured product descriptions. Tokenization is the first step, where the T5Tokenizer is used to tokenize the input prompt (with structured product attributes) and the target description into numerical tokens. The input data is padded and truncated so that all the sequences fall under the model's maximum token size of 128, and the target description is tokenized separately to be used as the desired output during training. Once tokenizing the data is complete, the next operation is dataset preparation, in which the dataset is arranged in the form of input-output pairs. This ensures that all input token IDs are properly aligned with their target label token IDs, helping the model learn the proper mapping between product attributes and natural language descriptions. A test input prompt is specified, including important product features like brand, type, sleeve length, neckline, fabric, and pattern information. This prompt is then tokenized with the T5 tokenizer, transforming the text into numerical input IDs for the model. The input that has been encoded is then input into the T5 model for generating text, wherein it generates a polished product description using beam search decoding (with four beams) to enhance the quality of output and guarantee early stopping when the response is done. Lastly, the generated text is decoded back to human-readable format, giving a structured and comprehensive product description depending on the input attributes.

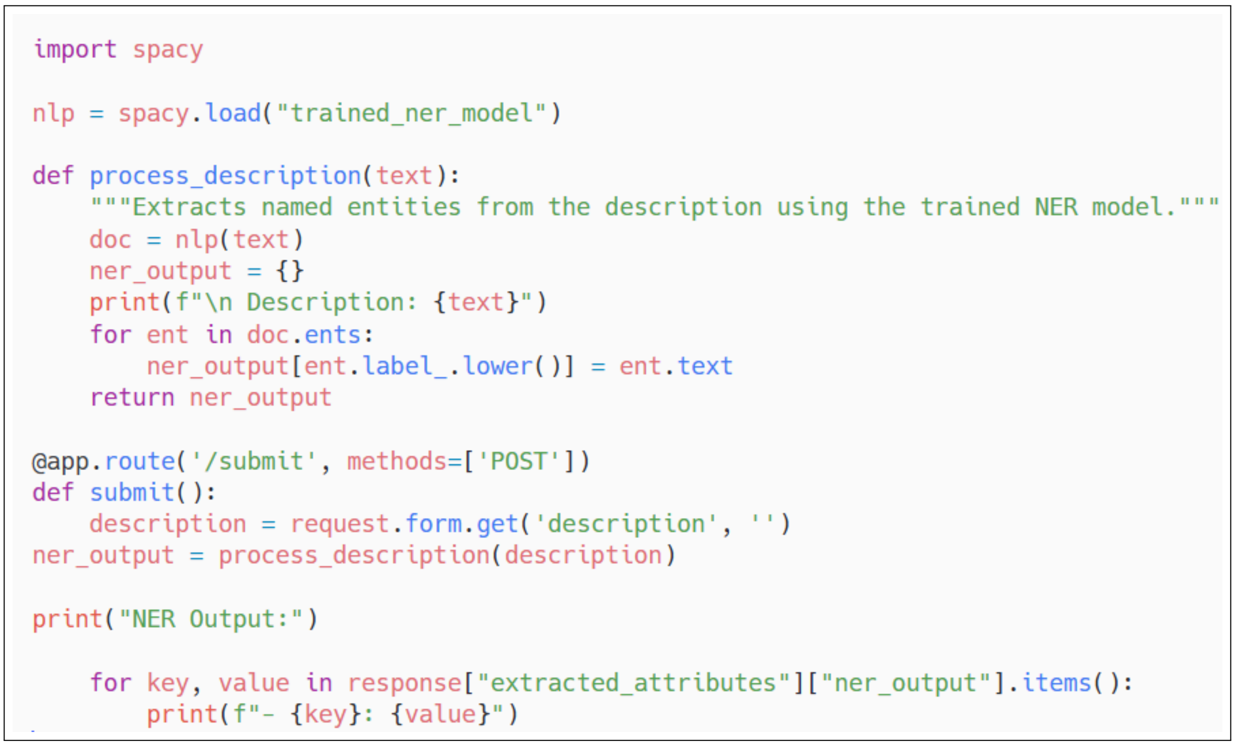
5.2 User Interface Design

The user interface of our project was meticulously designed using HTML, CSS, and JavaScript, ensuring a structured layout, visually appealing design, and dynamic interactivity for a seamless user experience. The apparel image and the seller's details can be input in the interface and the interface displays the structured description after the backend processes both text and image.

5.3 Implementation Strategies

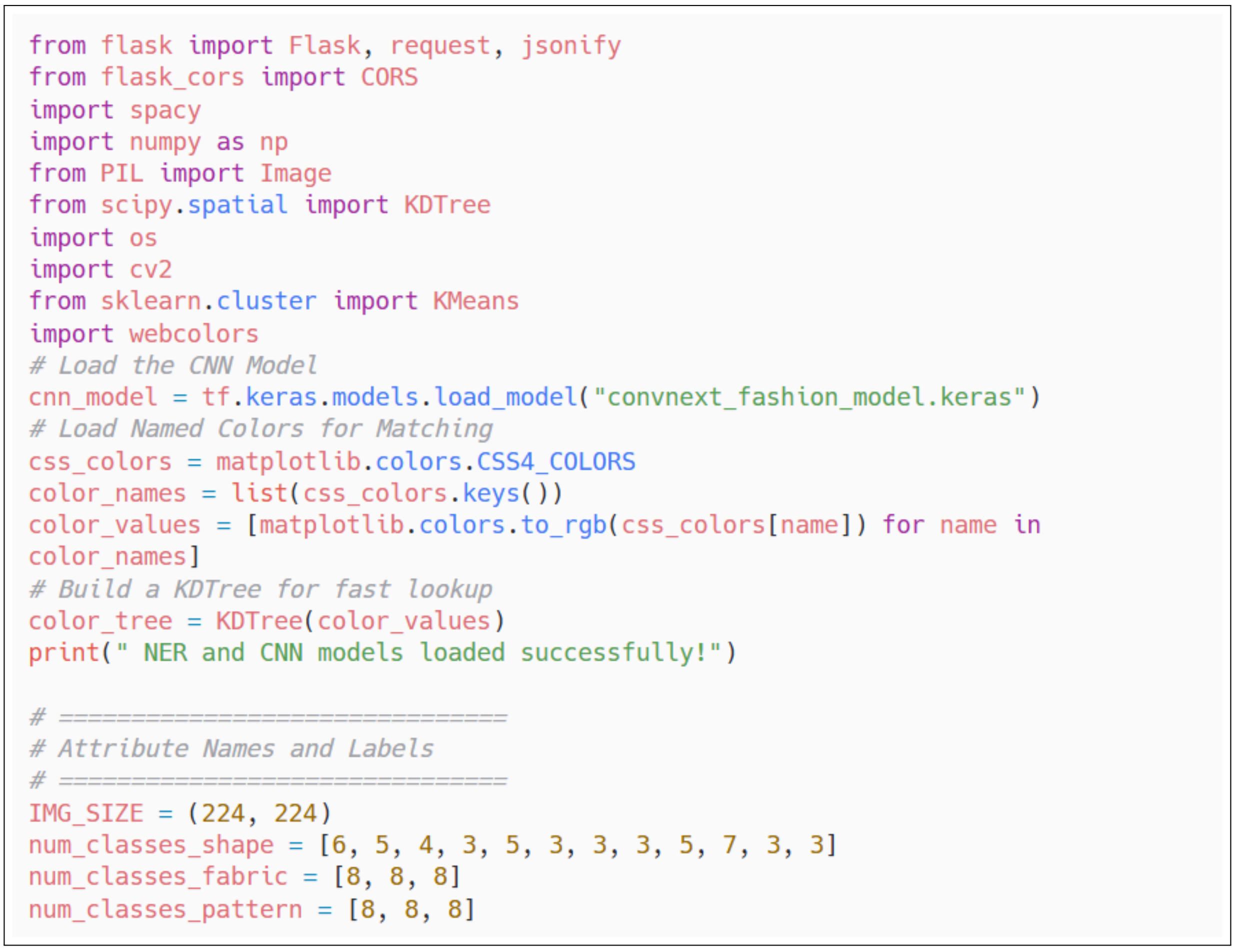
The entire application was built using Python. The following sections contain code snippets for the various modules.

5.3.1 Extracting brand and type of the apparel image using structured text extractor



```
import spacy

nlp = spacy.load("trained_ner_model")

def process_description(text):
    """Extracts named entities from the description using the trained NER model."""
    doc = nlp(text)
    ner_output = {}
    print(f"\n Description: {text}")
    for ent in doc.ents:
        ner_output[ent.label_.lower()] = ent.text
    return ner_output

@app.route('/submit', methods=['POST'])
def submit():
    description = request.form.get('description', '')
ner_output = process_description(description)

print("NER Output:")

    for key, value in response["extracted_attributes"]["ner_output"].items():
        print(f"- {key}: {value}")
```

5.3.2 Extracting features from the image of cloth



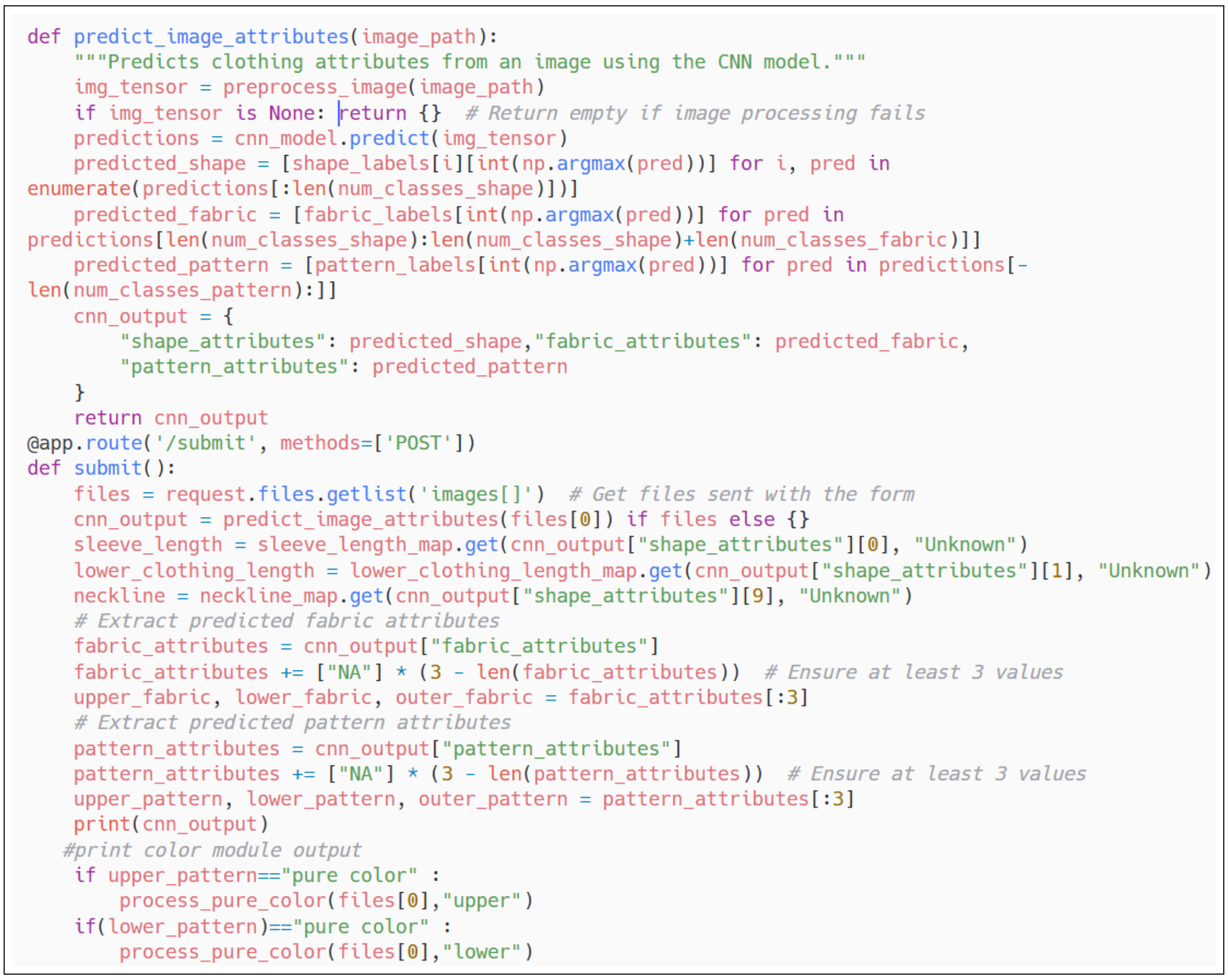
```
from flask import Flask, request, jsonify
from flask_cors import CORS
import spacy
import numpy as np
from PIL import Image
from scipy.spatial import KDTree
import os
import cv2
from sklearn.cluster import KMeans
import webcolors
# Load the CNN Model
cnn_model = tf.keras.models.load_model("convnext_fashion_model.keras")
# Load Named Colors for Matching
css_colors = matplotlib.colors.CSS4_COLORS
color_names = list(css_colors.keys())
color_values = [matplotlib.colors.to_rgb(css_colors[name]) for name in
color_names]
# Build a KDTree for fast lookup
color_tree = KDTree(color_values)
print(" NER and CNN models loaded successfully!")

# ===============================
# Attribute Names and Labels
# ===============================
IMG_SIZE = (224, 224)
num_classes_shape = [6, 5, 4, 3, 5, 3, 3, 3, 5, 7, 3, 3]
num_classes_fabric = [8, 8, 8]
num_classes_pattern = [8, 8, 8]
```

```
shape_labels = [
    ["sleeveless", "short-sleeve", "medium-sleeve", "long-sleeve", "not long-
sleeve", "NA"],
    ["three-point", "medium short", "three-quarter", "long", "NA"],
    ["no", "socks", "leggings", "NA"],
    ["no", "yes", "NA"],
    ["no", "eyeglasses", "sunglasses", "have a glasses in hand or clothes", "NA"],
    ["no", "yes", "NA"],
    ["no", "yes", "NA"],
    ["no", "yes", "NA"],
    ["no", "belt", "have a clothing", "hidden", "NA"],
    ["V-shape", "square", "round", "standing", "lapel", "suspenders", "NA"],
    ["yes", "no", "NA"],
    ["no", "yes", "NA"]
]

fabric_labels = ["denim", "cotton", "leather", "furry", "knitted", "chiffon",
"other", "NA"]
pattern_labels = ["floral", "graphic", "striped", "pure color", "lattice", "other",
"color block", "NA"]
sleeve_length_map = {
    "sleeveless": "Sleeveless",
    "short-sleeve": "Short Sleeve",
    "medium-sleeve": "Medium Sleeve",
    "long-sleeve": "Long Sleeve",
    "not long-sleeve": "Not Long Sleeve",
    "NA": "NA"
}

lower_clothing_length_map = {
    "three-point": "Three-Point",
    "medium short": "Medium Short",
    "three-quarter": "Three-Quarter",
    "long": "Long",
    "NA": "NA"
}
```

```python
neckline_map = {
    "V-shape": "V-Shape",
    "square": "Square",
    "round": "Round",
    "standing": "Standing",
    "lapel": "Lapel",
    "suspenders": "Suspenders",
    "NA": "NA"
}

COCO_CLASSES = {15: "person"}


def get_css_color_mapping():
    """Fetch CSS3 color mapping dynamically."""
    return {webcolors.hex_to_rgb(v): k for k, v in
webcolors.CSS3_NAMES_TO_HEX.items()}

def find_nearest_color(rgb):
    """Find the nearest CSS3 color name for a given RGB value."""
    try:
        return webcolors.rgb_to_name(rgb), rgb  # Exact match if available
    except ValueError:
        pass  # No exact match, find the closest one

    # Convert input RGB to HEX
    hex_code = webcolors.rgb_to_hex(rgb)

    # Get CSS3 color mappings
    css_colors = get_css_color_mapping()

    # Find the closest color using Euclidean distance
    min_distance = float("inf")
    closest_color = None
    for css_rgb, name in css_colors.items():
        distance = sum((c1 - c2) ** 2 for c1, c2 in zip(rgb, css_rgb))  #
Euclidean distance

        if distance < min_distance:
            min_distance = distance
            closest_color = name

    return closest_color, rgb  # Return the closest color


# Image Preprocessing
def color_preprocess_image(image_file):
    image = Image.open(image_file).convert("RGB")
    transform = T.Compose([
        T.Resize((512, 512)),  # Ensure the image is resized to 512x512 for
DeepLabV3
        T.ToTensor(),
    ])
    return transform(image).unsqueeze(0), np.array(image)

def get_segmentation_mask(image_file):
    input_tensor, original_image = color_preprocess_image(image_file)
    with torch.no_grad():
        output = model(input_tensor)['out']  # Model output should now work fine
    mask = torch.argmax(output.squeeze(), dim=0).detach().cpu().numpy()

    return mask, original_image
```

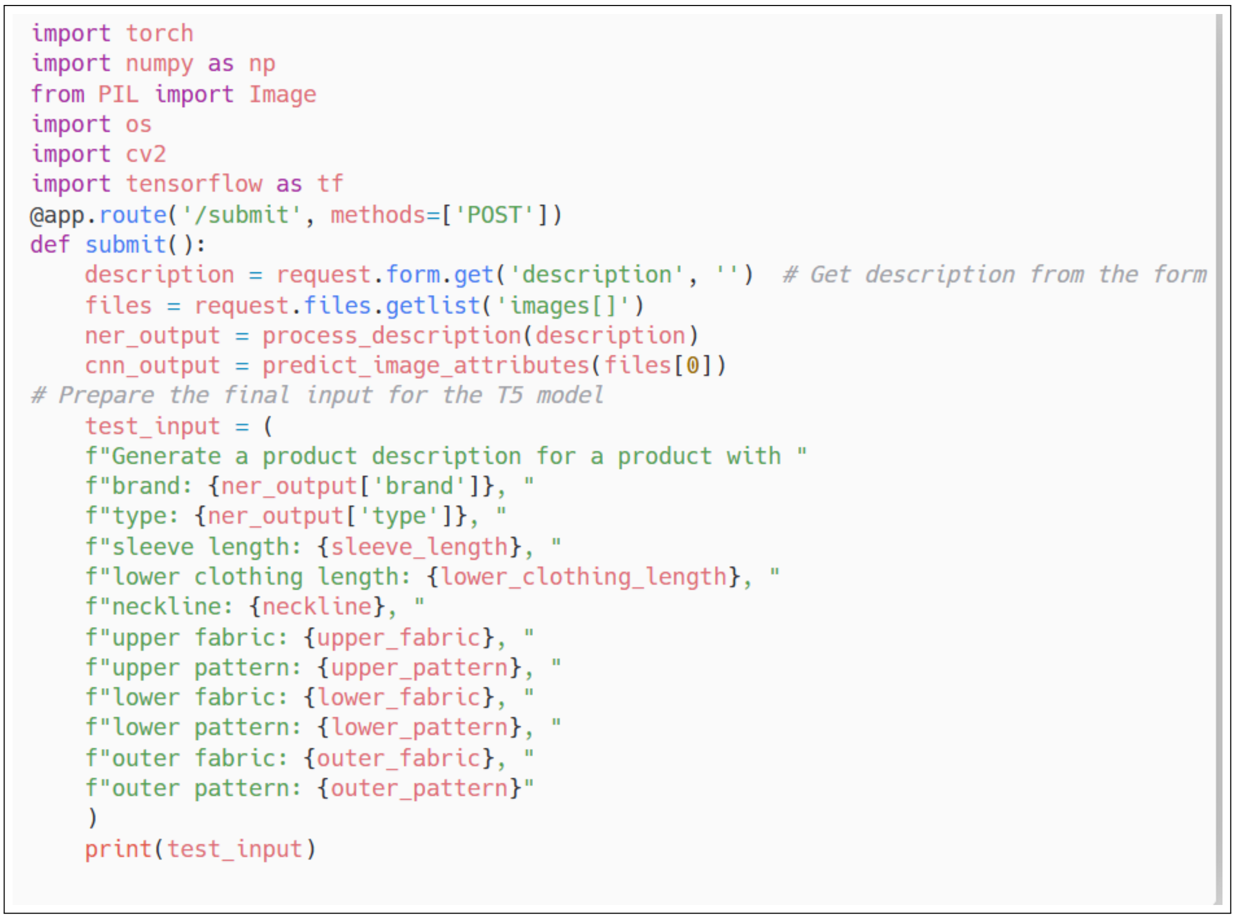
```
def extract_clothing_regions(mask, original_image):
    h, w, _ = original_image.shape
    person_mask = (mask == 15).astype(np.uint8)  # Extract only the "person" mask

    # Resize mask to match the original image size
    person_mask = cv2.resize(person_mask, (w, h), interpolation=cv2.INTER_NEAREST)

    # Define upper and lower body areas (approximate split)
    upper_body_mask = person_mask[:h//2, :]
    lower_body_mask = person_mask[h//2:, :]

    # Convert masks to uint8
    upper_body_mask = (upper_body_mask * 255).astype(np.uint8)
    lower_body_mask = (lower_body_mask * 255).astype(np.uint8)

    # Apply masks to the original image
    upper_clothing = cv2.bitwise_and(original_image[:h//2], original_image[:h//2],
mask=upper_body_mask)
    lower_clothing = cv2.bitwise_and(original_image[h//2:], original_image[h//2:],
mask=lower_body_mask)

    return upper_clothing, lower_clothing

def extract_dominant_color(image, k=3):
    pixels = image.reshape((-1, 3))
    pixels = pixels[pixels.sum(axis=1) > 0]  # Remove black pixels

    kmeans = KMeans(n_clusters=k, random_state=42, n_init=10)
    kmeans.fit(pixels)

    dominant_color =
kmeans.cluster_centers_[np.argmax(np.bincount(kmeans.labels_))]
    return tuple(map(int, dominant_color))  # Convert to integer tuple
```

```
def process_pure_color(image_file,pat):
    mask, original_image = get_segmentation_mask(image_file)
    upper_clothing, lower_clothing = extract_clothing_regions(mask, original_image)
    # Find the nearest color for the upper and lower clothing
    upper_color = extract_dominant_color(upper_clothing)
    lower_color = extract_dominant_color(lower_clothing)
    upper_color_name, upper_nearest_rgb = find_nearest_color(upper_color)
    lower_color_name, lower_nearest_rgb = find_nearest_color(lower_color)

    print("\nColor module:")
    # Get dominant colors
    if(pat =="upper"):

        print(f"Upper Clothing Color: {upper_color}")
        print(f"Nearest Upper Clothing Color: {upper_color_name} with RGB value:
{upper_nearest_rgb}")
    else:

        print(f"Lower Clothing Color: {lower_color}")
        print(f"Nearest Lower Clothing Color: {lower_color_name} with RGB value:
{lower_nearest_rgb}")

def preprocess_image(image_file):
    """Preprocesses an image uploaded via Flask before CNN model inference."""
    try:
        # Read image from file object as bytes
        image_bytes = image_file.read()

        # Decode image using TensorFlow
        image = tf.image.decode_jpeg(image_bytes, channels=3)

        # Resize image
        image = tf.image.resize(image, IMG_SIZE) / 255.0

        # Add batch dimension
        return tf.expand_dims(image, axis=0)
```



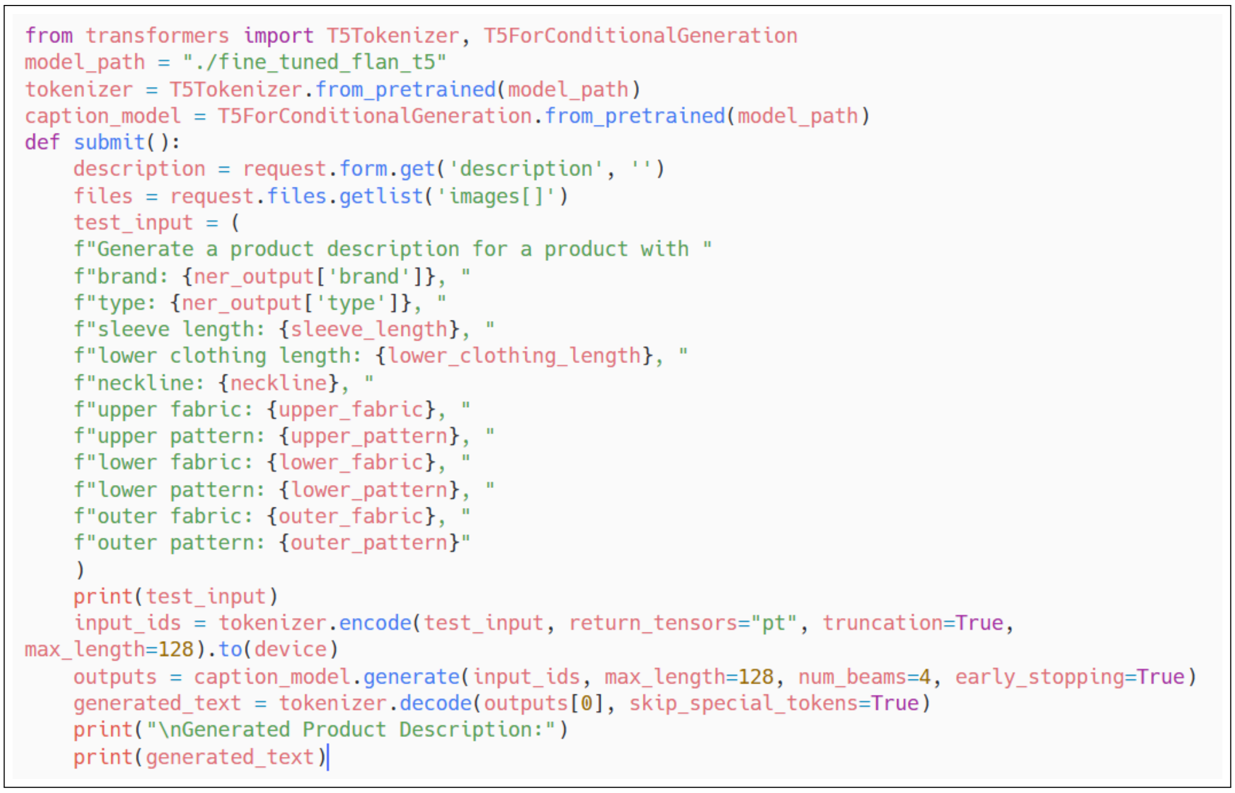
```python
def predict_image_attributes(image_path):
    """Predicts clothing attributes from an image using the CNN model."""
    img_tensor = preprocess_image(image_path)
    if img_tensor is None: return {}  # Return empty if image processing fails
    predictions = cnn_model.predict(img_tensor)
    predicted_shape = [shape_labels[i][int(np.argmax(pred))] for i, pred in
enumerate(predictions[:len(num_classes_shape)])]
    predicted_fabric = [fabric_labels[int(np.argmax(pred))] for pred in
predictions[len(num_classes_shape):len(num_classes_shape)+len(num_classes_fabric)]]
    predicted_pattern = [pattern_labels[int(np.argmax(pred))] for pred in predictions[-
len(num_classes_pattern):]]
    cnn_output = {
        "shape_attributes": predicted_shape,"fabric_attributes": predicted_fabric,
        "pattern_attributes": predicted_pattern
    }
    return cnn_output
@app.route('/submit', methods=['POST'])
def submit():
    files = request.files.getlist('images[]')  # Get files sent with the form
    cnn_output = predict_image_attributes(files[0]) if files else {}
    sleeve_length = sleeve_length_map.get(cnn_output["shape_attributes"][0], "Unknown")
    lower_clothing_length = lower_clothing_length_map.get(cnn_output["shape_attributes"][1], "Unknown")
    neckline = neckline_map.get(cnn_output["shape_attributes"][9], "Unknown")
    # Extract predicted fabric attributes
    fabric_attributes = cnn_output["fabric_attributes"]
    fabric_attributes += ["NA"] * (3 - len(fabric_attributes))  # Ensure at least 3 values
    upper_fabric, lower_fabric, outer_fabric = fabric_attributes[:3]
    # Extract predicted pattern attributes
    pattern_attributes = cnn_output["pattern_attributes"]
    pattern_attributes += ["NA"] * (3 - len(pattern_attributes))  # Ensure at least 3 values
    upper_pattern, lower_pattern, outer_pattern = pattern_attributes[:3]
    print(cnn_output)
   #print color module output
    if upper_pattern=="pure color" :
        process_pure_color(files[0],"upper")
    if(lower_pattern)=="pure color" :
        process_pure_color(files[0],"lower")
```

5.3.3 Merging both extracted features



```
import torch
import numpy as np
from PIL import Image
import os
import cv2
import tensorflow as tf
@app.route('/submit', methods=['POST'])
def submit():
    description = request.form.get('description', '')  # Get description from the form
    files = request.files.getlist('images[]')
    ner_output = process_description(description)
    cnn_output = predict_image_attributes(files[0])
# Prepare the final input for the T5 model
    test_input = (
    f"Generate a product description for a product with "
    f"brand: {ner_output['brand']}, "
    f"type: {ner_output['type']}, "
    f"sleeve length: {sleeve_length}, "
    f"lower clothing length: {lower_clothing_length}, "
    f"neckline: {neckline}, "
    f"upper fabric: {upper_fabric}, "
    f"upper pattern: {upper_pattern}, "
    f"lower fabric: {lower_fabric}, "
    f"lower pattern: {lower_pattern}, "
    f"outer fabric: {outer_fabric}, "
    f"outer pattern: {outer_pattern}"
    )
    print(test_input)
```

5.3.4 Generating structured description



```python
from transformers import T5Tokenizer, T5ForConditionalGeneration
model_path = "./fine_tuned_flan_t5"
tokenizer = T5Tokenizer.from_pretrained(model_path)
caption_model = T5ForConditionalGeneration.from_pretrained(model_path)
def submit():
    description = request.form.get('description', '')
    files = request.files.getlist('images[]')
    test_input = (
    f"Generate a product description for a product with "
    f"brand: {ner_output['brand']}, "
    f"type: {ner_output['type']}, "
    f"sleeve length: {sleeve_length}, "
    f"lower clothing length: {lower_clothing_length}, "
    f"neckline: {neckline}, "
    f"upper fabric: {upper_fabric}, "
    f"upper pattern: {upper_pattern}, "
    f"lower fabric: {lower_fabric}, "
    f"lower pattern: {lower_pattern}, "
    f"outer fabric: {outer_fabric}, "
    f"outer pattern: {outer_pattern}"
    )
    print(test_input)
    input_ids = tokenizer.encode(test_input, return_tensors="pt", truncation=True,
max_length=128).to(device)
    outputs = caption_model.generate(input_ids, max_length=128, num_beams=4, early_stopping=True)
    generated_text = tokenizer.decode(outputs[0], skip_special_tokens=True)
    print("\nGenerated Product Description:")
    print(generated_text)
```

5.4 Chapter conclusion

In this chapter, our system is explained to generate structured product descriptions from clothing images using Python with Flask for backend development and HTML, CSS, and JavaScript for the user interface. Major functionality includes brand and type extraction from clothing through a Structured Text Extractor model, visual feature extraction by a Convolutional Neural Network model called ConvNext, and the fusion of visual and textual features to create a combined representationThe system also creates well-organized product descriptions through the aggregation of these derived features in a properly organized manner. The intuitive interface ensures that smooth interaction takes place, thus allowing for effective processing of clothing images. The entire system enhances the precision and comprehensiveness of clothing descriptions, hence rendering it highly effective for e-commerce use.

Chapter 6

Results and Discussions

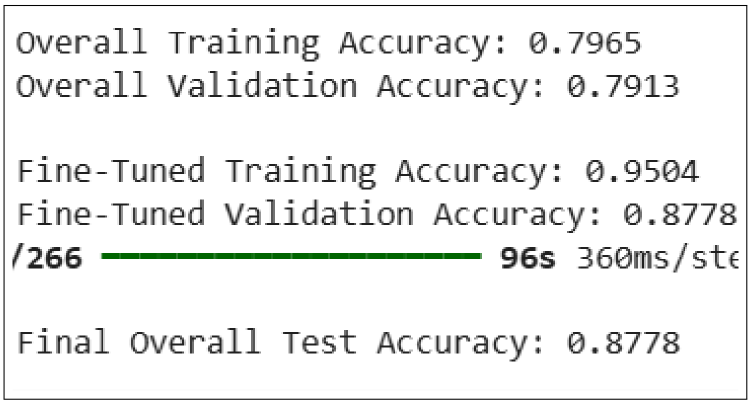
In this chapter, we delve into the results and discussions of the implemented product description generation using image and text analysis. The following sections provide an overview of the achieved results and in-depth discussions on the outcomes.

6.1 Overview

Here, the results of the research on developing and assessing a product description generation system through image and text analysis are given. The major objective of this research was to improve the accuracy and uniformity of fashion descriptions for online shopping websites by using image feature extraction(CNN). The system incorporates a Convolutional Neural Network (CNN) model for identifying visual features like material, sleeve length, and pattern from images of products, in addition to a Structured Text Extractor identifying brand and clothing type from seller descriptions. The discovered features are integrated to generate structured product descriptions using a Transformer-based FLAN-T5 model. At the development stage, deep preprocessing techniques were employed to pre-clean image and text data for the extraction of high-quality features. The analysis spans various significant facets, including model accuracy, quality of feature extraction, and quality of structured description generation. Moreover, heavy experimentation was performed to evaluate the system's robustness against noisy inputs, inconsistent text data, and apparel image variations. The findings showcase the capability of the system in enhancing product visibility, simplifying the listing process, and customer experience on online marketplaces.

6.2 Quantitative results

ConvNeXt model was trained with a dataset of apparel images having attributed labels. During the first stage of training, the model had a total training accuracy of 79.65% and a total validation accuracy of 79.13%. The model was fine-tuned with other training measures for enhanced performance, and this greatly improved accuracy. After fine-tuning, the model reached a training accuracy of 95.04% and a validation accuracy of 87.78%. Finally, during the testing phase, the model demonstrated a final test accuracy of 87.78%, confirming its effectiveness in extracting apparel features with high precision.



```
Overall Training Accuracy: 0.7965
Overall Validation Accuracy: 0.7913

Fine-Tuned Training Accuracy: 0.9504
Fine-Tuned Validation Accuracy: 0.8778
/266 ━━━━━━━━━━━━━━━━━━━━ 96s 360ms/ste

Final Overall Test Accuracy: 0.8778
```

Figure 6.1: ConvNeXt model output

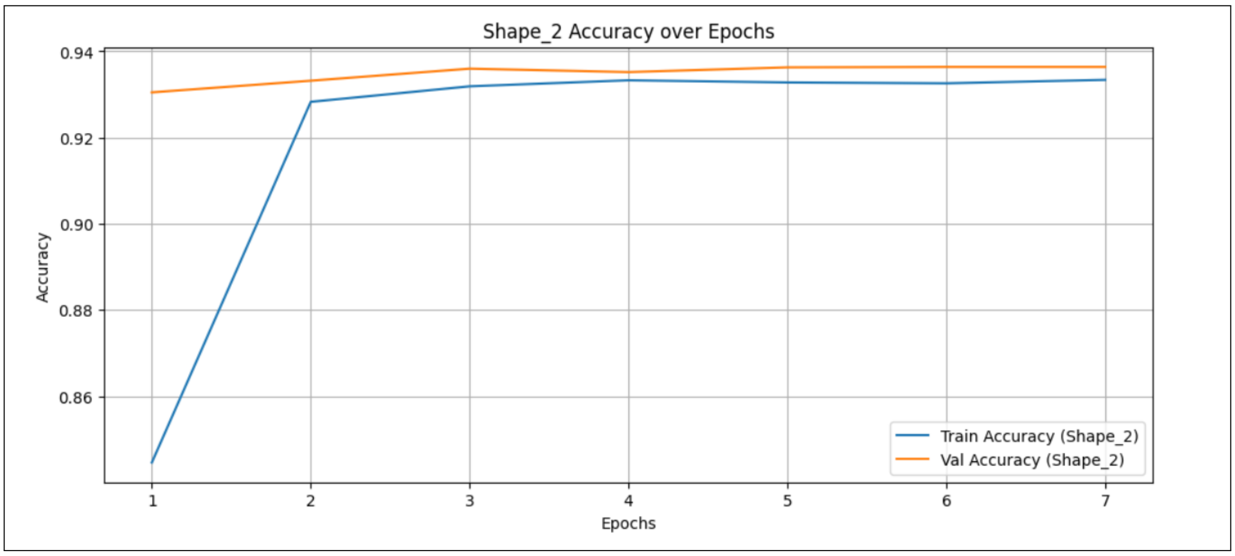


Figure 6.2: Shape attribute accuracy over epochs

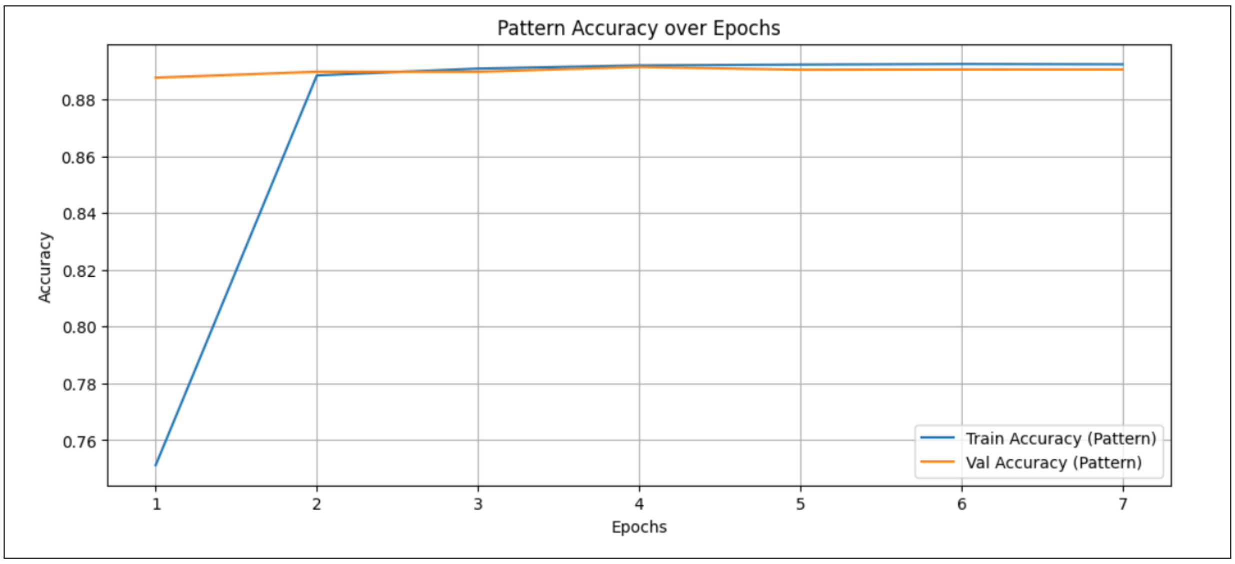


Figure 6.3: Pattern attribute accuracy over epochs

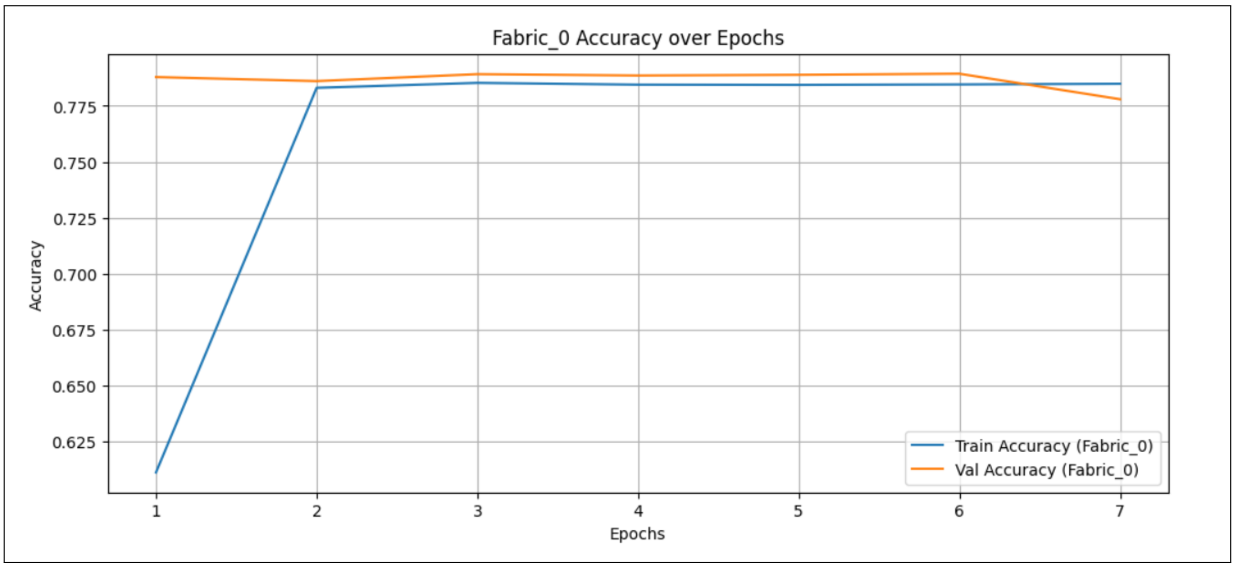


Figure 6.4: Fabric attribute accuracy over epochs

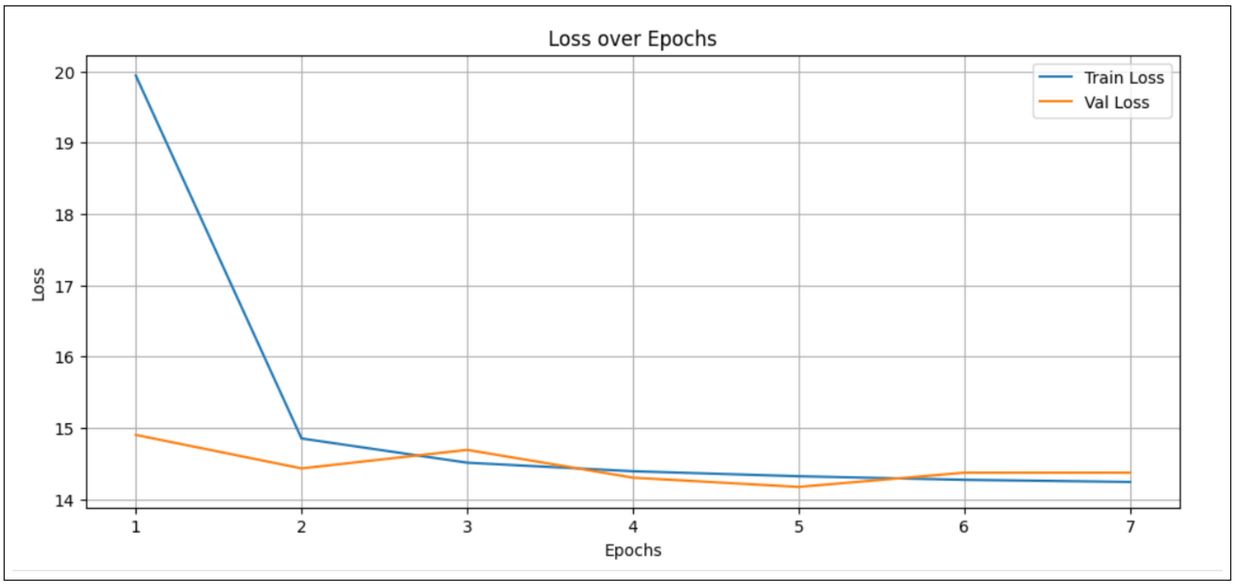


Figure 6.5: Loss over epochs

6.3 Comparing models

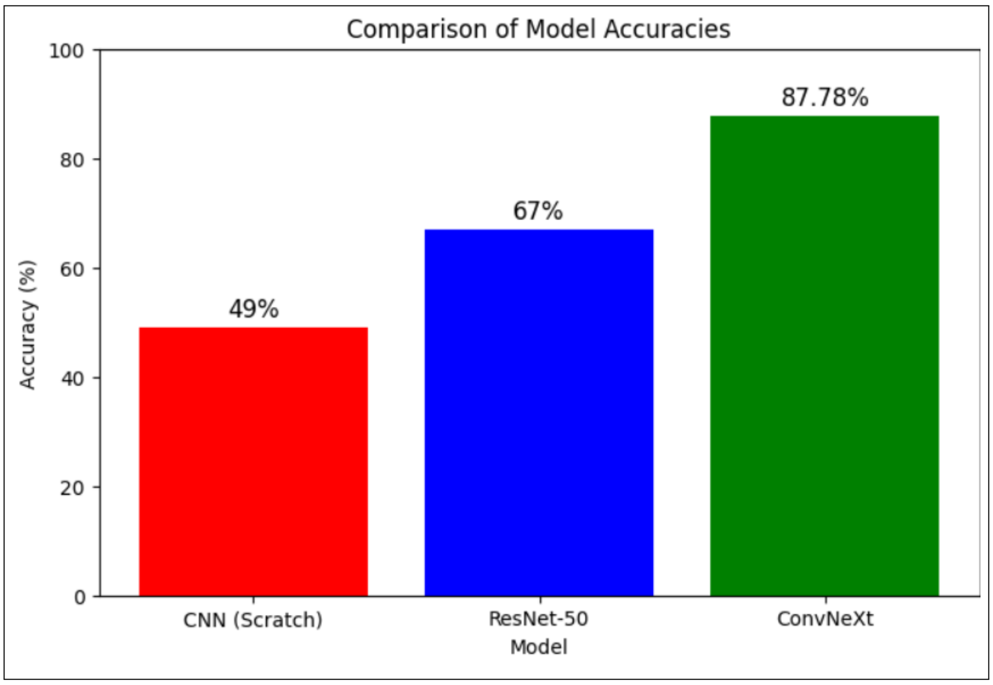
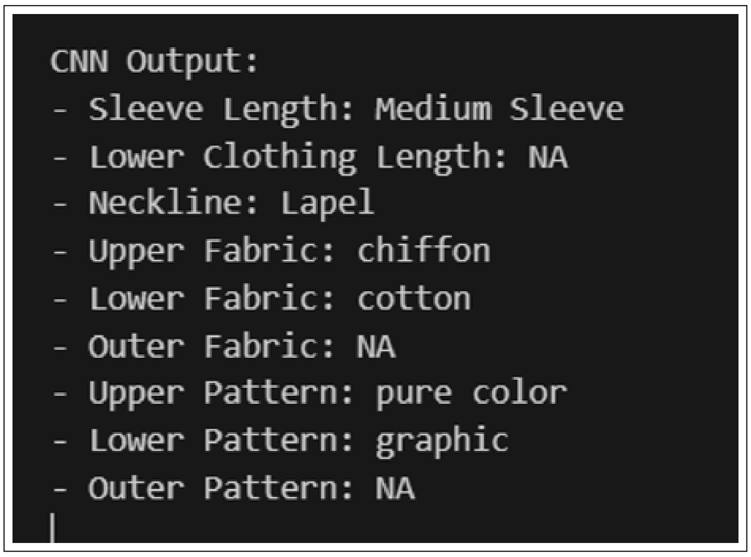


Figure 6.6: Model comparison

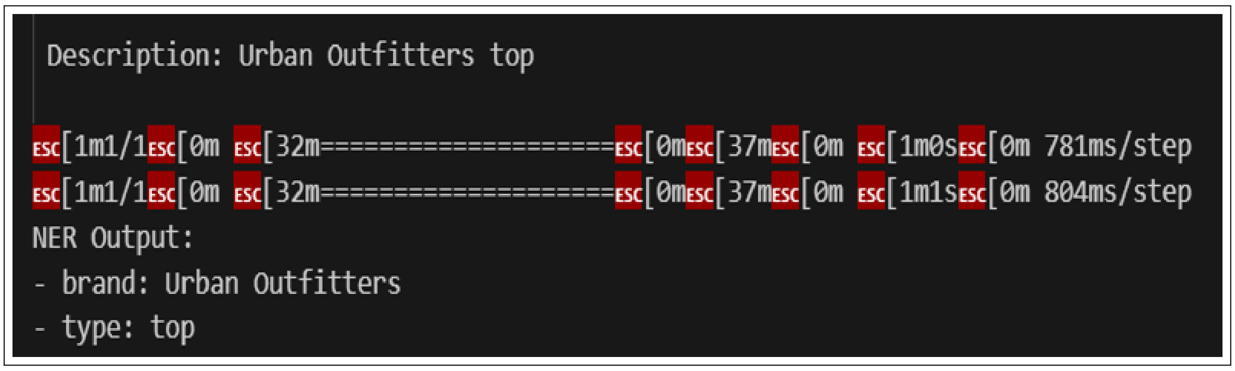
The comparison of the three models,CNN trained from scratch, ResNet-50, and ConvNeXt shows dramatic improvement in accuracy with more sophisticated architectures being used. The CNN from scratch model scores 49% accuracy, which means that it does pick up some patterns but fails to generalize well. ResNet-50, a well-known deep learning model for its residual connections, dramatically improves to 67% accuracy, gaining from deeper layers and improved feature extraction. Yet, the top accuracy of 87.78 % is offered by ConvNeXt, a recent architecture developed with considerations from both CNNs and Transformer-based models with better feature representation and robustness. This follows the trend showing the significance of employing state of the art architecture in enhancing classification performance, particularly in cases that involve intricate image features.

6.4 Test results



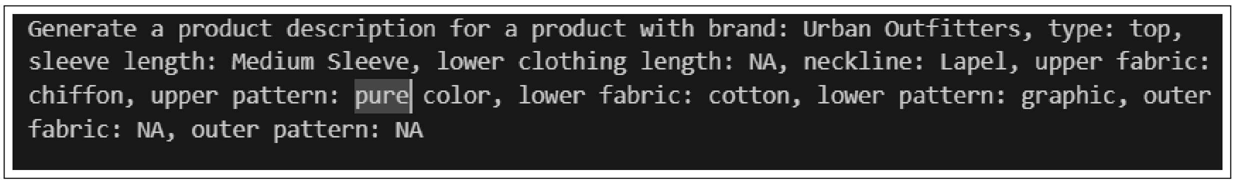
```
CNN Output:
- Sleeve Length: Medium Sleeve
- Lower Clothing Length: NA
- Neckline: Lapel
- Upper Fabric: chiffon
- Lower Fabric: cotton
- Outer Fabric: NA
- Upper Pattern: pure color
- Lower Pattern: graphic
- Outer Pattern: NA
```

Figure 6.7: Convnext model output



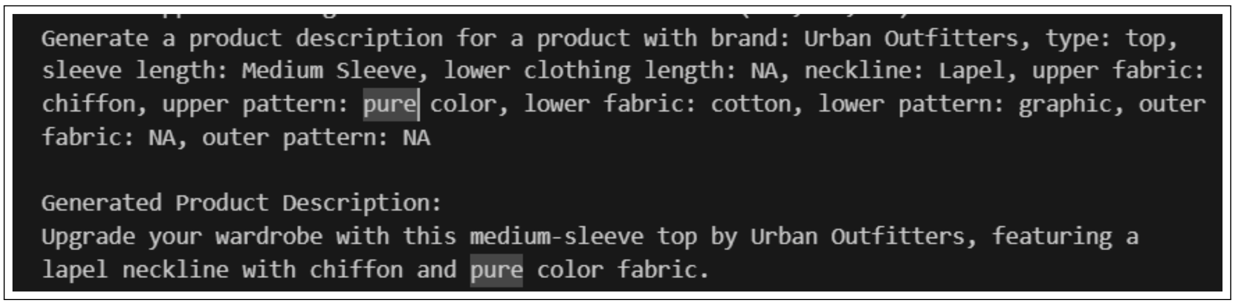
```
Description: Urban Outfitters top

ESC[1m1/1ESC[0m ESC[32m====================ESC[0mESC[37mESC[0m ESC[1m0sESC[0m 781ms/step
ESC[1m1/1ESC[0m ESC[32m====================ESC[0mESC[37mESC[0m ESC[1m1sESC[0m 804ms/step
NER Output:
- brand: Urban Outfitters
- type: top
```

Figure 6.8: Structured text extractor output



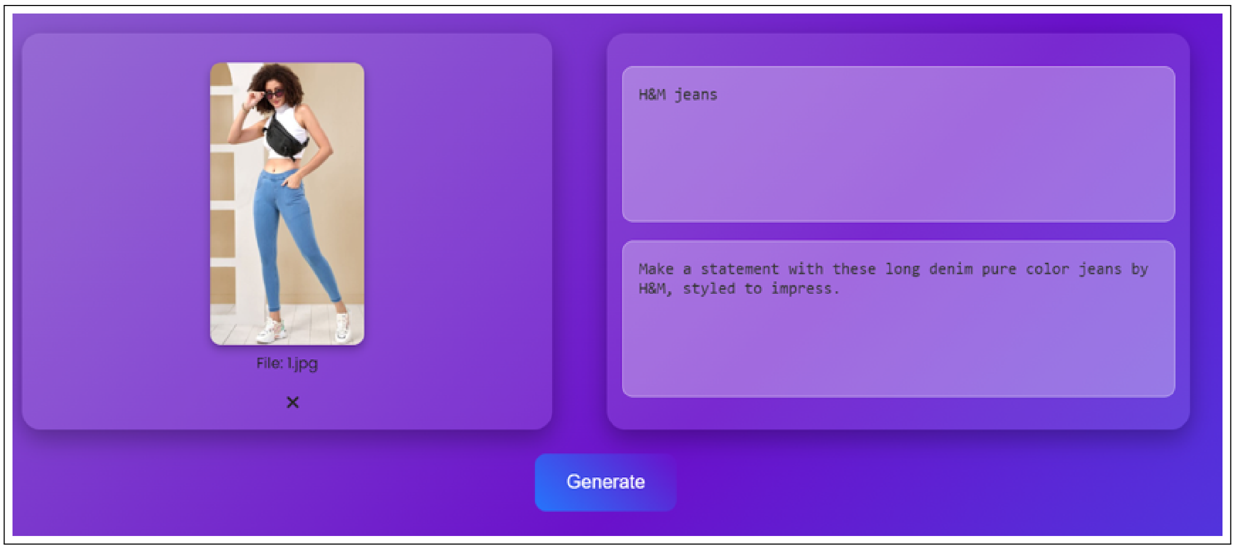
```
Generate a product description for a product with brand: Urban Outfitters, type: top,
sleeve length: Medium Sleeve, lower clothing length: NA, neckline: Lapel, upper fabric:
chiffon, upper pattern: pure color, lower fabric: cotton, lower pattern: graphic, outer
fabric: NA, outer pattern: NA
```

Figure 6.9: Combined feature vector



```
Generate a product description for a product with brand: Urban Outfitters, type: top,
sleeve length: Medium Sleeve, lower clothing length: NA, neckline: Lapel, upper fabric:
chiffon, upper pattern: pure color, lower fabric: cotton, lower pattern: graphic, outer
fabric: NA, outer pattern: NA

Generated Product Description:
Upgrade your wardrobe with this medium-sleeve top by Urban Outfitters, featuring a
lapel neckline with chiffon and pure color fabric.
```

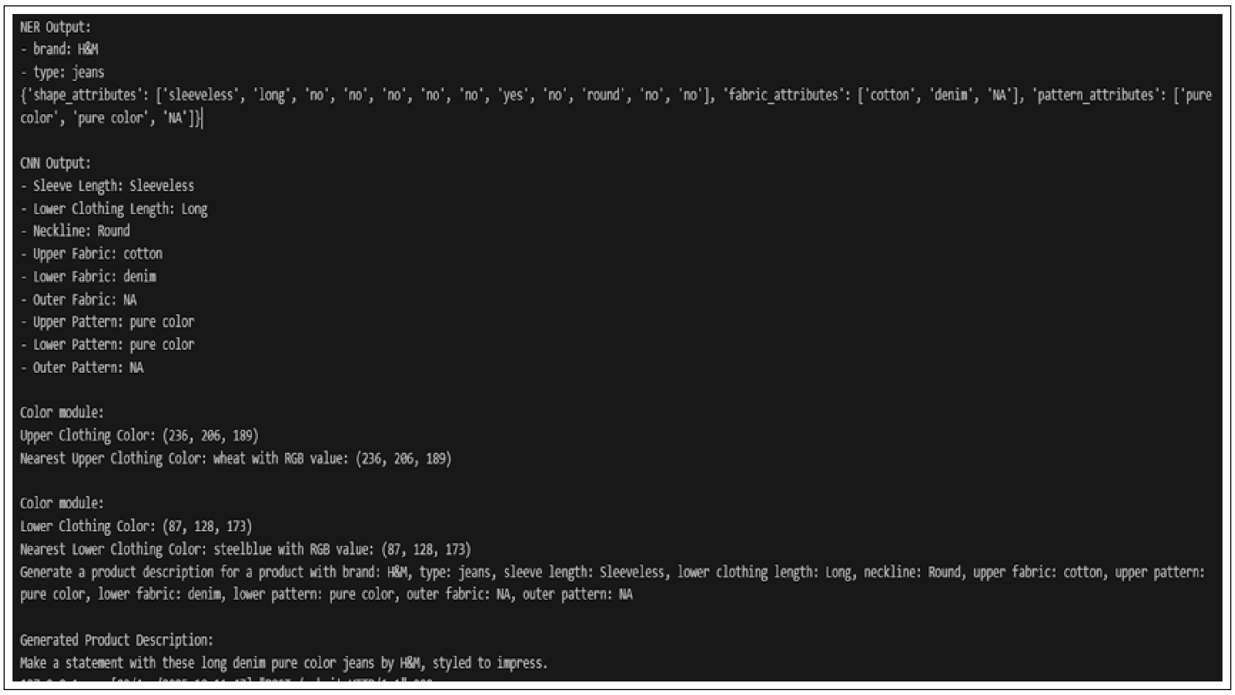
Figure 6.10: Generated structured description



H&M jeans

Make a statement with these long denim pure color jeans by H&M, styled to impress.

Figure 6.11: Front end



```
NER Output:
- brand: H&M
- type: jeans
{'shape_attributes': ['sleeveless', 'long', 'no', 'no', 'no', 'no', 'no', 'yes', 'no', 'round', 'no', 'no'], 'fabric_attributes': ['cotton', 'denim', 'NA'], 'pattern_attributes': ['pure
color', 'pure color', 'NA']}

CNN Output:
- Sleeve Length: Sleeveless
- Lower Clothing Length: Long
- Neckline: Round
- Upper Fabric: cotton
- Lower Fabric: denim
- Outer Fabric: NA
- Upper Pattern: pure color
- Lower Pattern: pure color
- Outer Pattern: NA

Color module:
Upper Clothing Color: (236, 206, 189)
Nearest Upper Clothing Color: wheat with RGB value: (236, 206, 189)

Color module:
Lower Clothing Color: (87, 128, 173)
Nearest Lower Clothing Color: steelblue with RGB value: (87, 128, 173)
Generate a product description for a product with brand: H&M, type: jeans, sleeve length: Sleeveless, lower clothing length: Long, neckline: Round, upper fabric: cotton, upper pattern:
pure color, lower fabric: denim, lower pattern: pure color, outer fabric: NA, outer pattern: NA

Generated Product Description:
Make a statement with these long denim pure color jeans by H&M, styled to impress.
[illegible]
```

Figure 6.12: Backend processed ouput

Chapter 7

Conclusion

The system accurately determines main visual attributes such as color, pattern, shape, and texture by utilizing image based feature extraction and offering a right vision of the product look. At the same time, the structured text extractor that is based on a positional mapping model handles the seller's inputs in the right manner and effectively extracts details such as material, brand, fit, and others from the product. It thereby erases even the non-viewable product details and adds them to the end output. An easy-to-use and natural user interface was created to assist sellers in inputting additional information, making each listing even more complete. The ultimate product description is generated automatically based on both visual and structured text data, yielding a rich, consistent, and professional description that has the feel and appeal of satisfying contemporary online shoppers' expectations.

This system is not designed for product classification but is solely aimed at creating descriptive, trustworthy, and well formatted product content. Its deployment has evident potential to save sellers' workload, enhance product visibility, and enhance the overall quality of listings on e-commerce sites.

Potential future improvements may involve offering support for multilingual product descriptions, dynamic templates by product category, and integration with current content management systems employed by vendors. Overall, this project illustrates the potential of harnessing computer vision and structured data processing to address an applicable, real-world issue in the field of e-commerce.

References

[1] A. Oluwasanmi, M. U. Aftab, E. Alabdulkreem, B. Kumeda, E. Y. Baagyere, and Z. Qin, "Captionnet: Automatic end-to-end siamese difference captioning model with attention," *IEEE Access*, vol. 7, pp. 106 773–106 783, 2019.

[2] G. Agarwal, K. Jindal, A. Chowdhury, V. K. Singh, and A. Pal, "Image and video captioning for apparels using deep learning," *IEEE Access*, vol. 12, pp. 113 138–113 150, 2024.

[3] Y. Zhang, K. He, and R. Song, "Image multi-feature fusion for clothing style classification," *IEEE Access*, vol. 11, pp. 107 843–107 854, 2023.

[4] A. Ueda, W. Yang, and K. Sugiura, "Switching text-based image encoders for captioning images with text," *IEEE Access*, vol. 11, pp. 55 706–55 715, 2023.

[5] Y. Zhang, K. He, and R. Song, "Image multi-feature fusion for clothing style classification," *IEEE Access*, vol. 11, pp. 107 843–107 854, 2023.

[6] W. Jiang, W. Zhou, and H. Hu, "Cross on cross attention: Deep fusion transformer for image captioning," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 34, no. 5, pp. 3563–3575, 2023.

[7] Y. Liu, Y. Yang, R. Xiang, and J. Ma, "Complementary shifted transformer for image captioning," *Neural Processing Letters*, vol. 55, pp. 8339–8363, 2023.

[8] L. Guo, J. Liu, X. Zhu, P. Yao, S. Lu, and H. Lu, "Image captioning using transformer-based double attention network," *Engineering Applications of Artificial Intelligence*, vol. 125, p. 106545, 2023.

[9] Vaishali and S. Hegde, "Image captioning on apparel using neural network," in *International Conference on Intelligent Computing, Communication and Information Security*, 2023, pp. 369–377.

[10] A. Afzal, S. U. Khan, and A. Ali, “A transformer-based urdu image caption generation,” *Journal of Ambient Intelligence and Humanized Computing*, 2024.

[11] S. He, W. Liao, H. R. Tavakoli, M. Yang, B. Rosenhahn, and N. Pugeault, “Image captioning through image transformer,” *arXiv preprint arXiv:2004.14231*, 2020.

Appendix A: Presentation

Product Description Generation Using Image and Text Analysis

PROJECT PRESENTATION

Team Members :

Nayana V Reji (u2103151)
Nikita Alex (u2103156)
Niranjan G Das (u2103157)
Praveen Raj C S (u2103166)

Guide :

Ms. Janani K
Assistant Professor
Dept of CSE
RSET

CONTENTS

PROBLEM DEFINITION

In the competitive e-commerce space, accurate product descriptions are vital for Search Engine Optimization and customer satisfaction. Many sellers struggle to create detailed, consistent descriptions, leading to poor visibility, confusion, and negative customer experiences. A system is needed to help sellers generate precise descriptions, improving searchability and reducing errors.

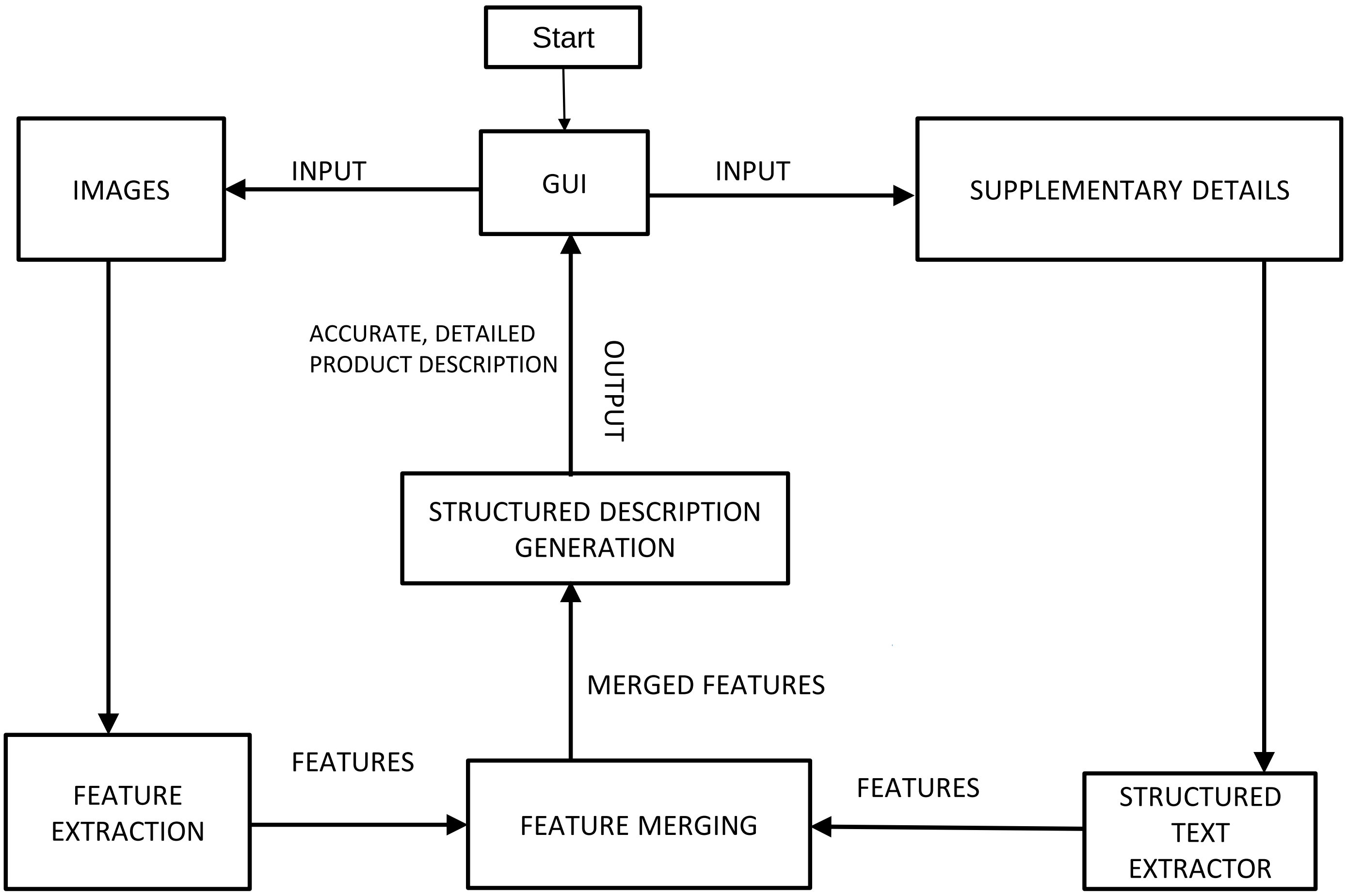
PURPOSE AND NEED

- Assist sellers in generating optimized and detailed product descriptions.
- Analyze product images to extract key features.
- Combine image data with small seller-provided descriptions.
- Ensure accurate and consistent descriptions to improve searchability and customer satisfaction.

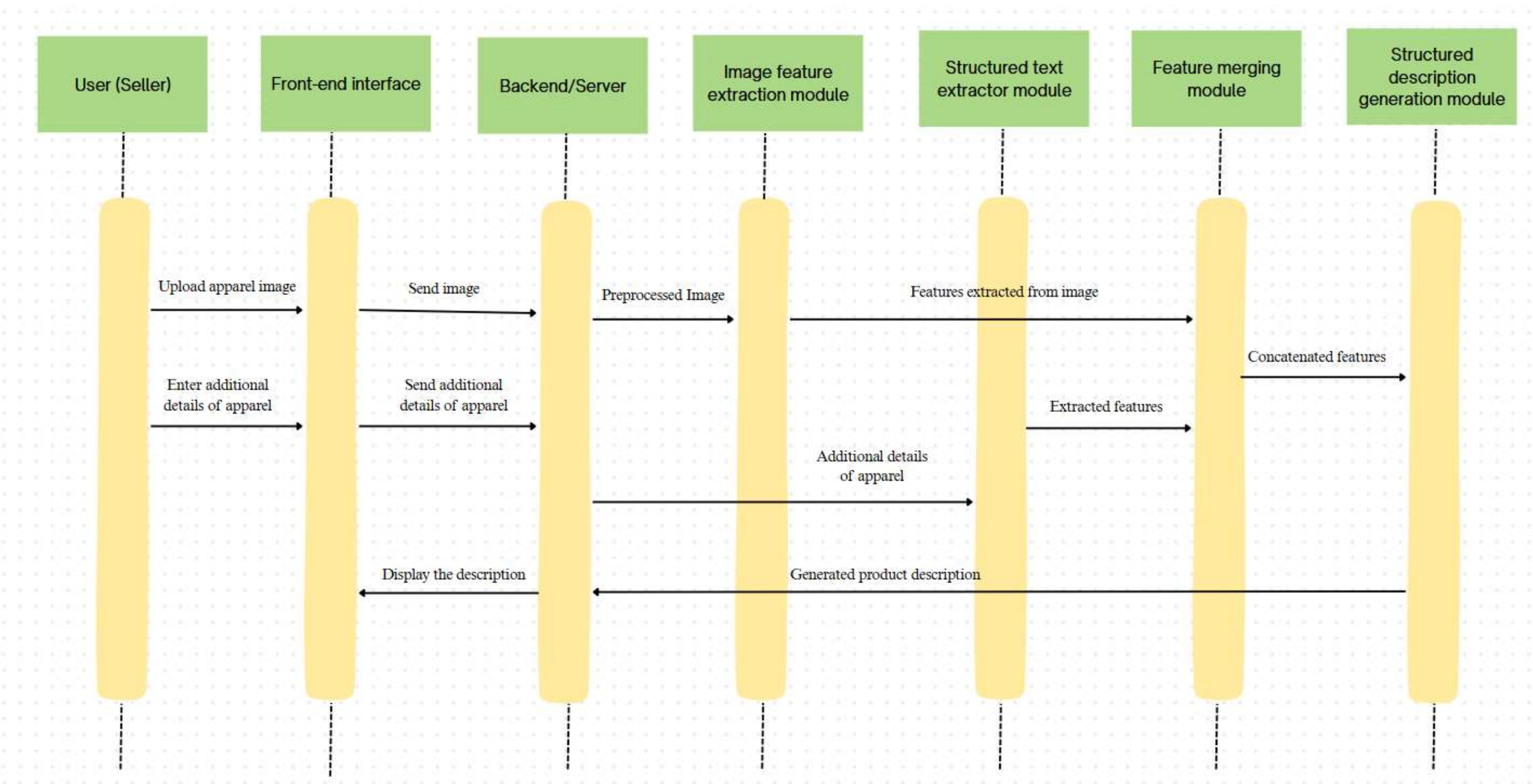
OBJECTIVES

To design and implement a user-friendly interface where sellers can easily input product images and additional product details that may not be directly inferred from the product images.This ensures that all critical and relevant information is captured, contributing to a more comprehensive, accurate, and detailed product description. This interface will also help sellers quickly review and update listings, making the system more efficient and accessible for a wide range of users.

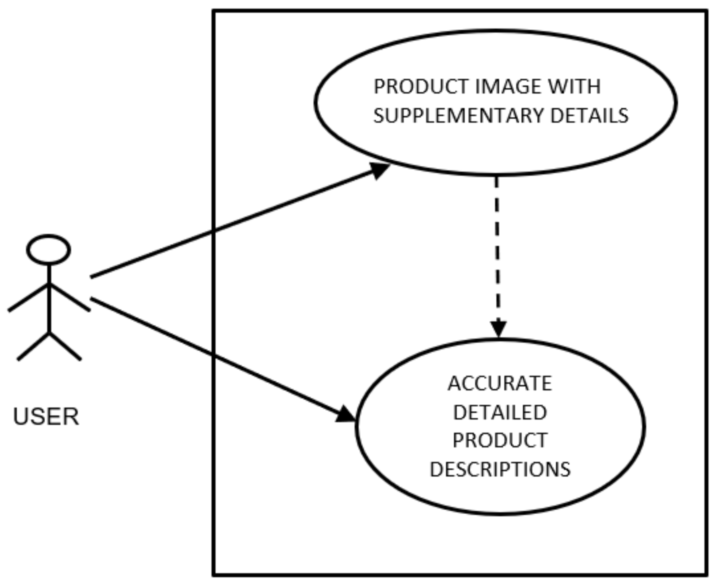
ARCHITECTURAL DIAGRAM



SEQUENCE DIAGRAM


User (Seller)
Front-end interface
Backend/Server
Image feature extraction module
Structured text extractor module
Feature merging module
Structured description generation module
Upload apparel image
Send image
Preprocessed Image
Features extracted from image
Concatenated features
Enter additional details of apparel
Send additional details of apparel
Extracted features
Additional details of apparel
Display the description
Generated product description

USE CASE DIAGRAM


PRODUCT IMAGE WITH SUPPLEMENTARY DETAILS
USER
ACCURATE DETAILED PRODUCT DESCRIPTIONS

LITERATURE SURVEY

Title	Dataset	Methodology	Result	Advantages	Disadvantages
A. Oluwasanmi (2019): CaptionNet: Automatic End-to-End Siamese Difference Captioning Model With Attention	Custom dataset of image pairs (before-after)	Siamese network with attention mechanism to generate captions based on differences in images	Improved capability in generating context-aware captions for image differences	• Attention mechanism improves focus on relevant image regions • End-to-end training pipeline	• Limited to visual difference tasks • May not generalize well to broader captioning scenarios

Title	Dataset	Methodology	Result	Advantages	Disadvantages
G. Agarwal (2024): Image and Video Captioning for Apparels Using Deep Learning	DeepFashion and custom apparel video datasets	CNN for feature extraction + LSTM for sequential caption generation	High-quality captions for apparel images and videos; good object recognition	• Strong object detection and apparel-specific tagging • Effective for both images and videos	• High computational complexity • Requires large labeled datasets

Title	Dataset	Methodology	Result	Advantages	Disadvantages
Y. Zhang (2023): Image Multi-Feature Fusion for Clothing Style Classification	Fashion-MNIST and street-style datasets	Feature fusion of color, texture, and shape + deep classification model (CNN)	Enhanced classification accuracy in predicting clothing style	• Improved accuracy due to multi-feature fusion • Better performance than single-feature models	• Risk of overfitting on small or biased datasets • Needs proper feature normalization

Title	Dataset	Methodology	Result	Advantages	Disadvantages
A. Ueda (2023): Switching Text-Based Image Encoders for Captioning Images With Text	Conceptual Captions & COCO datasets	Uses dynamic switching between multiple text-based image encoders (e.g., CLIP, ViLT) to adaptively process multimodal input (image + text) for caption generation	Achieves better flexibility in captioning various image types, especially those containing embedded text	• Multimodal understanding across visual and textual domains • Flexibility to adapt to different types of input	• Susceptible to biases in pretrained models • Limited contextual understanding of complex scenes

METHODOLOGY

IMAGE FEATURE EXTRACTION MODULE

Objective:

- Extract characteristic features (material, length, sleeve length, etc.) from clothing images.

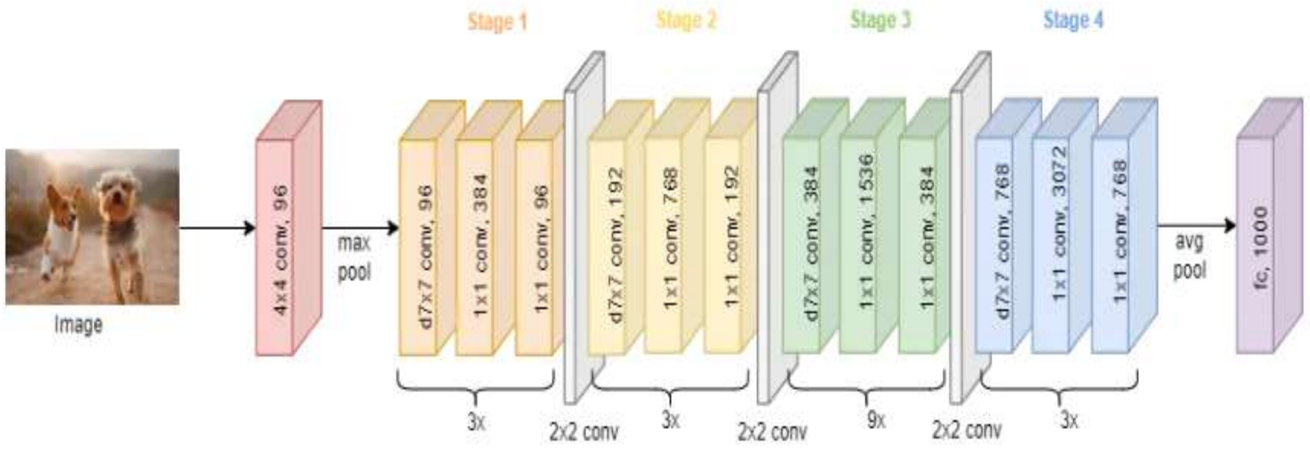
Implementation Details:

- Dataset : DeepFashion-MultiModal is a large-scale high-quality human dataset with rich multi-modal annotations. It has 44,096 images and there corresponding shape , pattern and fabric files with attribute annotated

- Model Architecture : Pretrained ConvNeXt model fine-tuned for attribute classification

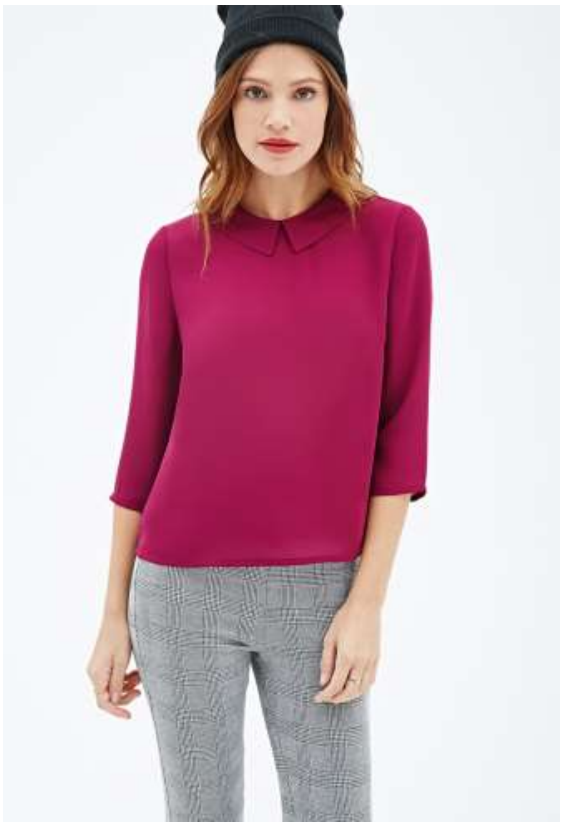
- Input to the model : Apparel image to extract features from
- Output from the model : Extracted shape , fabric and pattern attributes

Convnext Architecture

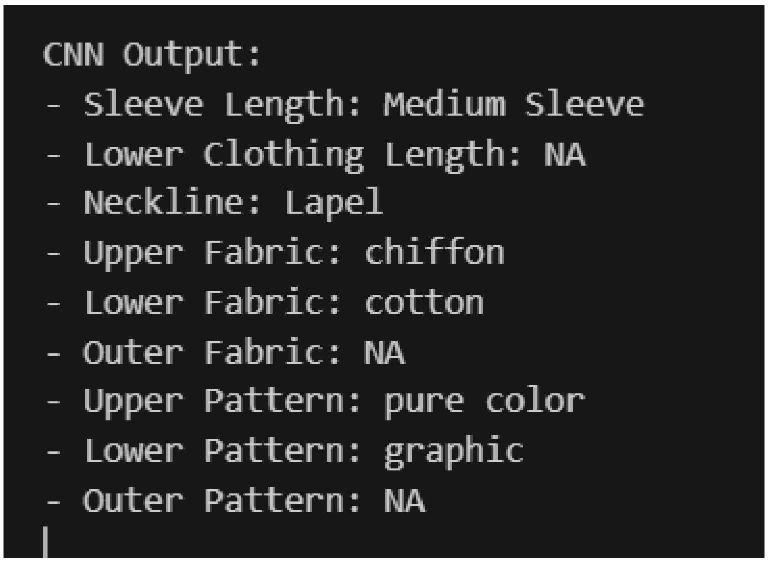


ConvNext — General Structure

TEST IMAGE



OUTPUT



```
CNN Output:
- Sleeve Length: Medium Sleeve
- Lower Clothing Length: NA
- Neckline: Lapel
- Upper Fabric: chiffon
- Lower Fabric: cotton
- Outer Fabric: NA
- Upper Pattern: pure color
- Lower Pattern: graphic
- Outer Pattern: NA
```

STRUCTURED TEXT EXTRACTOR

Objective:

- Extract structured product details (brand, clothing type) from textual descriptions using a position-based entity extraction approach.

Implementation Details:

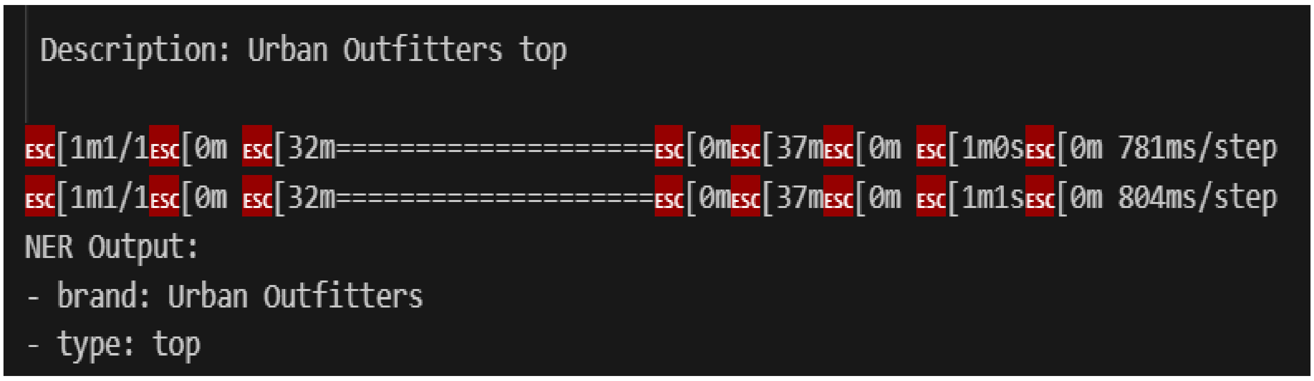
- Dataset: Labeled product descriptions annotated with named entities for clothing type and brand.

- Model Architecture: A custom rule-based system trained to identify brand and clothing type based on their positions and patterns in the description.

- Input to the model: Raw product descriptions from seller.

- Output from the model: Extracted brand and clothing type based on predefined positional mapping rules.

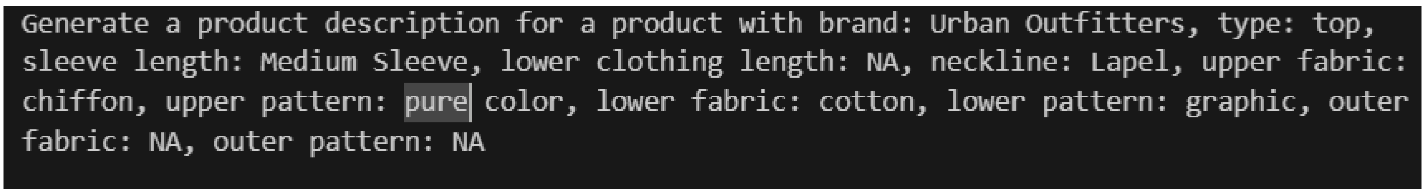
OUTPUT



```
 Description: Urban Outfitters top

ESC[1m1/1ESC[0m ESC[32m━━━━━━━━━━━━━━━━━━━━ESC[0mESC[37mESC[0m ESC[1m0sESC[0m 781ms/step
ESC[1m1/1ESC[0m ESC[32m━━━━━━━━━━━━━━━━━━━━ESC[0mESC[37mESC[0m ESC[1m1sESC[0m 804ms/step
NER Output:
- brand: Urban Outfitters
- type: top
```

FEATURE MERGING MODULE

- The structured entity extractor extracts information such as brand and type from the product description provided by the seller.
- The CNN model predicts clothing attributes like sleeve length, fabric type, pattern, neckline, and clothing length from the image.
- The extracted attributes from both models are formatted into a structured text prompt that contains all relevant details for generating an accurate product description.



```
Generate a product description for a product with brand: Urban Outfitters, type: top,
sleeve length: Medium Sleeve, lower clothing length: NA, neckline: Lapel, upper fabric:
chiffon, upper pattern: pure color, lower fabric: cotton, lower pattern: graphic, outer
fabric: NA, outer pattern: NA
```

STRUCTURED DESCRIPTION GENERATION MODULE

Objective: Generating structured product descriptions from the combined features of the apparel.

Implementation Details:

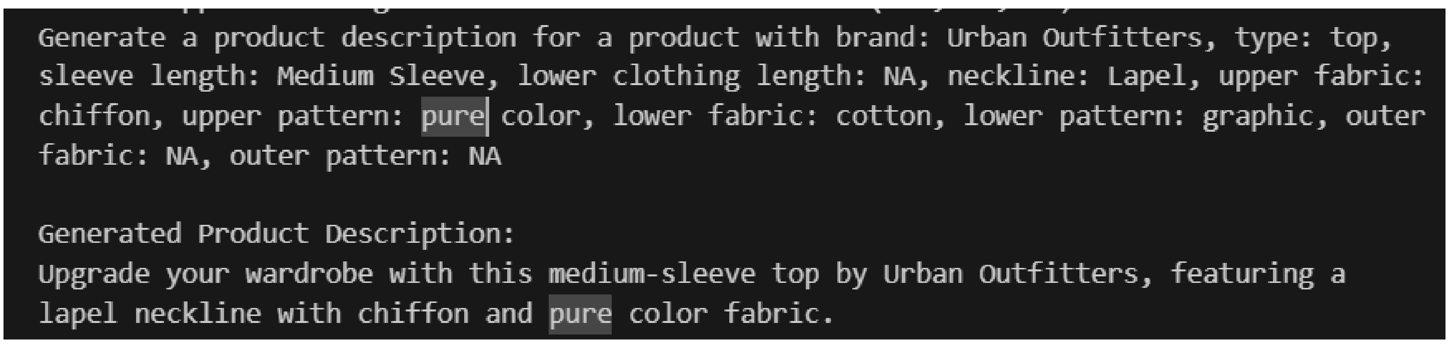
Dataset: contains features like brand,type,sleeve length,neckline etc along with the corresponding description.It has 3400 rows of features and their corresponding descriptions.

Model Architecture: Transformer based FLAN-T5(Fine-Tuned Text-To-Text Transfer Transformer Model) finetuned for generating structured product description using the labelled dataset.

Input: Combined features from the feature merging module

Output: A structured description of the apparel

OUTPUT



```
Generate a product description for a product with brand: Urban Outfitters, type: top,
sleeve length: Medium Sleeve, lower clothing length: NA, neckline: Lapel, upper fabric:
chiffon, upper pattern: pure color, lower fabric: cotton, lower pattern: graphic, outer
fabric: NA, outer pattern: NA

Generated Product Description:
Upgrade your wardrobe with this medium-sleeve top by Urban Outfitters, featuring a
lapel neckline with chiffon and pure color fabric.
```

EXPECTED OUTCOME

- A system that generates coherent, structured descriptions for products based on both image and textual inputs.
- An optimized search experience for users on e-commerce platforms.
- Automatic error-flagging to alert sellers to incorrect product descriptions.

ASSUMPTIONS

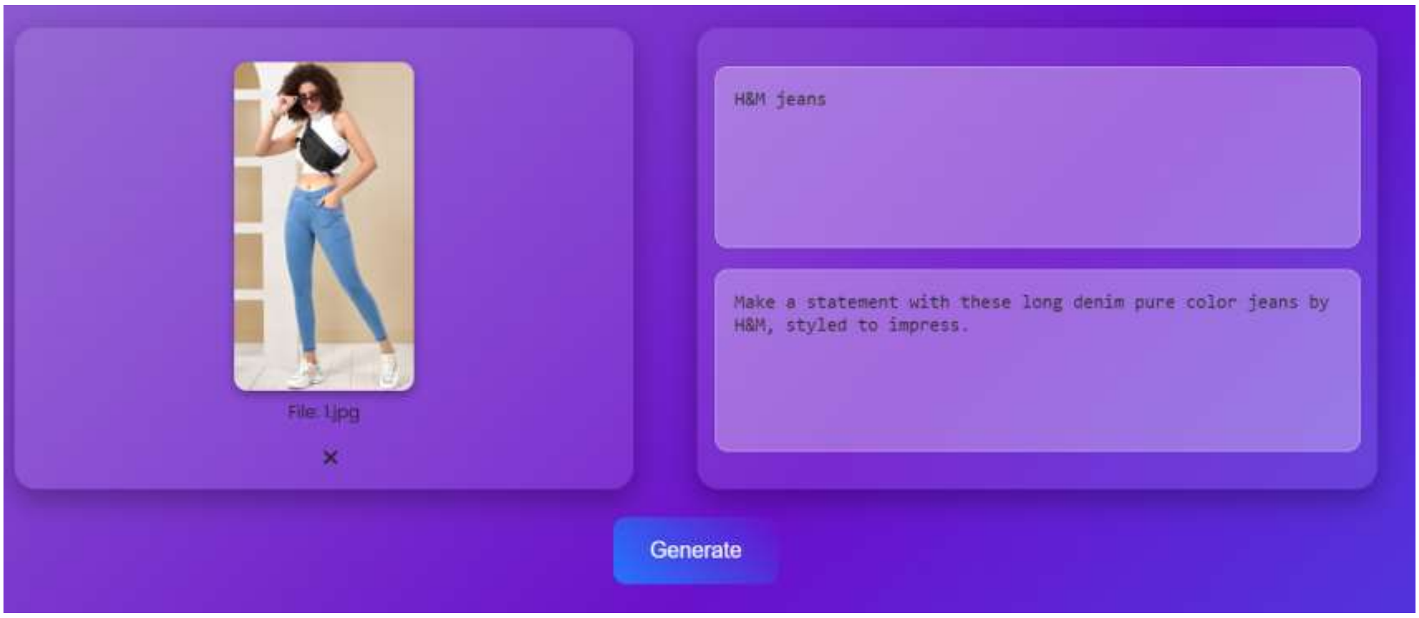
- The system assumes that the provided product images are clear and of good quality.
- it is assumed that the short product descriptions provided by the seller are accurate to the best of their knowledge, with the brand name mentioned first, followed by the clothing type, to enable effective structured text extraction using positional mapping.

RISKS AND CHALLENGES

- Difficulty in acquiring high-quality, diverse product images for training.
- The complexity of merging multimodal features from image and text data.
- Balancing accuracy and performance, especially with larger datasets.

OUTPUT

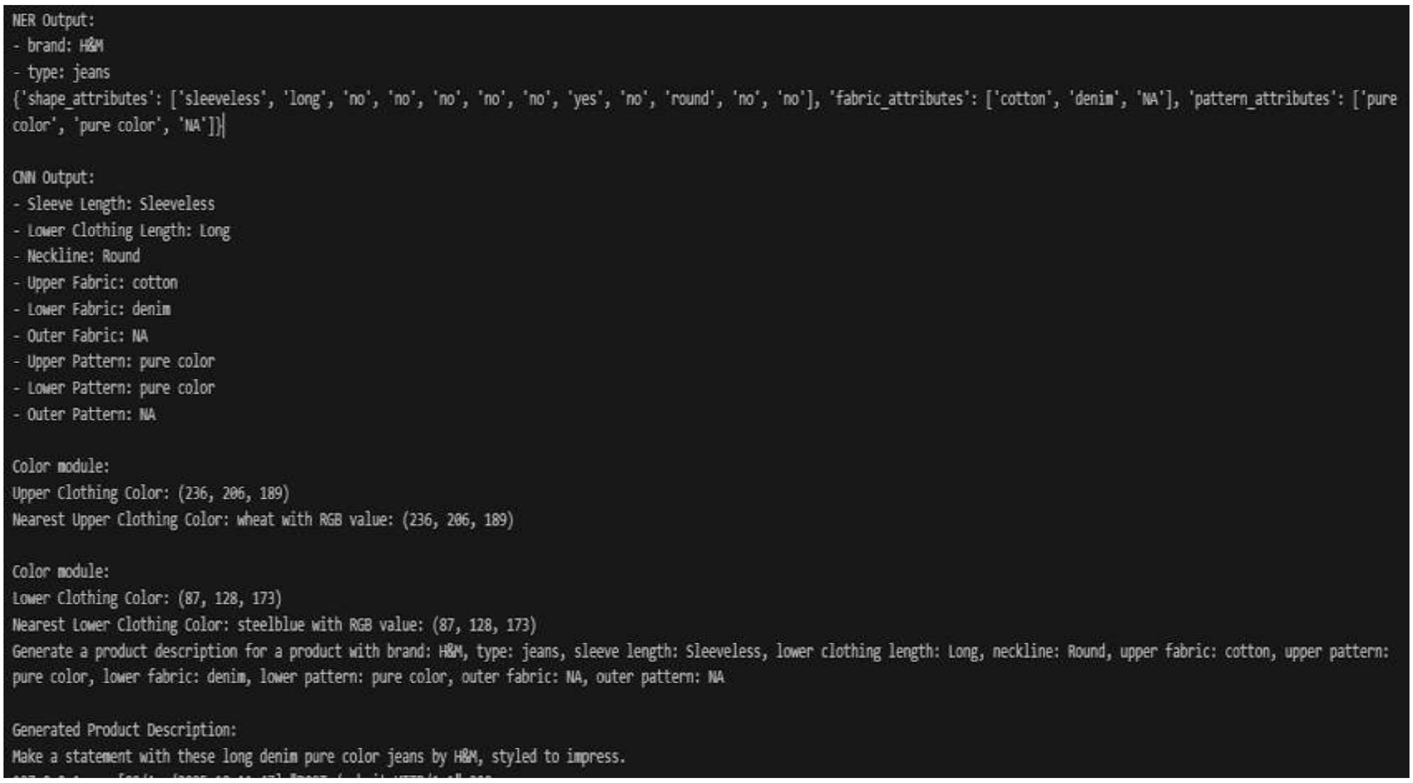
FRONT END



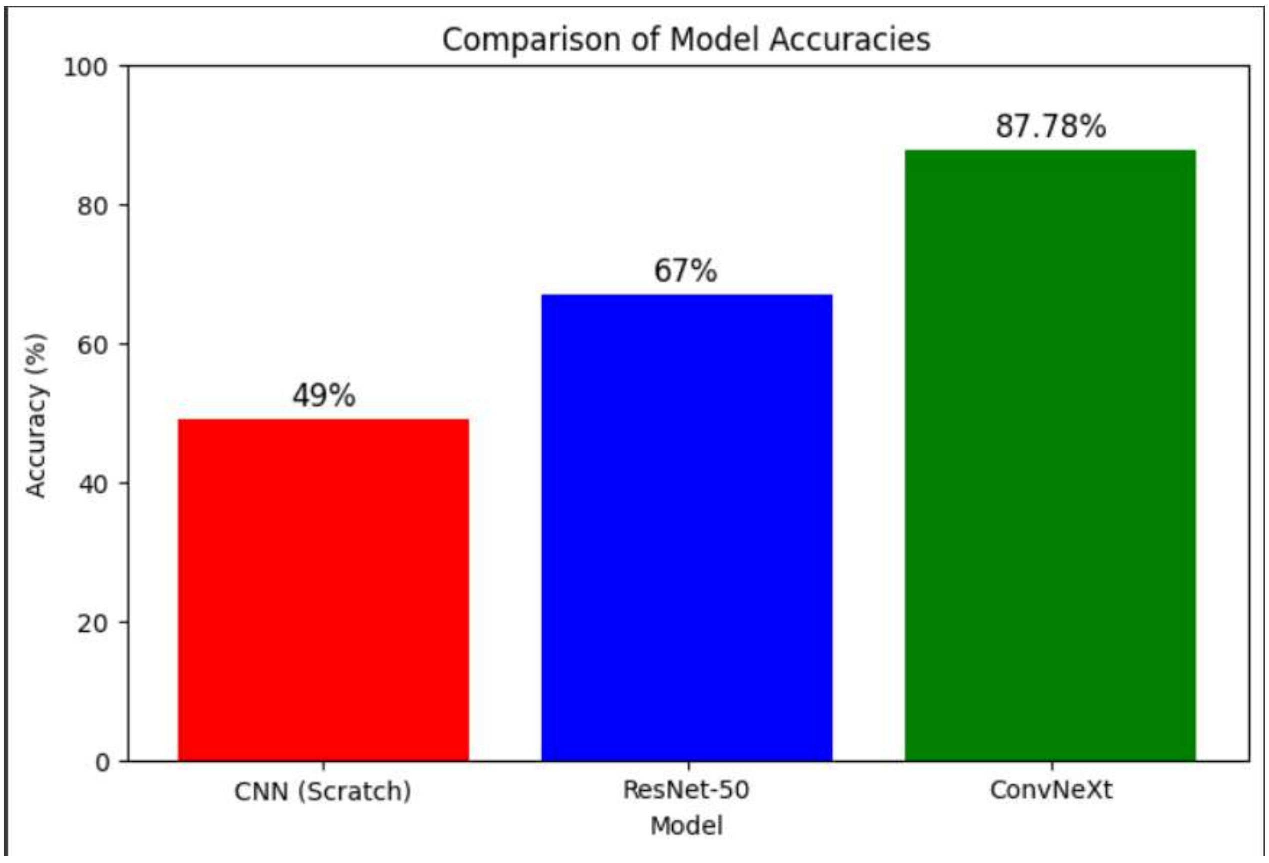
H&M jeans

Make a statement with these long denim pure color jeans by H&M, styled to impress.

BACK END



```
NER Output:
- brand: H&M
- type: jeans
{'shape_attributes': ['sleeveless', 'long', 'no', 'no', 'no', 'no', 'no', 'yes', 'no', 'round', 'no', 'no'], 'fabric_attributes': ['cotton', 'denim', 'NA'], 'pattern_attributes': ['pure
color', 'pure color', 'NA']}

CNN Output:
- Sleeve Length: Sleeveless
- Lower Clothing Length: Long
- Neckline: Round
- Upper Fabric: cotton
- Lower Fabric: denim
- Outer Fabric: NA
- Upper Pattern: pure color
- Lower Pattern: pure color
- Outer Pattern: NA

Color module:
Upper Clothing Color: (236, 206, 189)
Nearest Upper Clothing Color: wheat with RGB value: (236, 206, 189)

Color module:
Lower Clothing Color: (87, 128, 173)
Nearest Lower Clothing Color: steelblue with RGB value: (87, 128, 173)
Generate a product description for a product with brand: H&M, type: jeans, sleeve length: Sleeveless, lower clothing length: Long, neckline: Round, upper fabric: cotton, upper pattern:
pure color, lower fabric: denim, lower pattern: pure color, outer fabric: NA, outer pattern: NA

Generated Product Description:
Make a statement with these long denim pure color jeans by H&M, styled to impress.
```



Comparison of Model Accuracies
Accuracy (%)
100
80
60
40
20
0
49%
67%
87.78%
CNN (Scratch)
ResNet-50
ConvNeXt
Model

REQUIREMENTS

HARDWARE

- CPU: Intel i3 / Ryzen 3 or higher
- RAM: 8 GB
- Storage: 256 GB SSD
- GPU:Google colab,Kaggle

SOFTWARE

- Development Environment:VS-Code, Google colab,Kaggle
- Libraries: TensorFlow, Torch,Spacy

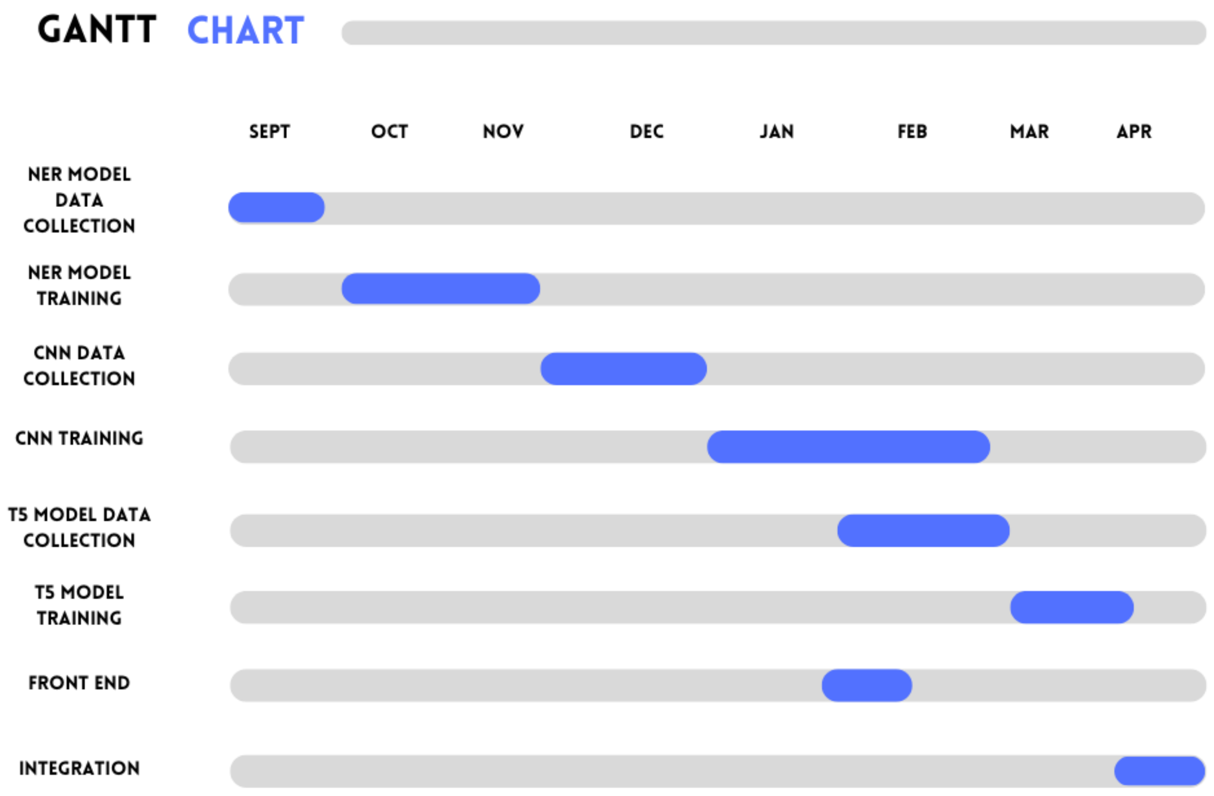
WORK BREAKDOWN & RESPONSIBILITIES

01 NAYANA V REJI
Image feature Extraction

02 PRAVEEN RAJ C S
Structured entity extractor

03 NIRANJAN G DAS
Feature Merging and Front-end

04 NIKITA ALEX
Structured Description Generation


GANTT CHART
SEPT
OCT
NOV
DEC
JAN
FEB
MAR
APR
NER MODEL DATA COLLECTION
NER MODEL TRAINING
CNN DATA COLLECTION
CNN TRAINING
T5 MODEL DATA COLLECTION
T5 MODEL TRAINING
FRONT END
INTEGRATION

FUTURE SCOPE

- Expand the system’s capabilities to support multiple languages, enhancing accessibility for global users and expanding market reach.
- Incorporate user feedback mechanisms to continually improve the quality of generated descriptions over time through supervised updates.
- Add a pre-processing step to evaluate image clarity and recommend improvements, ensuring more reliable feature extraction.

CONCLUSION

- The proposed system aims to significantly enhance product listings in e-commerce by automating the generation of accurate, detailed, and SEO-optimized descriptions.
- By integrating image feature extraction, natural language processing, and a cross-checking module, the system ensures consistency between product visuals and textual descriptions. The addition of a user-friendly interface for sellers to input supplementary details will further improve description quality.
- Ultimately, this solution will help sellers boost product visibility, streamline the listing process, and improve customer satisfaction, resulting in better sales performance and fewer product-related issues.

REFERENCES

- C. Rui, "Research on Classification of Cross-Border E-Commerce Products Based on Image Recognition and Deep Learning," in IEE Access, vol. 9, pp. 108083-108090, 2021
- S. Ren, K. He, R. Girshick, and J. Sun, ‘‘Faster R-CNN: Towards real-time object detection with region proposal networks,’’ in Proc. NIPS, vol. 28, 2015, pp. 1–9.
- O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, ‘‘ImageNet large scale visual recognition challenge,’’ Int. J. Comput. Vis., vol. 115, no. 3, pp. 1–42, 2015.
- G. Koch, R. Zemel, and R. Salakhutdinov, ‘‘Siamese neural networks for one-shot image recognition,’’ in Proc. ICML Deep Learn. Workshop, 2015, pp. 1–30.

Appendix B: Vision, Mission, Programme Outcomes and Course Outcomes

Vision, Mission, Programme Outcomes and Course Outcomes

Institute Vision
To evolve into a premier technological institution, moulding eminent professionals with creative minds, innovative ideas and sound practical skill, and to shape a future where technology works for the enrichment of mankind.

Institute Mission
To impart state-of-the-art knowledge to individuals in various technological disciplines and to inculcate in them a high degree of social consciousness and human values, thereby enabling them to face the challenges of life with courage and conviction.

Department Vision
To become a centre of excellence in Computer Science and Engineering, moulding professionals catering to the research and professional needs of national and international organizations.

Department Mission
To inspire and nurture students, with up-to-date knowledge in Computer Science and Engineering, ethics, team spirit, leadership abilities, innovation and creativity to come out with solutions meeting societal needs.

Programme Outcomes (PO)
Engineering Graduates will be able to:

1. Engineering Knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

2. Problem analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

3. Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

4. Conduct investigations of complex problems: Use research-based knowledge including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

5. Modern Tool Usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

6. The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal, and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

7. Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

8. Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. Individual and Team work: Function effectively as an individual, and as a member or leader in teams, and in multidisciplinary settings.

10. Communication: Communicate effectively with the engineering community and with society at large. Be able to comprehend and write effective reports documentation. Make effective presentations, and give and receive clear instructions.

11. Project management and finance: Demonstrate knowledge and understanding of engineering and management principles and apply these to one's own work, as a member and leader in a team. Manage projects in multidisciplinary environments.

12. Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and lifelong learning in the broadest context of technological change.

Programme Specific Outcomes (PSO)
A graduate of the Computer Science and Engineering Program will demonstrate:

PSO1: Computer Science Specific Skills
The ability to identify, analyze and design solutions for complex engineering problems in multidisciplinary areas by understanding the core principles and concepts of computer science and thereby engage in national grand challenges.

PSO2: Programming and Software Development Skills
The ability to acquire programming efficiency by designing algorithms and applying standard practices in software project development to deliver quality software products meeting the demands of the industry.

PSO3: Professional Skills
The ability to apply the fundamentals of computer science in competitive research and to develop innovative products to meet the societal needs thereby evolving as an eminent researcher and entrepreneur.

Course Outcomes (CO)

After the completion of the course the student will be able to:

Course Outcome 1: Model and solve real world problems by applying knowledge across domains (Cognitive knowledge level: Apply).

Course Outcome 2: Develop products, processes or technologies for sustainable and socially relevant applications (Cognitive knowledge level: Apply).

Course Outcome 3: Function effectively as an individual and as a leader in diverse teams and to comprehend and execute designated tasks (Cognitive knowledge level: Apply).

Course Outcome 4: Plan and execute tasks utilizing available resources within timelines, following ethical and professional norms (Cognitive knowledge level: Apply).

Course Outcome 5: Identify technology/research gaps and propose innovative/creative solutions (Cognitive knowledge level: Analyze).

Course Outcome 6: Organize and communicate technical and scientific findings effectively in written and oral forms (Cognitive knowledge level: Apply).

Appendix C: CO-PO-PSO Mapping

COURSE OUTCOMES:

After completion of the course, the student will be able to:

SL.NO	DESCRIPTION	Bloom's Taxonomy Level
CO1	Model and solve real-world problems by applying knowledge across domains (Cognitive knowledge level:Apply).	Level3: Apply
CO2	Develop products, processes, or technologies for sustainable and socially relevant applications. (Cognitive knowledge level:Apply).	Level 3: Apply
CO3	Function effectively as an individual and as a leader in diverse teams and comprehend and execute designated tasks. (Cognitive knowledge level:Apply).	Level 3: Apply
CO4	Plan and execute tasks utilizing available resources within timelines, following ethical and professional norms (Cognitive knowledge level:Apply).	Level 3: Apply
CO5	Identify technology/research gaps and propose innovative/creative solutions (Cognitive knowledge level:Analyze).	Level 4: Analyze
CO6	Organize and communicate technical and scientific findings effectively in written and oral forms (Cognitive knowledge level:Apply).	Level 3: Apply

CO-PO AND CO-PSO MAPPING

CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
CO1	3	2	1	2	2					2		3	3		
CO2	3	3	2	3	3				1	2		3		3	
CO3	3	3	3	3	3	2			2	3	2	3			3
CO4	3	3	3	3	3	2			3	3	2	3			3
CO5	3	3	3	3	3	2	2		3	3	3	3	3		
CO6	3	2	3	2	3	2	2	2	2	3	3	3			3

3/2/1: high/medium/low

JUSTIFICATIONS FOR CO-PO MAPPING

Mapping	**Level**	**Justification**
101003/CS822U.1-PO1	H	Students apply interdisciplinary knowledge to solve real-world problems, aligning with applying engineering fundamentals.
101003/CS822U.1-PO2	M	Identifying and analyzing problems across domains requires problem analysis skills.
101003/CS822U.1-PO3	L	Modeling solutions implies designing components or processes.
101003/CS822U.1-PO4	M	Solving real-world problems may involve investigations and data synthesis.
101003/CS822U.1-PO5	M	Applying modeling tools and technologies to real-world problems uses modern tools.
101003/CS822U.1-PO10	M	Communicating models and solutions requires effective communication.
101003/CS822U.1-PO12	H	Real-world problem-solving promotes lifelong learning and continuous improvement.
101003/CS822U.2-PO1	H	Developing sustainable products requires applying engineering and domain knowledge.
101003/CS822U.2-PO2	H	Requires identifying socially relevant needs and analyzing them critically.
101003/CS822U.2-PO3	M	Directly aligns with designing solutions for real-world, socially relevant applications.
101003/CS822U.2-PO4	H	Product/process development involves experimentation and synthesis.
101003/CS822U.2-PO5	H	Involves using modern tools and technologies for design and modeling.
101003/CS822U.2-PO9	L	Developing as part of a team for sustainable development fosters team collaboration.
101003/CS822U.2-PO10	M	Communicating developed products clearly is essential.
101003/CS822U.2-PO12	H	Students develop a mindset for innovation, sustainability, and ongoing learning.

101003/CS822U.3-PO1	H	Effective functioning in teams draws from a strong foundational knowledge base.
101003/CS822U.3-PO2	H	Executing tasks often involves analyzing problems collectively.
101003/CS822U.3-PO3	H	Contributions to team-based solutions require design understanding.
101003/CS822U.3-PO4	H	Team tasks may require analysis and experimentation.
101003/CS822U.3-PO5	H	Collaborative tasks may require software and tools.
101003/CS822U.3-PO6	M	Teams consider the societal impact of their work.
101003/CS822U.3-PO9	M	Directly aligned with teamwork and leadership qualities.
101003/CS822U.3-PO10	H	Communication is vital for team performance.
101003/CS822U.3-PO11	M	Managing tasks within teams builds management capabilities.
101003/CS822U.3-PO12	H	Participating in teams prepares for lifelong learning.
101003/CS822U.4-PO1	H	Planning and executing tasks require domain and engineering knowledge.
101003/CS822U.4-PO2	H	Task execution involves analyzing problem components.
101003/CS822U.4-PO3	H	Planning aligns with designing efficient and feasible solutions.
101003/CS822U.4-PO4	H	Execution may involve experimentation and inference.
101003/CS822U.4-PO5	H	Efficient task execution requires modern tools and resources.
101003/CS822U.4-PO6	M	Ethical considerations are part of professional practice.

101003/CS822U.4-PO9	H	Involves coordination with teams for task execution.
101003/CS822U.4-PO10	H	Effective communication ensures clarity in task execution.
101003/CS822U.4-PO11	M	Planning and execution link directly with project management.
101003/CS822U.4-PO12	H	Time-bound tasks and ethical practices instill lifelong learning.
101003/CS822U.5-PO1	H	Proposing innovations relies on fundamental knowledge.
101003/CS822U.5-PO2	H	Involves deep analysis and research into existing gaps.
101003/CS822U.5-PO3	H	Creative solutions require strong design capability.
101003/CS822U.5-PO4	H	Innovation depends on experimental and research practices.
101003/CS822U.5-PO5	H	Using tools and simulations to develop new solutions.
101003/CS822U.5-PO6	H	Innovations must consider societal and ethical implications.
101003/CS822U.5-PO7	M	Creative solutions often target sustainability.
101003/CS822U.5-PO9	H	Research is often collaborative.
101003/CS822U.5-PO10	H	Communicating findings and proposals is essential..
101003/CS822U.5-PO11	H	Innovation projects involve management and finance awareness.
101003/CS822U.5-PO12	H	Research and innovation nurture a lifelong learning mindset.
101003/CS822U.6-PO1	H	Communicating findings requires a solid understanding of the content.

101003/CS822U.6-PO2	M	Writing and explaining results implies clear problem understanding..
101003/CS822U.6-PO3	H	Documenting involves representing design or process workflows.
101003/CS822U.6-PO4	M	Communicating research involves summarizing investigations.
101003/CS822U.6-PO5	H	May use tools for data visualization and modeling.
101003/CS822U.6-PO6	M	Written work must reflect social and ethical awareness.
101003/CS822U.6-PO7	M	Communicating solutions with sustainability in mind.
101003/CS822U.6-PO8	M	Communication must align with ethical standards.
101003/CS822U.6-PO9	M	Students will be able to associate with a team as an effective team player to identify, formulate, review research literature, and analyze complex engineering problems.
101003/CS822U.6-PO10	H	Students will be able to associate with a team as an effective team player for designing solutions to complex engineering problems and design system components.
101003/CS822U.6-PO11	H	Students will be able to associate with a team as an effective team player, use research-based knowledge and research methods including design of experiments, analysis, and interpretation of data.
101003/CS822U.6-PO12	H	Students will be able to associate with a team as an effective team player, applying ethical principles and committing to professional ethics and responsibilities and norms of the engineering practice.
101003/CS822U.1-PSO1	H	Students are able to develop Computer Science Specific Skills by modeling and solving problems.

101003/CS822U.2-PSO2	M	Developing products, processes or technologies for sustainable and socially relevant applications can promote Programming and Software Development Skills.
101003/CS822U.3-PSO3	H	Working in a team can result in the effective development of Professional Skills.
101003/CS822U.4-PSO3	H	Planning and scheduling can result in the effective development of Professional Skills.
101003/CS822U.5-PSO1	H	Students are able to develop Computer Science Specific Skills by creating innovative solutions to problems.
101003/CS822U.6-PSO3	H	Organizing and communicating technical and scientific findings can help in the effective development of Professional Skills..