Doodle2Clothing
A clothing design recognition and searching model from a doodle drawing

by

MD Fahim Afridi Ani
20101103
Asif Zubayer Palak
20101179
Shaikh Atisha Rahbath Dip
20101241
Arnab Pramanik
20101110
Tayaba Amin Medha
19301035

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science

Department of Computer Science and Engineering
School of Data and Sciences
Brac University
January 2024

Declaration

It is hereby declared that

1. The thesis submitted is my/our own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

Student's Full Name & Signature:

MD Fahim Afridi Ani

20101103

Asif Zubayer Palak

20101179

Shaikh Atisha Rahbath Dip

20101241

Arnab Pramanik

20101110

Tayaba Amin Medha

19301035

Approval

The thesis/project titled "Doodle2Clothing: A clothing design recognition and searching model from a doodle drawing" submitted by

1. MD Fahim Afridi Ani (20101103)
2. Asif Zubayer Palak (20101179)
3. Shaikh Atisha Rahbath Dip (20101241)
4. Arnab Pramanik (20101110)
5. Tayaba Amin Medha (19301035)

Of Fall, 2023 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on January 18, 2024.

Examining Committee:

Supervisor:
(Member)

DR. Farig Yousuf Sadeque

Assistant Professor
Department of Computer Science and Engineering
Brac University

Program Coordinator:
(Member)

Dr. Golam Rabiul Alam

Associate Professor
Department of Computer Science and Engineering
Brac University

Head of Department:
(Chair)

Dr. Sadia Hamid Kazi

Chairperson and Associate Professor
Department of Computer Science and Engineering
Brac University

Abstract

In recent years, pattern recognition has made significant progress; as computers can now identify and classify patterns of objects from various types of visual sources. These substantial forward strides in image and pattern recognition research can help users in ways that can improve the end-user experience for a lot of applications. In our research, we would like to solve a problem that shoppers usually face: finding a clothing based product with a specific design or pattern the shopper has in mind. With the use of Computer Vision and pattern recognition, our system will be able to recognize designs etched onto the product after classifying the type of clothing from doddle sketches provided by the users and we will be able to get the optimal images or products from the internet for the users. We will be training our model by collecting doodle sketches from end users by showing them an image of a clothing product and the model will extract features by comparing similarities between the sketch and the original picture based on its shape, color, design, etc and we expect to generate a search query which will be executed to find the product which is the closest resemblance to the doodle drawn by the user. With the help of our model, shoppers will be able to find their desired cloth without the hassle of excessive browsing; thus we will be enhancing Human-Computer interaction and provide a better and easier shopping experience for shoppers.

Keywords: Computer Vision; Pattern recognition; Doodle sketches; Human-Computer interaction

Acknowledgement

First and foremost, we express our gratitude and admiration to the Almighty Allah, without whose assistance our thesis would not have been successfully completed without significant obstacles.

Additionally, we express our gratitude to our Supervisor, Dr. Farig Yousuf Sadeque, for his invaluable guidance and unwavering assistance. We greatly benefited from his experience and guidance in establishing our study methods. His astute remarks prompted us to consider an alternative perspective and elevated the quality of our research.

Finally, we would like to express our appreciation to our esteemed parents for their altruistic assistance and motivation during our study endeavor.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction

Technology has become an indispensable component of our everyday existence. Consequently, we are inevitably reliant on this technology for socializing, entertainment, communication, and various other purposes. Of all these options, shopping online or offline is a significant aspect of entertainment in our lives. We often purchase numerous goods from both online and brick-and-mortar shopping establishments. Clothing retail has experienced significant growth within the retail industry. In the present era, characterized by a multitude of choices, the process of selecting an item might prove to be exceedingly challenging. However, if we possessed a technology capable of aligning our imaginative concepts with tangible clothing items and presenting us with a selection of outcomes, it would significantly simplify the process of shopping. By merely sketching a doodle of the desired product, this technology would generate the most similar output, thereby stimulating our creativity.

It is possible to assume that we all have been in a place where we all have spent an enormous amount of time searching for the right product online by pondering on what the right sentence will be to get the product whose image we have in mind or getting our time wasted by waiting on the salesman to check stock for the product and by roaming from shop to shop in the mission to locate the product of choice. As the world is advancing with the development of AI, all the business sectors are shifting towards an online or a digital module and trying to make the user experience as hassle-free and as easy as possible to stand out from their competitors. The innovations in computer vision opened up doors for endless possibilities and made our far-fetched dream into reality. Such as, the new Amazon Go, is a new era of convenient stores which eliminates the inconvenience for customers to wait in queue at the register and seek changes, causing delays.

Despite all these, there still are problems that need better solutions and can be solved by implementing AI, especially in developing countries where Artificial Intelligence is a new concept and is in its preliminary stages. There almost everything is done manually as lack of knowledge and exposure to the advancements or technologies, people are reluctant to make a digital pivot, thus costing them a lot of time and efficient business modules. Just imagine how efficient the business and browsing system will be if there is a way that people can doodle drawings of what design

or product they are seeking and searching for it. People no longer have to spend an extensive amount of time browsing online and shop owners can check whether they have similar products or not without manually going through their inventory. To make this possible the first obstacle will be to make a system that will be able to detect the product, next it should be able to classify it and then recognize the designed patterns in that. Ultimately, using all the information it should be able to generate a string for a search engine, of an online or an offline server system, through which similar products can be located.

With the help of edge detection, patterning recognition, caption generation, and reverse string search we should accomplish the task. Our proposed system will detect the edges of the object from the doodle, classify what it is and recognize the pattern or design of that object and generate a string describing the object explicitly to be used for searching. As we all are shoppers, we know much time will save us by making our shopping experience so much more efficient. Along with helping the customer, it will also aid business owners as well, they don't have to check their inventory each time, a single device will let them know what they have or not and an online platform business with a niche customer base will get more and easier exposure to their customer as their customer will be able to get them easily before tiring themselves by searching other popular global product sites.

A total of approximately 13,000 doodle images were created for this research, sourced from various individuals using diverse reference pictures. Each reference photo contains approximately seven doodle samples sourced from various origins. Initially, we gathered and processed the data using many resources such as individuals, image revolver, Script and other contributing variables. In addition, data augmentation has been utilized to generate a vast image dataset from the input drawings. Afterward, the samples were resized using image processing techniques to achieve a consistent size. Then, we applied additional necessary pre-processing procedures to properly prepare our image dataset, including Imagenet, ResNet18, ResNet50, YOLOv8x, LabelBinarizer, Xception, Blip-2, GPT-2, Vision Encoder-Decoder and others. Finally, two distinct approaches have been trained. For our first approach, we implement 4 models. They are- R-CNN with Xception, Blip-2, GPT-2 and Vision Encoder-Decoder, ResNet50+LSTM (n-Layers). For our second approach, we categorized distinct features into multilabel classification to conduct our multiple feature classfication.

1.2 Motivation

The objective of this research endeavor is to develop a highly efficient technology that can quickly locate the most similar clothing goods available online based on a doodle design. As more individuals opt to work remotely, we offer a convenient method for purchasing clothing that caters to people's creativity without any limitations. We intend to develop an interface for our proposed solution that will enhance user efficiency and ensure accurate selection of desired clothing items.

Chapter 2

Research

2.1 Problem Statement

Shopping nowadays is a tedious task as one might not get the product as expected because of not knowing the product-specific search string and availability of a vast amount of e-commerce websites. Also searching through different vendor shops consumes valuable time and energy. This creates an inefficient shopping experience for any person. We can develop a system based on AI and computer vision where we can generate a searchable string from doodle sketches that can fetch similar to exact products making the user experience much more efficient. Consider a scenario when someone attends a birthday party and notices something interesting while having fun. It can be anything like jewelry, clothing, shoes, and so on. That user might want to purchase one for themselves because they like it so much. They might ask about the product personally or visit every website that they can think of to find the goods, but doing so can take a lot of time, and there's a possibility that the product might not be found manually. The goal of this research is to use image identification, pattern recognition, and some algorithms, methodologies, and doodle drawing approaches to locate the closest or the exact product from the internet or a shop's database. Drawing doodles is the key to this endeavor as they are also based on human memories of the product. The user can use this to locate the nearest or the exact results. During this procedure, strings will primarily be generated and used to search for the product through the items available on the internet. Now, let's say the user wants to go to the market to get that item but forgets to bring photographs of related search items. Despite that, the stores will also have our proposed model, where the user can redraw to find the item they are looking for from the shop's inventory. Having said that, the google search engine also plays a crucial part in this situation because it helps to focus the search among different products. Which product is more likely and which is not can be determined using this model. The engine will produce images starting from the one that is closest to the original design and moving outward based on this comparison. Additionally, if anyone looks for the product online without sketching anything, they will still have to enter several websites that have the potential to access the user's files and steal their data while accepting cookies on other websites' pages, this can give some phishing websites access to the information or there can be potential viruses which can get injected into users' devices. However, this can be solved by computing the image retrieval process from the doodle drawing as this will immediately display the websites that

specifically contain the closest match to the object of our imagination which will reduce the number of websites that the user has to browse. Additionally, this can support the expansion of niche businesses. From this doodle drawing procedure, Google can filter out and rank that niche product first if the product matches the input the best. Normally, Google filters out and provides its users with the most relevant results. Because users notice their products first, niche enterprises may then see faster growth than they did previously. In addition, a lot of people find it difficult to explain a product that they might choose to describe because it is easier for them. Thus, they can design the product and achieve the best results they want from this model.

2.2 Research Objective

- The study aimed to identify comparable products online or from a vendor shop utilizing various ways of image recognition, pattern recognition, caption creation, reverse image search, and Google page ranking to generate product images mimicking the provided doodle drawings.
- The ability to recognize or detect items from photographs is becoming more and more important as a result of technological improvement and the emergence of AI.
- We will be able to identify proportionally similar products online or offline by simply making doodles on users' phones with the aid of AI-based models and the many methodologies and strategies suggested in this study.
- Upon completion, we intend to publish our work or launch it as a user-oriented product to make users' experience better.
- Users will be able to find their preferred fashion products considerably more quickly and save a lot of time.
- Niche online e-commerce sites and local vendor shops will be benefited because of having an efficient business with more exposure to clients.

Chapter 3

Literature Review

As the advent of the internet becomes more accessible to people across the world, physical services have surfaced into the online world as well making them accessible to any user on the internet. E-Commerce is a sector that spawned into cyberspace as enterprises and proprietors moved forward to selling their products online.

However, the vast collection of products online makes it difficult for consumers to find the product they are looking for. Often consumers know what they want, but they cannot digitally query it online due to the input limitations of search engines. But in recent years, image recognition technology has advanced significantly to mitigate such inconveniences.

To ease the process of finding a desired product, we are proposing a system that uses Edge detection, pattern recognition, reverse image search, and caption generation to find exact keywords to query on search engines.

To realize our envisioned work, we needed to find relevant work on the aforementioned technologies. To do so, our methodology for finding related works consisted of searching specific keywords such as "Edge detection" on scholarly search engines such as Google Scholar. Afterward, we attempted to choose works that had a greater number of citations and were published in recent years. With several papers at hand, we skimmed those works and analyzed their abstract to find a paper that best aligns with our envisioned work.

Our envisioned work uses Edge-Detection technology which was utilized in these papers:

A paper authored by Michael Felsberg, Rao Muhammad Anwer, Salman Khan, Jorma Laaksonen, Hisham Cholakkal, Fahad Shahbaz Khan, and Ankan Kumar Bhunia[23] proposes a new two-stage framework that is coarse-to-fine which relates a hand drawing to a set of structural relations of the different parts of the object body.

They used a probabilistic course sketch decoder on two datasets: Creative Birds and Creative Creatures, which contain over 8000 sketches of birds and creatures with annotations. The framework distinguishes the sketches by Comprehensive Sketch

Part Composition and Fine-level Diverse Sketch Generation. The metrics used are Frechet Inception Distance (FID) and Generation Diversity (GD).

A relevant study was performed by Zhuowen Tu and Saining Xie[7] that models a new CNN based on detecting edges that exhibit performance which is state-of-the-art on edge detection of native images. Feature learning that is multi-scale and multi-level and comprehensive image training and prediction are the issues addressed by the suggested method - Holistically-Nested Edge Detection (HED) using convolutional neural networks and deep learning models, the system performs image-to-image prediction.

HED framework is trained on datasets: Berkeley Segmentation and Benchmark (BSDS 500) and NYU Depth. Both datasets contained RGB and HHA features on which the proposed HED framework achieves a result of ODS = 0.782 and ODS = 0.746 respectively.

In another study by Srikumar Ramalingam, Ming-Yu Liu, Chen Feng, and Zhiding Yu[12], borders of objects are detected and assigned to one or more semantic categories. This was achieved by the use of CASENet, a trainable CNN architecture that is semantic edge detection which is category-aware. CASENet is evaluated over the dataset Cityscapes containing high-quality annotations. In terms of performance, CASENet: the suggested layered architecture, outperforms various methods which are state-of-the-art by a notable margin and shows that the proposed multi-label learning framework improves edge detection learning.

Another edge-detection method was studied by Xiang Bai, Ming-Ming Cheng, Xiaowei Hu, Kai Wang, and Yun Liu[9] to very efficiently produce edges that are high-quality and have the possibility of use in other vision tasks. They used Richer Convolutional Features (RCF) and proposed an original CNN architecture using features that are fine detail and semantic for detecting edges. By modifying the VGG16 network, on numerous available datasets such as Multicue (BSDS500) and NYUD, were able to gain outstanding performance having an ODS F-value of 0.811 (8 FPS) and 0.806 (30 FPS) respectively.
Pattern recognition is the second part of our envisioned work and the following papers discussed its methodologies.

Julia Lasserre, Katharina Rasch, and Rolland Vollgraf[17] aimed their study to recognize the pattern of clothing worn by a person with a neutral background. By feeding a dataset containing half and full-body cloth images by Zalando company - the convolutional network attempts to match the query image to different article images. The models they used to achieve this were: static-fc14-linear, static-fc14-non-linear, fc14-non-linear, 128floats-non-linear, fDNA-ranking-loss, fDNA-linear, all-in-two-nets, and Studio2Shop. Siamese CNN and static architecture were used to build their model with the right leg taking inputs produced by fDNA and the left consists of VGG16 CNN. After running about 20000 query images on 50000 Zalando articles, the proposed Studio2Shop model outperformed all other models due to its non-linear matching module.

Julia Lasserre and Rolland Vollgraf conducted another study with Christian Bracher[16] to merge their previous Street2Fashion model with the Fashion2shop model to create a new model.

Street2Fashion2Shop to relate street clothing images with shop clothing images. Chictopia10K and Fashionista was the datasets used to train the model with body parts divided separately and then the model was tested on DeepFashion In-Shop-Retrieval and LookBook datasets. To evaluate performance, the Street2fashion model was compared with the Mask-RCNN model, and the Model2shop model was compared with fDNA1.0-ranking-loss, fDNA1.0-linear, Studio2Shop with fc14, Studio2Shop with 128floats, Studio2Shop (with fDNA1.0) and Studio2Shop [20] with fDNA. The Street2Fashion model outperforms the Mask-RCNN model with a mean accuracy of 0.97 for the Mask-RCNN model and 0.985 for the Street2Fashion model. Kristine Guo, James WoMa, and Eric Xu's[14] study were conducted to build a classifier for the hand-drawn doodles from Google's Quick Draw! Game. To solve this, KNN with K-Means++ and Weighted Voting is implemented. As a comparison against the KNN methods, the CNN model is also introduced on this dataset, which is best known for learning the local features of an image. To train the model, the authors used 1% of Quick draw! a dataset containing over 50 million images across 345 categories. To check the accuracy of the models a lenient method An evaluation metric was used. Since there are many categories included in the dataset and to distinguish similar doodles, the methods were not only evaluated with raw accuracies of the model but also with a scoring metric.

Similar to the study mentioned above Aneeshan Sain, Yongxin Yang, Ayan Kumar Bhunia, Yi-Zhe Song, and Tao Xiang[24], conducted a study to create a style-agonistic SBIR framework that learns the variations in a sketching style of the same object category. The models used in this paper are Category-Level SBIR accepts a sketched query and tries to retrieve images of the same category. The second model is Fine-grained SBIR whose aim is directed at instance-level photo-sketch matching. Sketchy and the TU-Berlin extension were the two datasets used for category-level SBIR, and QMUL-Chair-V2 and QMUL-Shoe-V2 were utilized for FG-SBIR. Various methods were compared with the proposed framework on these four datasets like SOTA SBIR Triplet-ATTN Triplet-SN, CC-Gen, Triplet-RL for FG-SBIR and GDH, DSH, for category-level SBIR, Disentangled D-DVML, D-TVAE and other models such as B-Basic-SN, B-Cross-Modal, B-Meta-SN, B-SN-Group in this which the proposed model outperformed all other models in all four datasets.

Nguyet Minh Phu, Connie Xiao, and Jervis Muindi's[40] paper aim to recognize hand-drawn doodles using various machine-learning techniques and find their meaning. To solve this, the authors used Machine learning and deep learning algorithms. For the classical machine learning approach, Logistic regression and Support vector machine (SVM) was used. For the deep learning techniques, they chose Convolutional Neural Network (CNN) and Transfer learning from the ImageNet architecture. The dataset used to train these models was collected from Google's Quick Draw! Datasets containing millions of hand-drawn doodles of around 345 categories. For their machine-learning approach, the Logistic regression model was able to reach an accuracy of 79% and the SVM model performed worse than the Linear regression

model. For the deep learning approach, the simplified CNN outperformed all other algorithms and models in both precision and training time. The simplified CNN reached an accuracy of 81.83% on 50 different classes.

Afterward, our envisioned work aims to generate keywords based on the image - the papers below discuss similar problems:

Kudzai Felix Mawoneke, Xin Luo, Youqin Shi, and Kenji Kita[22] conducted a study to develop a model to classify fashion product images according to their categories. They used CNN models to find the vector features of the images and deduce which class the input item belongs to. They used a dataset called Fashion Product Images Dataset containing over 46,000 color images classified into 45 subcategories which include 'top wear', 'bottom wear', 'shoes', 'socks' etc. The dataset is split in such a way that the model is trained on 38,000 images and 6,446 images were used for testing. After training and testing the model, it can be deduced that the accuracy and precision of the image search system depend on the accuracy of the image features obtained using the CNN model. Classes that had the highest number of training images, attained a greater degree of accuracy in correctly classifying an item that belonged to it.

Darren Guinness, Edward Cutrell, and Meredith Ringel Morris[13] developed the Caption Crawler system to provide alternative texts to images on the internet for visually impaired users. Image type dataset was used, each labeled by either Machine-Generated or Human-Generated Captions. Machine-Generated captions used computer vision through deep learning to automatically generate captions for the images. Whereas the Human-Generated approach consisted of online workers identifying and creating tags for the images. Their system was implemented using computer vision while the human-authored captions were applied in replicas of those images, as many images can appear in multiple places online. During the test phase of uncaptioned images, a paired t-test on 72 websites showed t(72) = - 5.60, p¡.001. To check the performance of Reverse Image search engines, Yiltan Bitirim, Selin Bitirim, Duygu Celik, Ertugrul, and Onsen Toygar[20] conducted an investigation on the performance of Google's Reverse Image Search performance on finding similar images using fresh Image Queries from five different categories to identify how accurately Google's Reverse Image Search can find similar images. The dataset consisted of images of 5 categories, each having subcategories. A criterion called Precision, which measures the percentage of items that were retrieved that are pertinent to IQ was used to evaluate the effectiveness of the results.

$$Precision = \frac{number\,of\,relevant\,images\,retrieved}{total\,number\,of\,retrieved\,images}$$

The precision of various cut-off points was evaluated for each IQ and the Average Precision (AP) for each cut-off point was noted. Upon plotting the AP vs Cut-off point graph, it was found that the AP decreases with the increasing cut-off points for all the categories. In other words, the search engine's performance decreased as more images were evaluated.

G. Kavitha and E. Gurumoorthi[39] proposed a technique that uses Content-based Image Retrieval is a part of pattern recognition, image processing, and computer vision to find better matches to input images from reverse image searches. An image dataset is used for this proposed model from which, they are identified and separated based on their shape features, color features, and texture features. The proposed system uses the CBIR technique which searches images based on their pixels, meaning that CBIR does not rely on the metadata of the image, but instead focuses on the color, texture, and shape of an image from the dataset to produce a feature vector. These features of the input image are examined and compared to the feature vectors in the image dataset. Fuzzy heuristics are used to measure performance by determining three values shape, color, and texture. With these values, the Mamdani fuzzy inference method is used to perform fuzzy rules in the system to determine the similarity value between the input image and the sample image.

Ryan Kiros, Jimmy Lei Ba, Kelvin Xu, Aaron Courville, Ruslan Salakhutdinov, Richard S. Zemel, Kyunghyun Cho and Yoshua Bengio[8] proposed a model that describes the content of images by generating captions. The datasets used are Flickr9k, Flickr30k, and MS-COCO. Here two variation techniques are shown: a "hard" stochastic attention mechanism and a "soft" deterministic attention mechanism. In the attention mechanism model, the Encoder uses convolutional features and the decoder uses the LSTM network. A doubly stochastic regularization was introduced while training the "soft" model. By employing stochastic gradient descent and flexible learning rates, the model's two variations were trained. The Adam algorithm and RMSProp were determined to be efficient for the Flickr30k/MS-COCO and the Flickr8k datasets respectively. For annotation of the decoder the Oxford VGGnet pre-trained on ImageNet without fine-tuning was used. BLEU and METEOR were used to determine the performance where it was found that state-of-the-art performance on Flickr8k, Flickr30k, and MS-COCO and the state-of-the-art performance METEOR on MS-COCO were able to improve significantly.

A paper by Samy Bengio, Dumitru Erhan, Oriol Vinyals, and Alexander Toshev[5], introduces a deep learning recurrent architecture-based generative NIC model. The model employs the final hidden layer as an input to the RNN decoder, which creates sentences, and CNN as an image encoder. In RNN, the hidden state is updated using a linear function using LSTM and for representing images they use CNN. Sampling and BeamSearch were the approaches taken by NIC to generate sentences. The datasets used here: are Pascal VOC 2008, Flickr8k, Flickr30k, MS-COCO, and SBU. On the Pascal dataset, NIC produced a BLEU score of 59, above the current state-of-the-art of 25, whereas human performance achieved 69. The evaluation metrics employed were BLEU, METEOR, and Cider. We advance on Flickr30k from 56 to 66 and SBU from 19 to 28.

Xinlei Chen, C. Lawrence Zitnick[6] proposes a model that uses a Bi-directionals RNN model structure and also a language model usually has 3,000 to 20,000 words. For training, the Backpropagation Through Time (BPTT) algorithm was used and fine-tuned from the 1000-class ImageNet models that have already been trained. The datasets used here are PASCAL-1K, Flickr8k and 30K, and MS-COCO. Per-

plexity, BLEU, METEOR, and CIDEr metrics were used to gauge the outcomes. The model's BLEU and METEOR results (18.5 and 19.4, respectively, using the MS-COCO dataset) are barely inferior to those of humans (21.7 and 25.2), whereas CIDEr findings (52.1) are significantly inferior to those of people (85.4).

Jianfeng Gao, Xiaodong He, Zhe Gan, Chuang Gan, and Li Deng[10] proposed a novel framework StyleNet for producing engaging captions for pictures and videos. It creates a brand-new model component called a factored LSTM that automatically extracts the style factors from a corpus of monolingual text. It accomplishes this by using two types of data: styled monolingual text data and factual image/video-caption paired data. StyleNet combines Convolutional Neural Networks (CNNs) with Recurrent Neural Networks (RNNs) for image captioning. The Flickr 30K picture caption dataset served as the foundation for their new dataset, FlickrStyle10K. For evaluation, BLEU, METEOR, ROUGE, and CIDEr metrics are used. In terms of several automatic evaluation criteria and the ability to accurately represent the style factors in caption generation, StyleNet outperforms the baseline techniques when given a desired style.

The Abstract Scene Graph (ASG) structure was proposed by Qin Jin, Shizhe Chen, Qi Wu, and Peng Wang[21] to represent user motive at a finer level and manage what and how detailed the generated interpretation should be. In order to identify user motives and semantics in the graph and produce fitting captions, the original ASG2Caption model, which consists of a graph encoder that is role-aware and a language decoder for graphs, is introduced. These datasets, VisualGenome and MS-COCO were used. The evaluation metrics used are BLEU, METEOR, ROUGE, CIDEr, and SPICE. Due to the awareness of the control signal ASG, controllable baselines perform better than non-controllable baselines in controllability evaluation. The model surpasses controlled baselines that make use of the same ASGs control signal inputs in terms of performance. The model's method for producing captions is more diverse than comparable approaches when it comes to evaluating diversity, especially when looking at the SelfCider score.

Lastly, to find the product - our envisioned work needs to use keywords to query the search engines. The papers below investigate the rankings criterion of search engines:

Komal Kumar Bhatia, A. K. Sharma, and Neelam Duhan[3] tried to solve the problem of ranking the page on search engines using some algorithms from WWW (World Wide Web). The dataset is called Web mining, Data Mining where Web mining can split into Web Content Mining, Web Structure Mining, and Web Usage Mining. The technique used in this particular problem is the PageRank Algorithm. They also used the Weighted Page Rank Algorithm technique. The third method that they used is the Page Content Rank Algorithm which consists of Term extraction, Parameter Calculation, Term Classification, and Relevance Calculation. The last model used is the HITS Algorithm, consisting of the Sampling Step and the Iterative Step. The performance of those algorithms and techniques was measured by Page Rank - O(log N), less, more, Weighted Page Rank - ¡O(log N), less (higher than PR), more, Page Content Rank - O(m), more, less, and HITS - ¡(O(logaN) (higher than WPR), more (less than PCR), less.

Michael P. Evans[2] conducted a study to list the most widely utilized methods for getting a web page to rank well in Google. A tool from SEOBook.com called SEO for Firefox was used to collect the data (SEOBook, 2006). To estimate how many pages were assigned to it, they used a table from V7ndotcom Elursrebmem from their expanded database to Google Ranking. Additionally, they offer many data graphs for various categories, including date, page link, pages, page rank, websites, etc. The factors that affect the ranking that was discovered were that High PageRank in Google plays a major part in a page's rankings. Also, most of the SEOs goal is to attain a high PageRank, Comparative SEOs have many in-links to their page which may appear to have success, usage of the DMoz technique, having older domains for higher ranking and del.icio.us bookmarks are also proportional to the success of the page.

Moray Allan, Josip Krapac, Frederic Jurie, and Jakob Verbeek[4] introduce query-relative features-based generic classifiers.353 image search queries make up the new, sizable public data set in the paper. These offer the top images and related metadata that an online search engine has returned. This study made a data-driven recommendation for an image re-ranking technique that primarily relies on textual and visual aspects and doesn't necessarily require a separate model. The data set contains the original textual question for every one of the 353 search terms. There are more than 200 photos available for 80% of searches. The data set has 71478 pictures in total including the Fergus Dataset. This paper mainly uses Generic classifiers, Query-specific classifiers, and Binary logistic discriminant classifier models. They were able to make use of the model's re-ranking performance by evaluating the average accuracy (AP) for the scores it allocates to the photographs for every individual search and taking the mean overall searches. These outcomes can be contrasted using the precision of the photos and the mean average precision (MAP) of the rankings from search engines. Finally, they concluded that query-relative models significantly outperformed the raw search engine ranking, with a mean average precision increase of over 10%.

Caroline M. Eastman and Bernard J. Jansen conducted a study[1] to examine literature, research methods, and research outcomes from the perspectives of coverage, ranking, and relative precision. The datasets that have been used in this paper are Coverage, Relative Precision, and Ranking. There are many models/ techniques used in this paper which are the Selection of Queries, Selection of Documents, Searching Environment, Searching Rules, and Data Collection Method. The number of documents that MSN found compared favorably to the other two search engines. 570 of the 600 queries returned 10 or more results. 13 questions produced no results, while 17 inquiries produced at least one response but less than ten responses. From the total number of searches, the top 10 outcomes from each of the three search engines returned a total of 5,748 documents. 3,328 relevant papers and 2,420 irrelevant documents were gathered, in accordance with the evaluations of the four raters.

A paper authored by Maro Vlachopoulou, Christos Ziakis, Makrina Karagkiozidou, and Theodosios Kyrkoudis[19] put forth two ideas. The goal of the first section is to conduct a review of the literature on the elements that affect website rankings

in the SERPs and to identify the most important elements that improve ranking. The second section contains the results of our manual investigation using different keywords. This article includes 24 SEO elements as datasets.

The techniques used for this study are PRISMA methodology, The normalization of the data, the Spearman correlation coefficient, Three-factor correlation, and SEO techniques.

The quantity and quality of backlinks, as well as the bounce rate and SSL certificate, to a lesser extent, remain important variables despite changes in the algorithm.

Chapter 4

Background Study

Several pre-trained models, including Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), Multiclass Classification, Multilabel classification, and Multitask Classification, have been utilized in this research for classification purposes. In addition, we employed pre-trained models, including YOLOv8x, Resnet18, Xception etc. for the purpose of feature extraction. Below is some background information regarding these models:

4.1 Convolutional Neural Network (CNN)

The Convolutional Neural Network (CNN) [34] is primarily employed for image processing in deep learning models. Furthermore, it can be utilized for a wide range of data analysis and categorization activities. A Convolutional Neural Network (CNN) is an artificial neural network (ANN) that is specifically engineered to detect patterns and classify images based on these patterns. A Convolutional Neural Network (CNN) is primarily composed of an input layer, many hidden layers that include both convolutional and non-convolutional layers, and an output layer. The main purpose of the hidden layers is to detect and examine patterns within the photos. CNN operates based on the concept of weight sharing. Every layer of neurons has a weight and a threshold value, and they form connections with other neurons that have the same weight. Neuronal activation occurs when the output surpasses the threshold, resulting in the transmission of the output to the subsequent layer of the network. Here is an illustration of the CNN architecture:

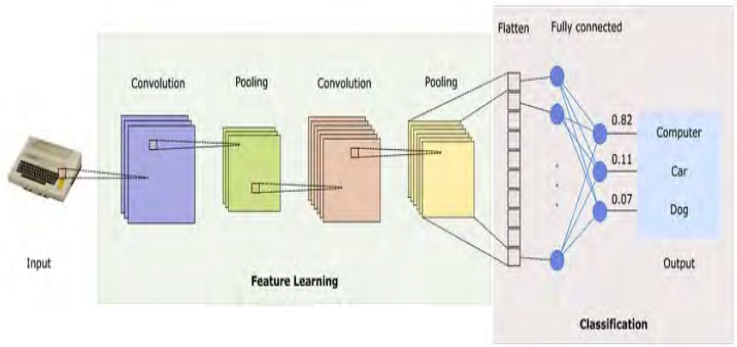

Figure 4.1: CNN Architecture [34]

A typical Convolutional Neural Network (CNN) comprises the following components. They are-

1. Convolutional layer
2. Pooling layer
3. Fully-Connected layer

1. **Convolutional layer:** This layer is a fundamental component of Convolutional Neural Networks (CNNs) responsible for performing convolution operations. The Kernel/Filter is the element in this layer responsible for executing the convolution operation (matrix). The kernel performs horizontal and vertical changes based on the stride rate until the entire image is scanned. The kernel is smaller in size compared to a picture, although it possesses greater depth. Consequently, if the image consists of three (RGB) channels, the kernel's height and width will be relatively small in terms of space, but its depth will encompass all three channels.

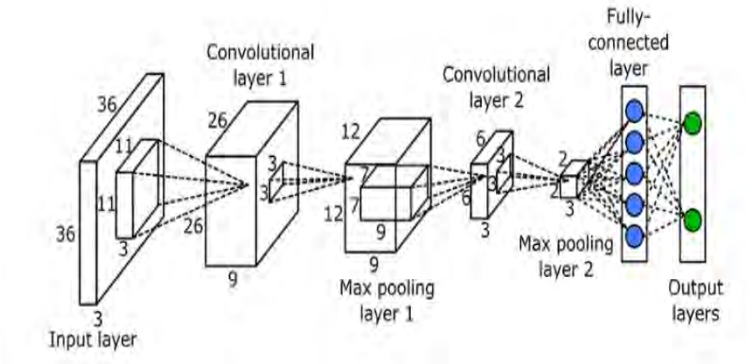

Figure 4.2: A visualization of Convolutional Layer [34]

We determine the dimension of the output based on this formula:

$$W_o = \left\lfloor \frac{W_i - F + 2P}{S} \right\rfloor + 1 \tag{4.1}$$

$$H_o = \left\lfloor \frac{H_i - F + 2P}{S} \right\rfloor + 1 \tag{4.2}$$

Where,
Wo = width of the output image
Ho = height of the output image
Wi = width of the input image
Hi = height of the input image
F = size of the filter
S = the value of stride
P = padding value

2. **The Pooling Layer:** This layer is responsible for decreasing dimensionality. It helps to decrease the computational resources needed to process the data. Pooling can be categorized into two distinct types: maximal pooling and average pooling. Max pooling returns the highest value obtained from the region of the picture covered by the kernel. Average pooling returns the mean value of all the pixels within the image region covered by the kernel.

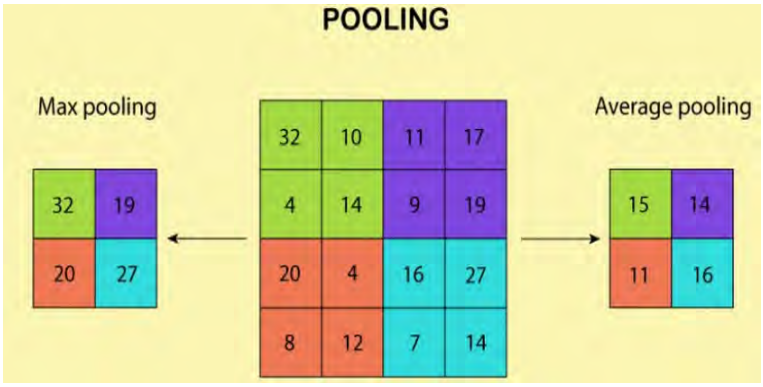

Figure 4.3: A visualization of Pooling Layer [34]

3. **Fully Connected Layer:** This layer operates on a flattened input, where each input is connected to every neuron. Subsequently, the compressed vector is transmitted via several further fully connected layers, where the usual mathematical functional operations are executed. The classifying method commences at this juncture. Fully connected layers are commonly located at the last stages of convolutional neural network topologies, if they exist.

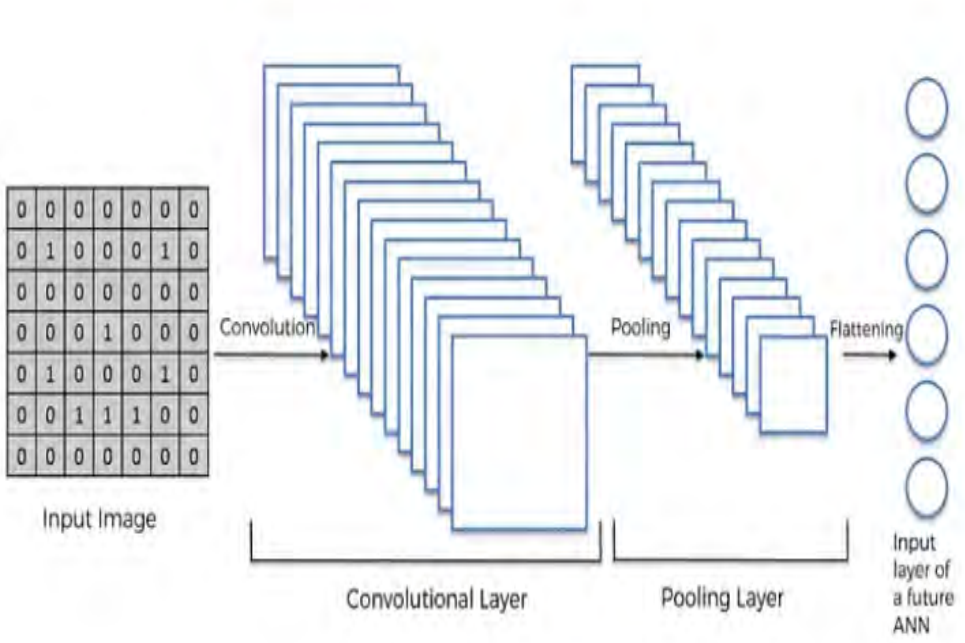

Figure 4.4: A visualization of Fully Connected Layer [34]

4. **Activation Function:** The activation function of the last fully connected layer is often different from the activation functions used in the other layers. Every action requires the choice of a suitable activation function. The softmax function is an activation function commonly used in multiclass classification problems. It takes the output real values from the final fully connected layer and normalizes them to target class probabilities. Each probability value varies between 0 and 1, and the sum of all probability values is always equal to 1. The ReLu activation function is widely used and serves as an alternative to both the sigmoid and tanh functions.

   (a) **ReLU function:** A rectified linear unit (ReLU) is an activation function that adds non-linearity to a deep learning model and resolves the problem of vanishing gradients. The formula follows:

$$\text{ReLU}(x) = \max(0, x) \tag{4.3}$$

   The function interprets the positive component of its input. It is a widely used activation function in deep learning. Here is a comparison of all the activation functions:

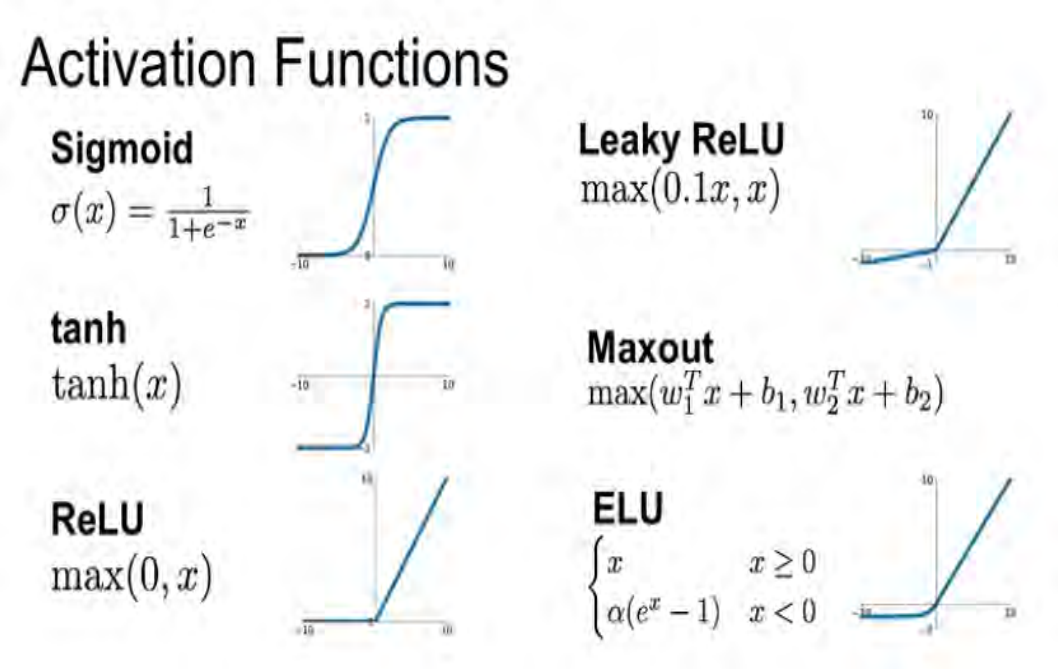

Figure 4.5: Activation Functions [15]

4.2 Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) [25] is a specific sort of recurrent neural network (RNN) structure that aims to solve the issue of the vanishing gradient problem commonly encountered in conventional RNNs. LSTMs excel at processing sequential data and time-series information. These devices possess memory cells capable of storing and retrieving information over long periods of time, rendering them well-suited for tasks such as natural language processing, speech recognition, and handwriting recognition. LSTMs are equipped with gates that regulate the transmission of information, enabling them to effectively record extensive interdependencies in sequential data. Here is an illustration of LSTM for better understanding.

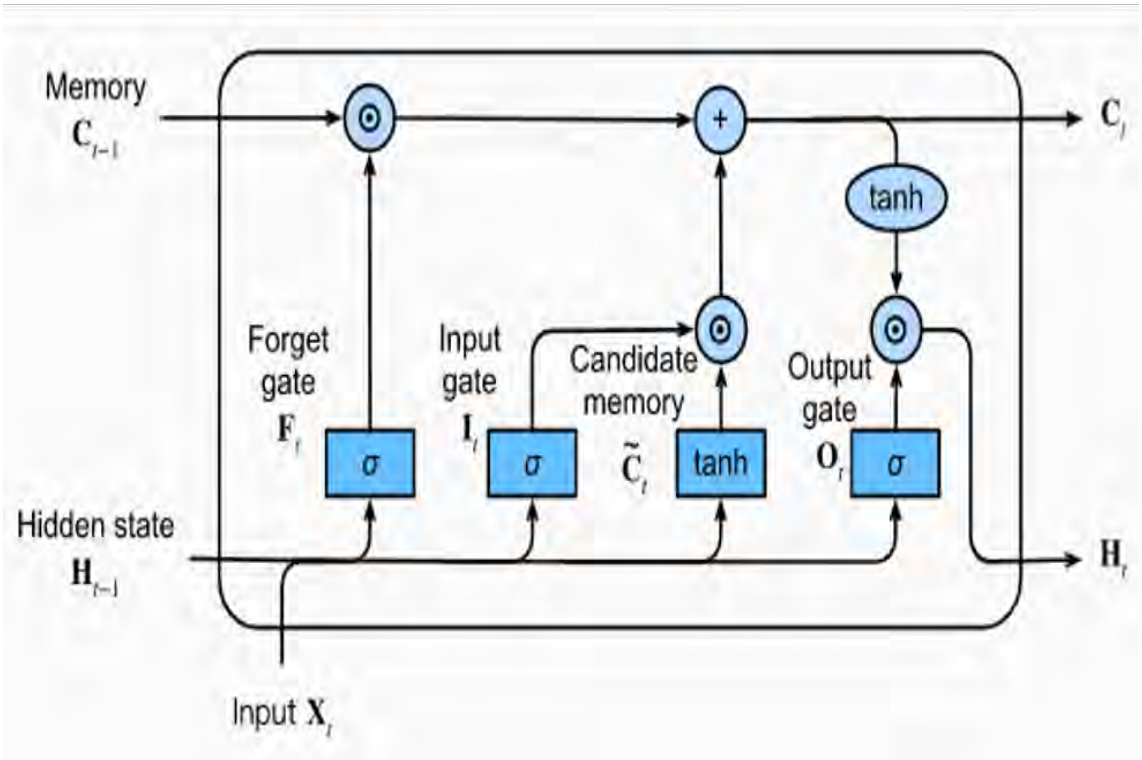

Figure 4.6: Long Short-Term Memory (LSTM) [25]

The LSTM architecture consists of a sequential arrangement of four neural networks and distinct memory units known as cells. Cells store information, whereas gates perform memory modifications. There exist three gates —

1. **Forget Gate:** The forget gate is responsible for eliminating irrelevant information from the cell state. The gate receives two inputs: xt (input at the specific time) and ht-1 (prior cell output). These inputs are multiplied with weight matrices and then added to the bias. The outcome of the calculation is then fed into an activation function, which produces a binary result. If the output for a specific cell state is 0, the information is discarded, however if the output is 1, the information is preserved for future utilization. The formula for the forget gate is as follows:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{4.4}$$

2. **Input Gate:** The input gate is responsible for incorporating valuable information into the cell state. To begin with, the information is regulated through the utilization of the sigmoid function. This function filters the values in a manner akin to the forget gate, employing the inputs ht-1 and xt to determine which data should be retained. Next, a vector is generated using the hyperbolic tangent function (tanh) that produces an output ranging from -1 to +1. This vector encompasses all the potential values from ht-1 and xt. Finally, the vector values and the controlled values are multiplied together to yield the relevant information. The formula representing the input gate is as follows:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{4.5}$$

$$c_t = f_t \cdot c_{t-1} + i_t \cdot \tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \tag{4.6}$$

3. **Output Gate:** The output gate is responsible for obtaining valuable information from the current cell state and presenting it as output. Initially, a vector is created by applying the hyperbolic tangent function to the cell. Subsequently, the information is modulated using the sigmoid function and selectively filtered based on the inputs ht-1 and xt for retention. Finally, the vector values and the controlled values are multiplied together to form the output and input for the subsequent cell. The formula representing the output gate is:

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{4.7}$$

After that, we can execute h_t for our next cell compilation of the LSTM which follows:

$$h_t = o_t \cdot \tanh(c_t) \tag{4.8}$$

4.3 Multiple Feature Classification

In our research, we used four disticnt features. They are Sleeve type, color, pattern and dress type. For each features, we used Multilabel Classification Model and integrated the four features altogether to generate query.

4.3.1 Multilabel Classification

This method is employed in scenarios where there are multiple classes and the data we aim to categorize may not correspond to any of the classes or may belong to all of them simultaneously. For instance, inside our research dataset, there could be a dress that is either a solid color or a blend of different colors. To distinguish distinct color combinations within the same class, it is necessary to employ multi-label classification, which enables the identification of several colors. That dataset is given below where we used multilabel classification:

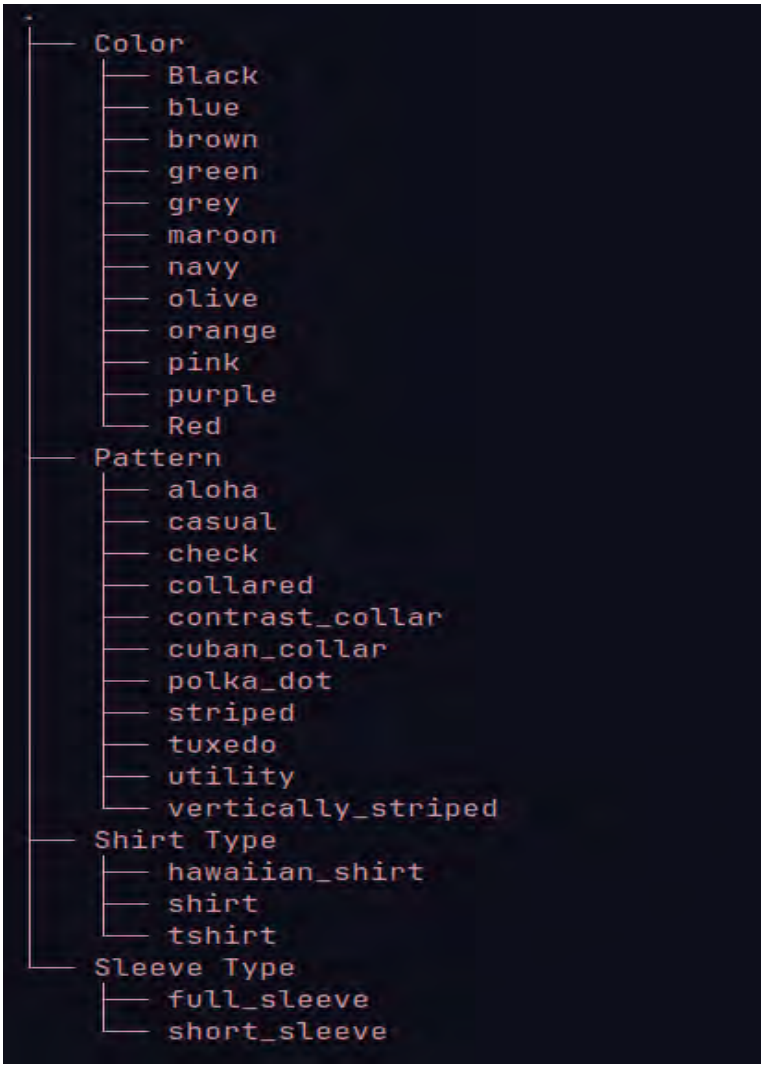

Figure 4.7: Multilabel Classification

4.4 Pre-trained Transfer Learning Models

We have employed various pre-trained transfer learning models to extract features. Each model plays a crucial role in filtering our dataset to ensure its suitability for the models.

4.4.1 YOLOv8x

YOLOv8 [31] represents the most recent version in the YOLO series of object detectors designed for real-time applications. It provides exceptional performance in terms of both accuracy and speed. YOLOv8 has different type models such as YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l, YOLOv8x.

These models have been utilized to integrate new features and improvements that make it a highly appropriate choice for various object identification tasks across a wide range of applications. We are utilizing the YOLOv8x model for processing an extensive dataset. The YOLOv8x model incorporates state-of-the-art backbone and neck architectures, resulting in improved feature extraction and object detection abilities. The system employs an anchor-free split Ultralytics head, leading to enhanced accuracy and a more streamlined detection process in comparison to approaches that depend on anchors. Its design aims to create a meticulous and swift equilibrium between precision and velocity, rendering it highly suitable for real-time recognition of objects in diverse application fields. The platform offers a range of pre-trained models that may be customized for various tasks and performance requirements, making it easier to discover the most suitable model for your specific use case. Below is a depiction of the workflow of YOLOv8.

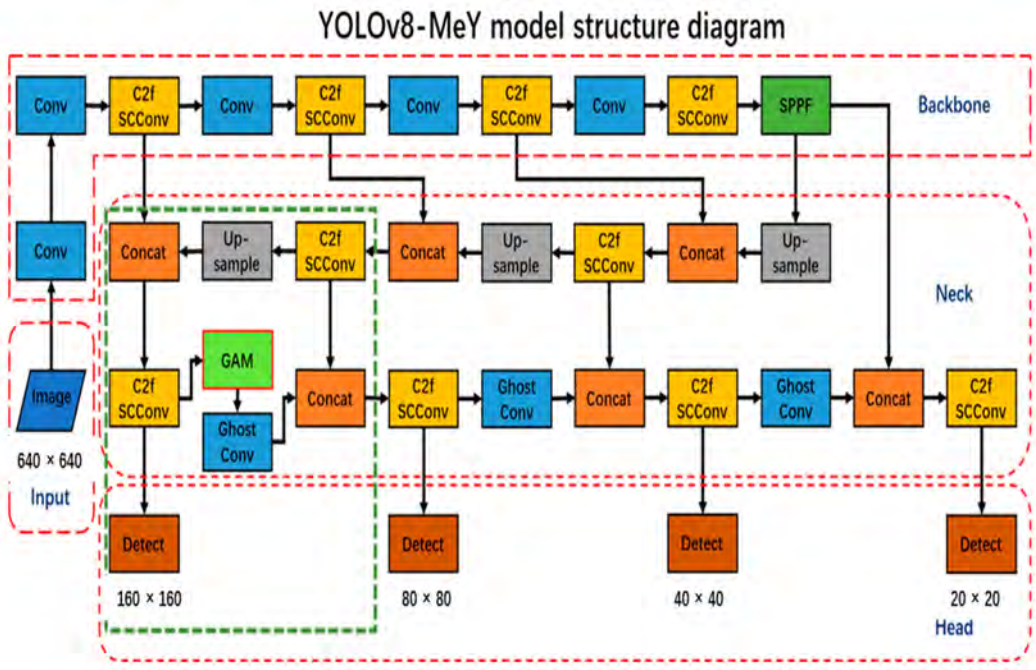

Figure 4.8: YOLOv8 Model Structure Diagram [31]

4.4.2 ResNet18

ResNet18 [29] is a particular iteration of the ResNet (Residual Network) design, initially presented by Microsoft Research in the publication named "Deep Residual Learning for Image Recognition" by Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. The numerical value "18" in ResNet18 denotes the network's depth, signifying that it comprises a total of 18 layers. The depth encompasses both convolutional and fully linked layers.Here is a visualization of a residual block:

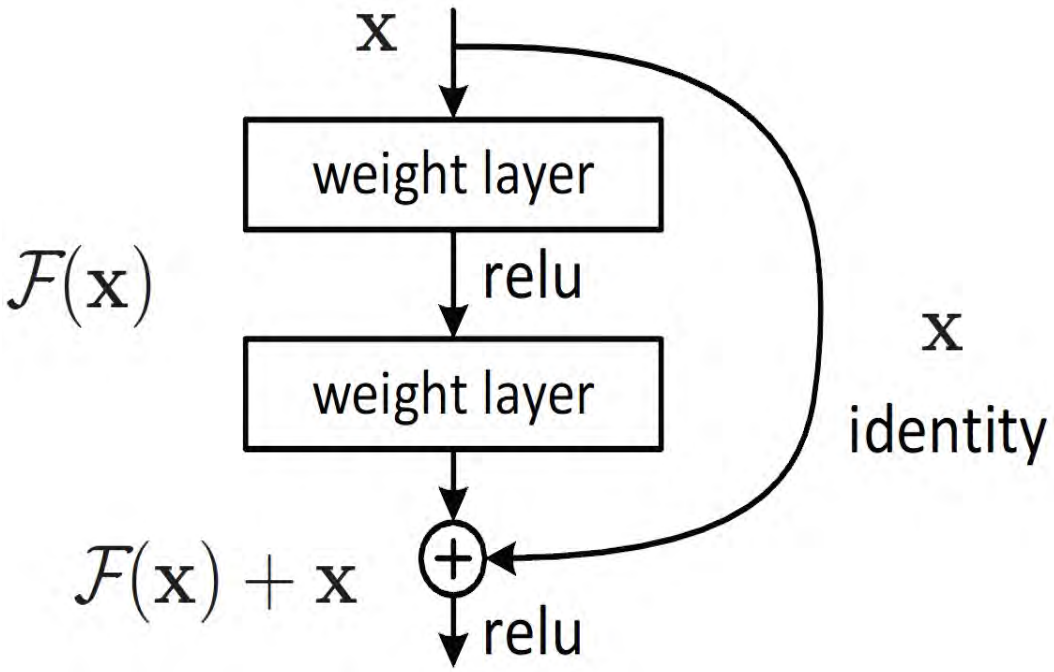

Figure 4.9: A residual block [29]

Some notable characteristics of ResNet18:

1. **Residual Blocks:** The core component of ResNet is the residual block. Each block comprises of two convolutional layers and a shortcut connection that bypasses one or more levels. The inclusion of shortcut connections effectively addresses the issue of the vanishing gradient problem and enables the neural network to effectively learn identity mappings.

2. **Layer Stacking:** ResNet18 consists of many stacked residual blocks. The arrangement of these blocks allows for the training of highly complex networks.

3. **Downsampling:** The architecture incorporates downsampling layers (strided convolutions) to decrease spatial dimensions, resulting in a broader receptive field for higher-level information.

4. **Global Average Pooling:** ResNet models frequently utilize global average pooling instead of traditional fully linked layers at the end of the network. This process reduces the number of spatial dimensions to a single value for each channel.

ResNet18 is commonly selected for situations that prioritize computing efficiency, such as on devices with limited resources or in real-time applications. This is because

ResNet18 is quite shallow. ResNet18, along with other ResNet architectures, has gained extensive usage and has been pretrained on big datasets, such as ImageNet. These preexisting models are frequently employed as tools for extracting features or as initial references for transfer learning in diverse computer vision assignments. Here is a diagram of ResNet18 architecture:

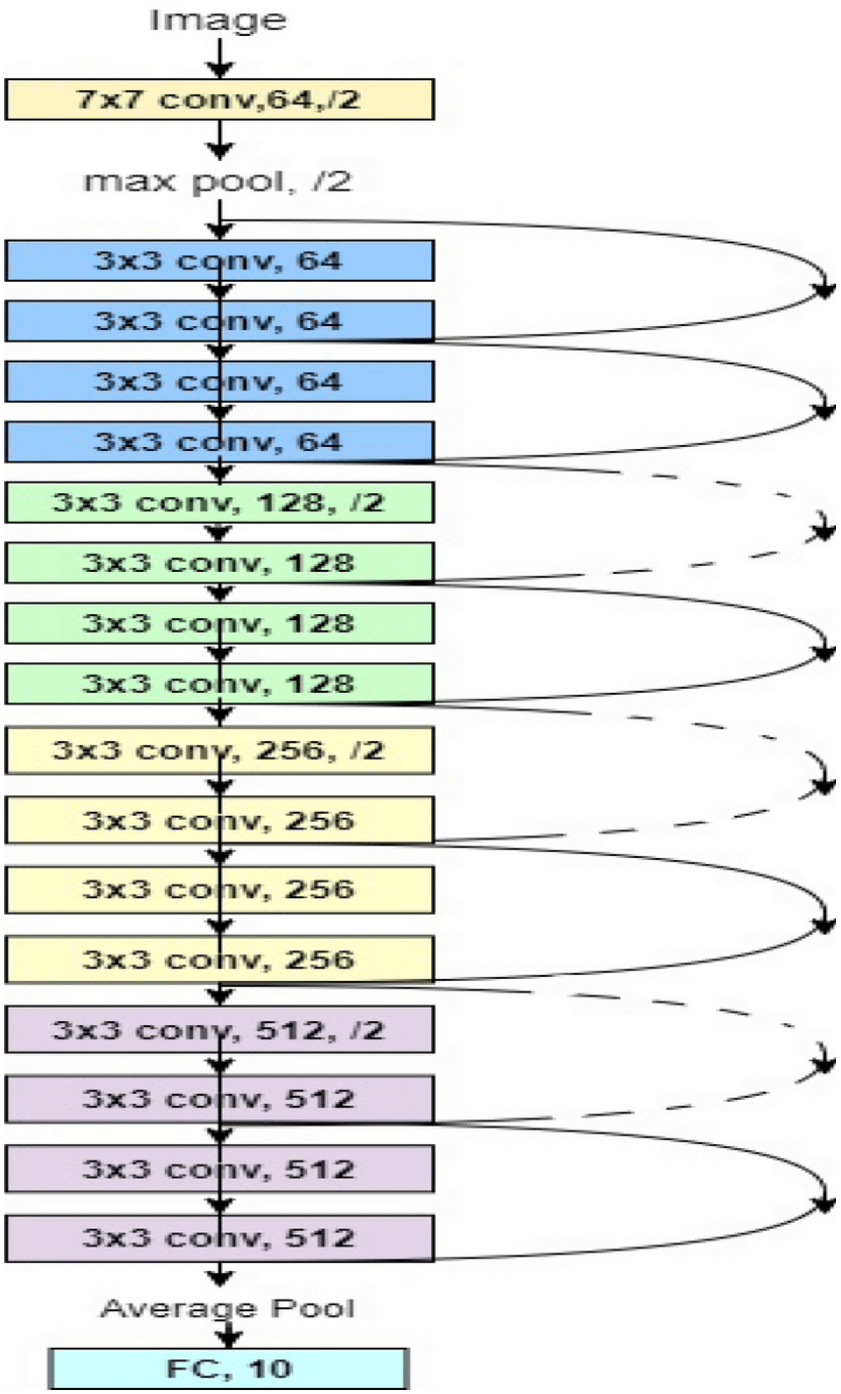

Figure 4.10: ResNet18 architecture [37]

4.4.3 ResNet50

ResNet-50 [27] is a distinct iteration of the ResNet (Residual Network) framework. ResNet is a complex neural network structure that uses convolutional layers to overcome the difficulties of training neural networks with many layers. The paper titled "Deep Residual Learning for Image Recognition" was presented at the 2016 Computer Vision and Pattern Recognition (CVPR) conference by Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun.
Some notable characteristics of ResNet50:

1. **Residual Blocks:** The primary breakthrough of ResNet is in the utilisation of residual blocks. A residual block consists of a shortcut link, also known as a skip connection or identity mapping, which bypasses one or more levels. This enables the network to acquire residual functionalities, facilitating the training of more complex networks.

2. **Bottleneck Architecture:** ResNet-50 has a bottleneck architecture within its residual blocks. Every block is composed of three convolutional layers: 1x1, 3x3, and 1x1 convolutions. The 1x1 convolutions decrease the dimensionality, enhancing the computational efficiency of the model.

3. **Depth:** The number "50" in ResNet-50 represents the network's depth, which refers to the number of layers it has. The model consists of 50 layers, comprising convolutional layers, activation layers, and batch normalisation layers.

4. **Pre-trained Model:** ResNet-50 is commonly trained in advance on extensive datasets, such as the ImageNet dataset. Pre-training enables the model to acquire universal characteristics from a wide variety of photos. Pre-trained ResNet-50 is frequently employed by researchers and practitioners as an initial model for transfer learning in targeted activities.

5. **Applications:** ResNet-50 is extensively utilised for diverse computer vision applications, such as image classification, object recognition, and picture segmentation. It has attained exceptional performance on various benchmark datasets.

6. **Skip Connections:** Skip connections within the residual blocks facilitate the smooth flow of gradients during backpropagation, hence alleviating the issue of vanishing gradients. This enables the training of highly complex networks.

Both ResNet18 and ResNet50 belongs to a series of ResNet architectures that have different depths, including ResNet34, ResNet101, and ResNet152. The more profound variations typically encompass more intricate characteristics but entail more computing demands. When utilising ResNet-50, it is customary to perform fine-tuning on specialised datasets or tasks in order to exploit the information acquired from extensive pre-training. The broad popularity of ResNet architectures in the deep learning community can be attributed to the availability of pre-trained models and their efficacy.

Given below is an illustration of ResNet50 architecture.

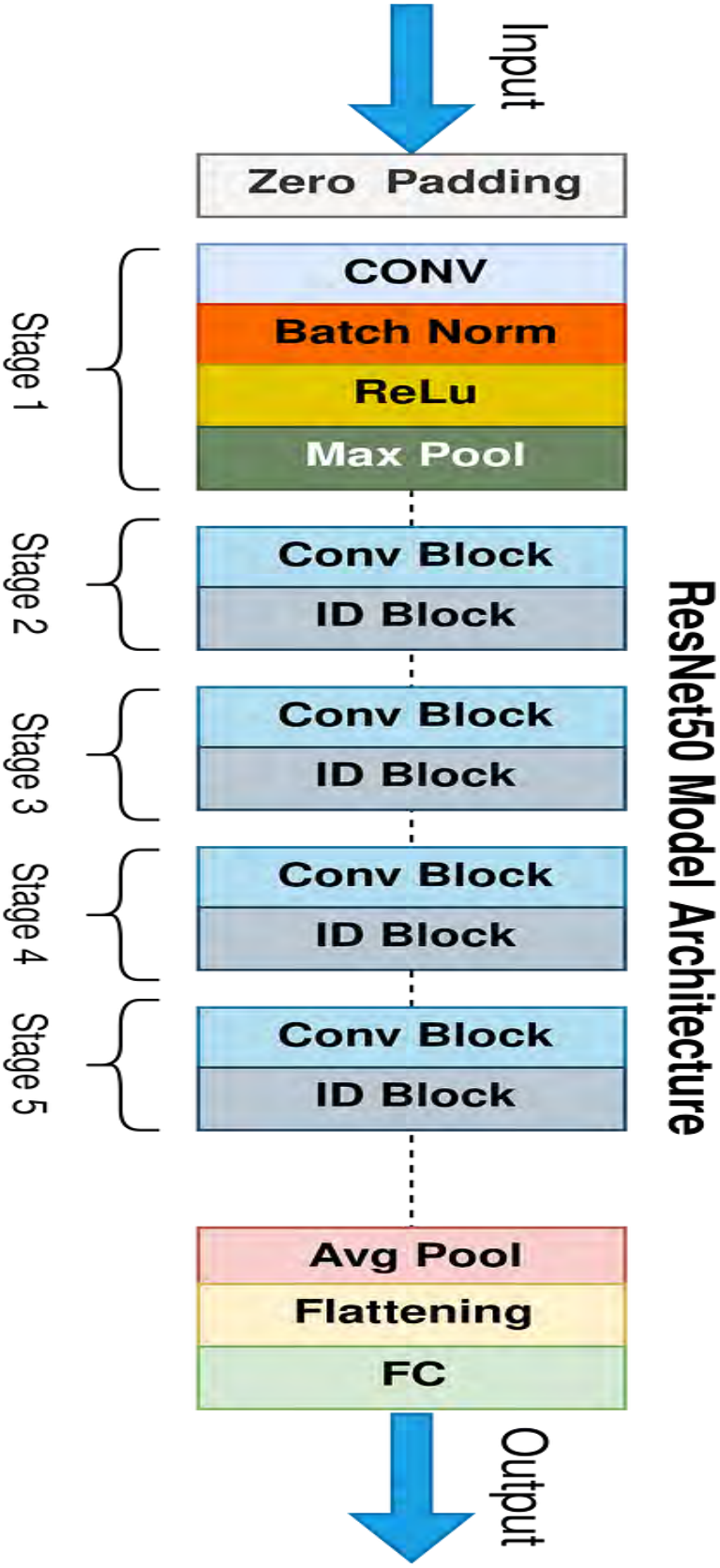

Figure 4.11: ResNet50 Model Architecture [27]

4.4.4 Xception

Xception [18], also known as Extreme Inception, is a deep learning framework that was developed as an expansion of the Inception architecture. The work titled "Xception: Deep Learning with Depthwise Separable Convolutions" was authored by Francois Chollet, the founder of the Keras deep learning toolkit, and was published in 2017. The Xception model is renowned for its utilization of depthwise separable convolutions, which are designed to efficiently capture intricate patterns while minimizing the amount of parameters in comparison to conventional convolutional layers. The main advancement of Xception is the utilization of depthwise separable convolutions instead of conventional convolutions, resulting in enhanced computational efficiency of the model. Here is a visualization of this model:

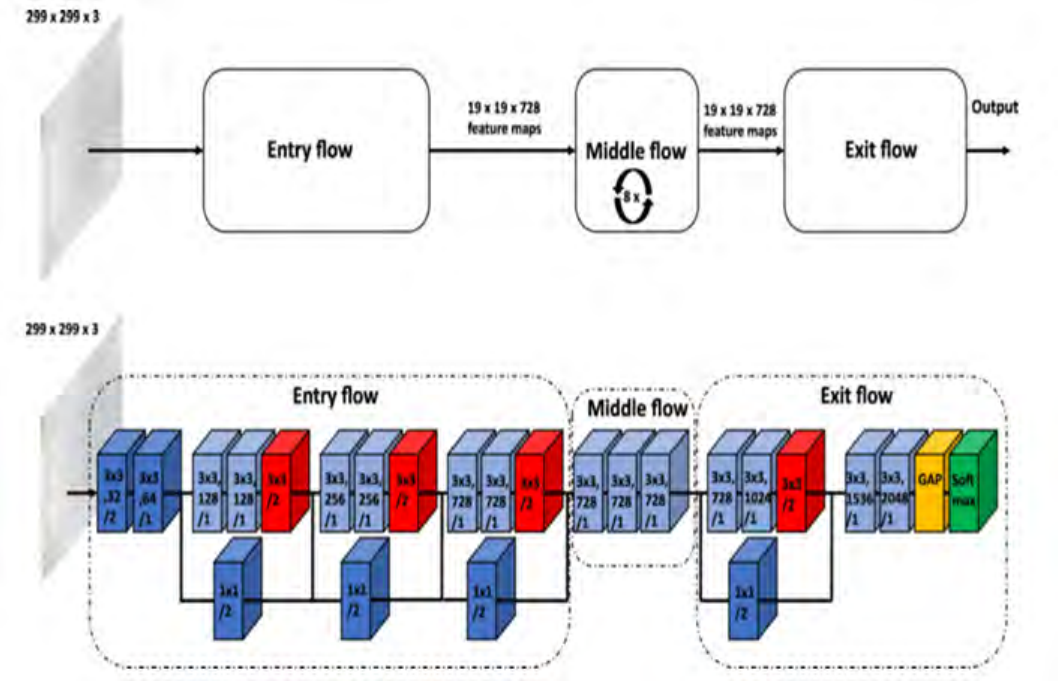

Figure 4.12: Xception Model Structure Diagram [42]

The Xception architecture comprises the primary components and features:

1. **Depthwise Separable Convolutions:** This refers to a type of convolutional operation that consists of two separate steps-

   (a) Depthwise convolution

   (b) Pointwise convolution

   The depthwise convolution applies a single filter to each input channel independently, whereas the pointwise The Xception model heavily utilizes depthwise separable convolutions, which involve performing a depthwise convolution followed by a pointwise convolution. Depthwise convolution applies spatial filtering to each channel separately. Pointwise convolution aggregates information from different channels.

2. **Ingress and Egress:** The Xception architecture is structured into an initial flow and a final flow. The entrance flow catches basic characteristics and

progressively expands the area of perception. The exit flow diminishes the spatial dimensions and generates the ultimate output.

3. **Residual Connections:** Xception employs residual connections, similar to ResNet, to aid in the training of extremely deep networks. The user did not provide any text. Residual connections facilitate the smoother propagation of gradients throughout the network.

4. **Global Average Pooling:** Xception commonly employs global average pooling as a preprocessing step prior to the ultimate fully linked layer. Global average pooling is a technique that condenses the spatial dimensions of each channel into a single value, resulting in a more concise representation.

Xception has demonstrated strong performance in a range of computer vision tasks, including image categorization and object recognition. It is especially suitable for situations where computing efficiency is important, making it valuable in contexts with limited resources. It is commonly employed as a feature extractor or backbone in transfer learning situations, when pre-trained models on extensive datasets (such as ImageNet) are adjusted for specific tasks.

4.4.5 Blip-2

Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models. BLIP-2 [32], also known as Bootstrapping Language-picture Pre-training, is an artificial intelligence model capable of executing diverse multi-modal tasks such as visual question answering, picture-text retrieval (image-text matching), and image captioning. The system has the ability to analyse an image, comprehend its content, and produce a pertinent and succinct caption. BLIP-2 facilitates the comprehension of images by language models while preserving their original structure. This is accomplished by employing a querying transformer (q-former) which serves as an intermediary connecting the image and the language model.

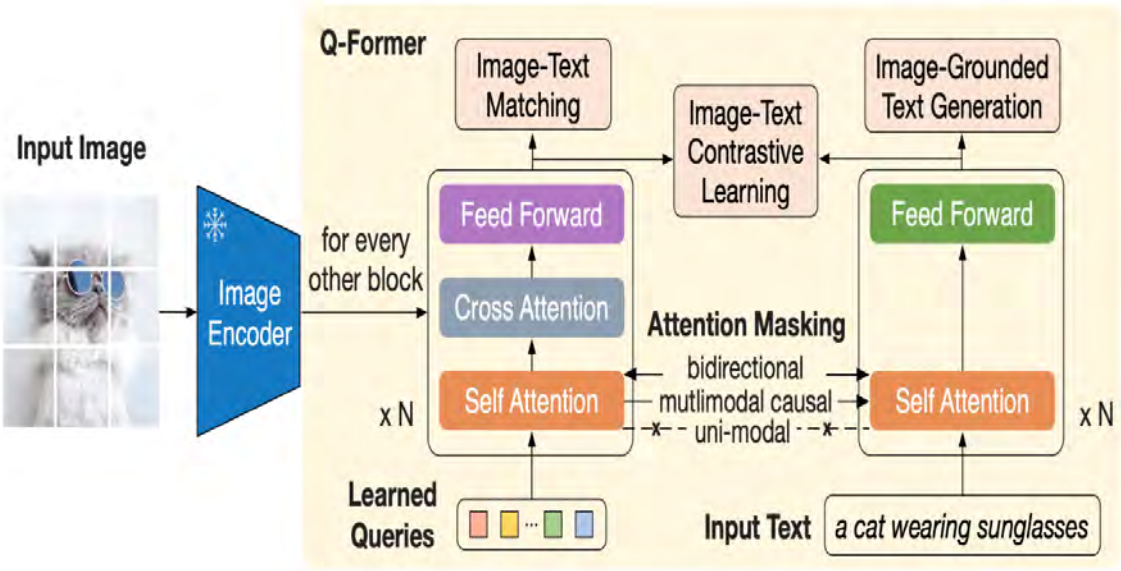

Figure 4.13: BLIP-2 Architecture [32]

Q-Former is a transformer-based architecture consisting of two sub-modules: (1) an image transformer that interacts with the visual features obtained from the frozen image encoder, and (2) a text transformer that is capable of encoding and decoding texts. The system utilises a collection of trainable querying vectors to extract pertinent visual characteristics that capture the most informative portion of the text associated with the image. Here is an illustration for Q-Former in BLIP-2.

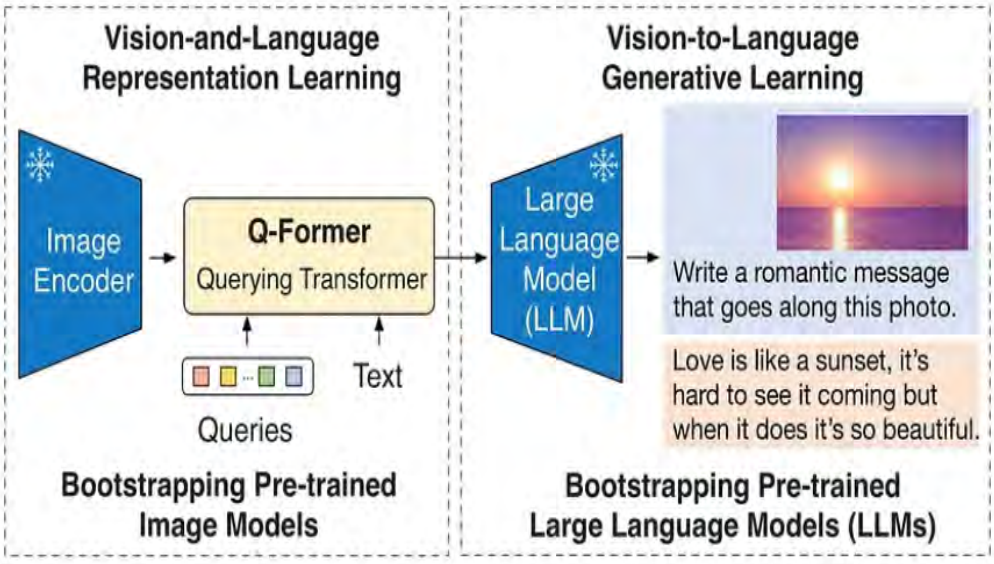

Figure 4.14: Q-Former in BLIP-2 Architecture [32]

Q-Former utilises a collection of query vectors that may be adjusted through learning. It undergoes pre-training in two distinct phases: The problem of the frozen language model not being trained on image data can be addressed by employing two stages:

- Vision-language representation learning with a fixed image encoder
- Vision-to-language generative learning stage with a fixed text encoder

This approach ensures that the extracted visual representations are accurately interpreted with additional assistance.

4.4.6 GPT-2

GPT-2 [36], also known as "Generative Pre-trained Transformer 2," is an advanced artificial intelligence model for language processing that was created by OpenAI. The model is the successor of the original GPT and was introduced in a research article titled "Language Models are Few-Shot Learners," published by OpenAI in 2019.
The salient attributes and distinguishing traits of GPT-2 encompass:

1. **Transformer design:** GPT-2 is constructed based on the Transformer design, which was initially presented in the publication "Attention is All You Need" by Vaswani et al. in 2017. Transformers have emerged as a conventional framework for a wide range of natural language processing (NLP) activities.

2. **Diverse Data Pre-training:** GPT-2 undergoes pre-training using a vast dataset that encompasses various and wide sections of the internet. The model undergoes pre-training where it is exposed to a diverse array of text from various areas and subjects.

3. **Extensive:** The scale of GPT-2 is one of its notable characteristics. The standard version of the model contains an extensive set of parameters, particularly 1.5 billion, which positions it as one of the largest language models at its debut.

4. **Generative Nature:** GPT-2 possesses the ability to generate language that is both coherent and contextually appropriate, making it a generative model. When provided with a cue, the system is capable of generating text passages that closely resemble human writing in terms of both style and content.

5. **Few-Shot and Zero-Shot Learning:** GPT-2 demonstrates its capacity to do few-shot and zero-shot learning. Few-shot learning pertains to the model's ability to extrapolate from a limited number of examples given in the prompt, whereas zero-shot learning refers to the generation of text on subjects that were not encountered during training.

6. **Contentious Launch:** GPT-2 first attracted attention and generated controversy due to concerns about its potential for misuse. OpenAI initially refrained from releasing the entire model because to these concerns, but eventually chose to make the complete model available.

GPT-2 has been utilised for diverse natural language processing endeavours, including text augmentation, condensation, interpretation, and imaginative composition. Here is an illustration of GPT-2 Model Architecture:

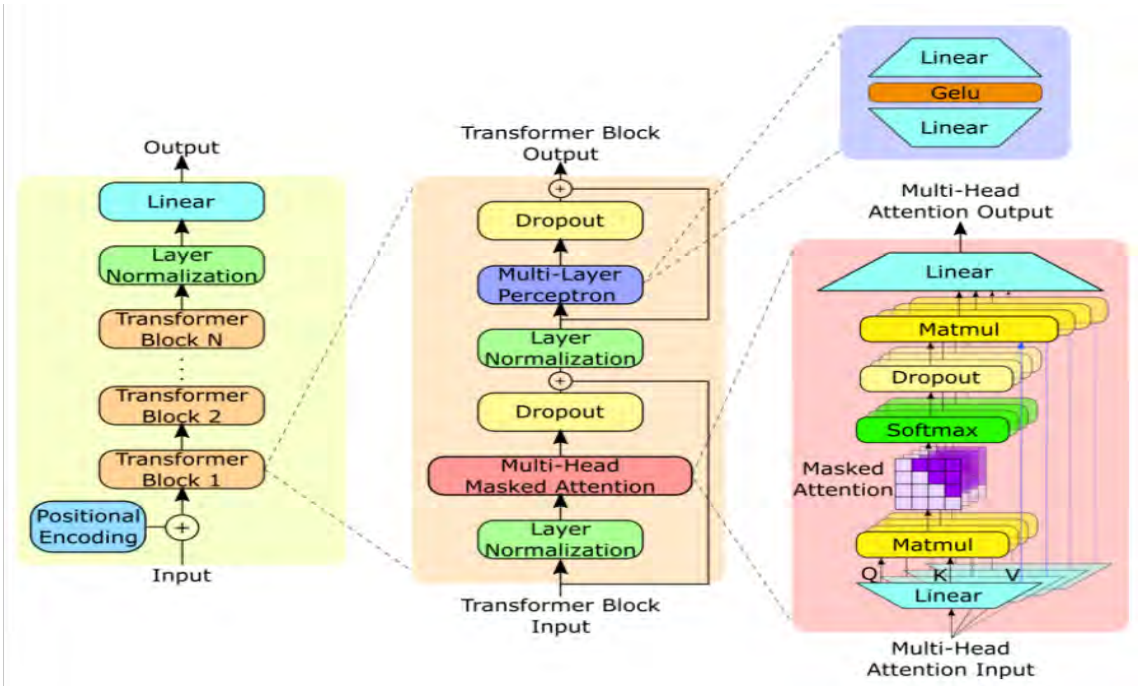

Figure 4.15: GPT-2 Model Architecture [38]

4.4.7 Vision Encoder-Decoder

Vision Encoder-Decoder [33] architecture comprises two primary components: an encoder and a decoder.

1. **Encoder:** The encoder's primary role is to capture and extract pertinent characteristics from the input data, usually in the form of an image. The input is encoded into a condensed representation or feature vector. Convolutional neural networks (CNNs) are frequently employed as encoder networks in computer vision tasks. These networks have the ability to accurately extract hierarchical and spatial characteristics from images.

2. **Decoder:** The decoder receives the encoded representation from the encoder and proceeds to reconstruct or generate an output. The process operates in the opposite direction, converting the condensed feature vector into the intended output format. The decoder is typically engineered to generate an output that possesses identical dimensions to the input, particularly in tasks such as picture segmentation or reconstruction.

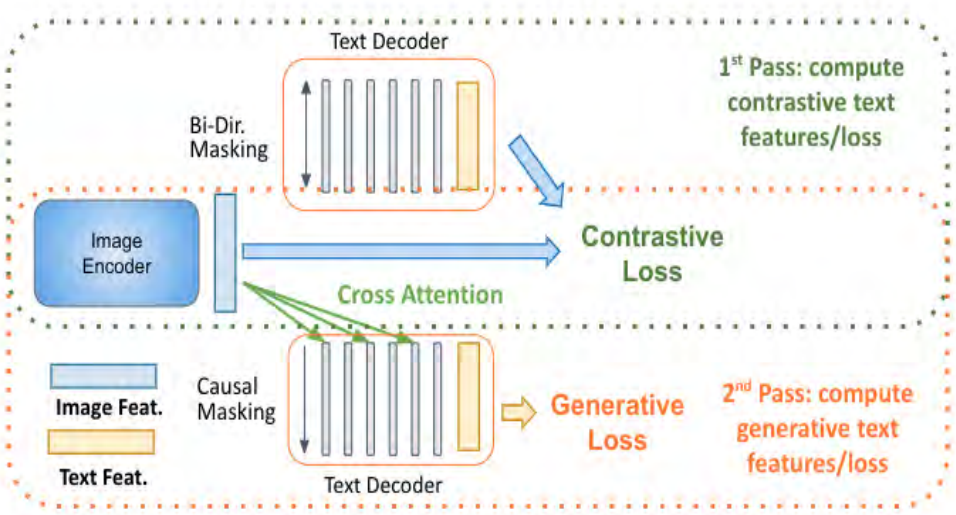

Figure 4.16: Vision Encoder-Decoder Architecture [33]

The vision encoder-decoder architecture is extensively employed in many computer vision applications.

- **Image Segmentation:** This refers to the process of dividing an image into many segments or regions based on certain criteria, such as colour, texture, or shape. It is a fundamental task in computer vision and is used in various applications In tasks such as semantic segmentation, the encoder extracts features from the input image, while the decoder produces a segmentation map that assigns a particular class label to each pixel.

- **Image-to-Image Translation:** In tasks such as style transfer or image synthesis, the encoder captures the style or content of a picture, while the decoder produces a new image with the appropriate attributes.
- **Image Reconstruction:** Auto encoders utilise an encoder to condense the input image into a latent representation, which is then used by the decoder to recreate the original image. This technique is frequently employed for the purpose of reducing noise and compressing images.
- **Image Generation with Conditions:** Conditional Generative Adversarial Networks (cGANs) employ an encoder-decoder architecture to create images based on specific input circumstances.

The vision encoder-decoder design leverages the expressive capabilities of deep neural networks, enabling it to acquire intricate hierarchical characteristics from visual data. This architectural design has played a crucial role in pushing forward the latest developments in different computer vision jobs.

Chapter 5

Methodology

5.1 Research WorkFlow

The research study follows a systematic technique that involves the following steps:

1. Data collection
2. Data pre-processing
    - Background removal and cropping
    - Data Augmentation
    - Resize
    - Rescale
    - Encoding
3. Selection of suitable pre-trained models for our research
4. Product detection using the chosen pre-trained models
5. Train-test split of the dataset
6. Implementing the dataset with two approaches
    - **First Approach:** In this approach, we have implemented 4 different models. They are-
        - RCNN with Xception
        - Blip-2
        - GPT-2+Vision Encoder-Decoder
        - ResNet50+LSTM (n-Layers)
    - **Multiple Feature Classification Approach:** In this approach, we have implemented Multi label classification for each features. The Features are-
        - Sleeve Type
        - Pattern
        - Color
        - Dress Type

1. **First Approach:** The process begins with a YOLOv8x model that has already been trained. This model is used to find the edges of things of interest in pictures. These boxes are then used to crop the pictures to only show the parts that are needed, making a set of doodle images. The doodle pictures that were taken out are then fed into an Xception Encoder-Decoder network, which uses them to take out features. These features capture the visual characteristics of the doodles. Subsequently The captions connected to the doodle images are also analysed.Irrelevant words and punctuation have been eliminated, and numerical values have been converted to their respective written forms. The purpose of this phase is to cleanse and standardize the captions in order to improve the following processing.This technique includes various caption generation models, each presenting its distinct methodology for generating captions, such as R-CNN with Xception encoder decoder, Blip-2, GPT-2+Vision Encoder and Decoder, RestNet+LSTM (n-Layers). As a whole, each of these models produces a potential caption for doodle images.

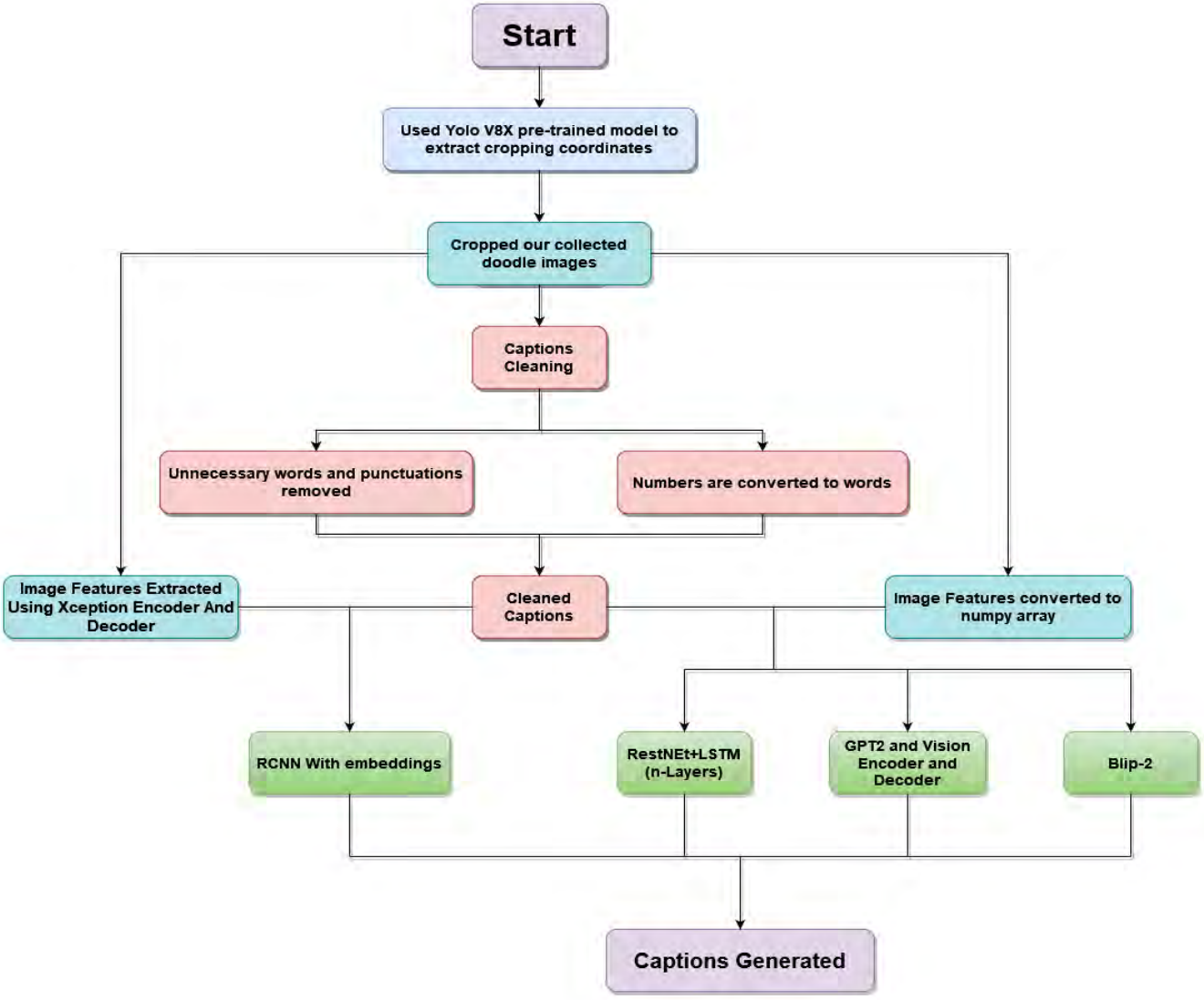

Figure 5.1: The workflow of the First Approach

2. **Multiple Feature Classification Approach:** This approach involves the development of a multiple feature classifier that is specifically built to categorize drawings and generate search queries simultaneously. The output of this classifier will be multi-label. This technique involves the creation of separate classes for different attributes of the dress, such as pattern, color, and type. Furthermore, they are categorized into multiple discrete classes and labels. For example, the color class could be composed of various subclasses such as red, blue, yellow, white, green, and it can also have various combination which is why multi-label classification is necessary. Subsequently, we trained our separate classes to accurately identify the inherent characteristics, and we combined all the classes to achieve a full depiction of the class prompts. We employed the merged image to create the intended inquiry that closely corresponds to our amalgamated product. An illustration depicting the workflow of multiple feature classification is:

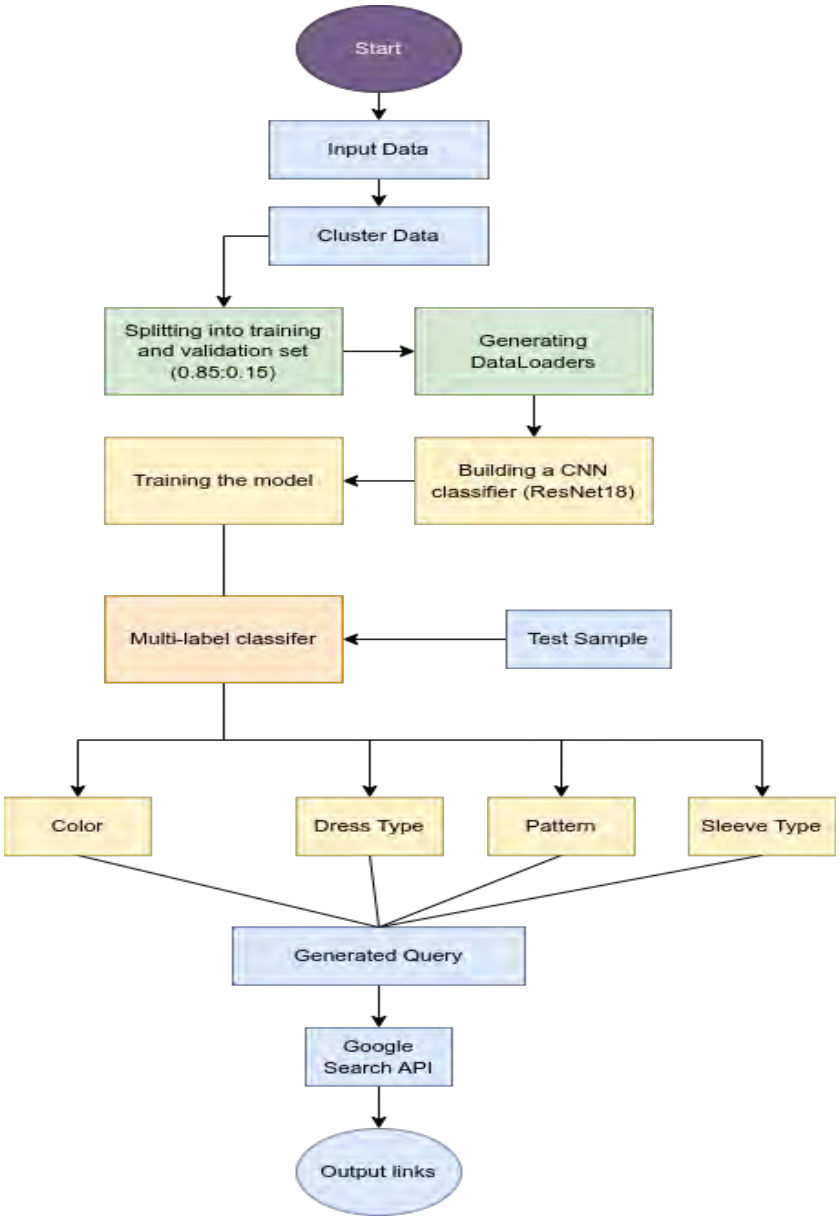

Figure 5.2: The workflow of Multiple Feature Classification Approach

5.2 Dataset

Our research utilizes a dataset consisting of around 13,000 sample images. We gather them from the beginning with intense manual labor. Our initial approach involved gathering reference photographs from various apparel categories. This led to the creation of 6 to 7 sample doodle drawings for each reference image. Additionally, we developed a spreadsheet to track the links associated with these images. To generate captions for the images, we using image revolver. We employed distinct preprocessing techniques for our two distinct models. Each data required to be adjusted according to its corresponding requirement based on the suggested model.

5.2.1 Dataset Collection

We systematically executed a series of steps to prepare our dataset for utilization in our proposed models. Following is an in-depth explanation of each step implemented to ensure the success of our data collection process.

1. **Collect Reference Pictures:** The first thing we need to do is decide what clothes we want to gather. In this case, we talked about full and half-sleeve t-shirts and shirts, jackets for guys, as well as t-shirts, shirts, and crop tops for women. We look for high-quality reference pictures of the clothes that have been chosen. These pictures should show how the clothes come in different styles, colors, and designs. We get these pictures from online fashion booklets, e-commerce sites, and other sources that are useful.

2. **Essential data Collection:** We maintain a record of the Google search query, the corresponding Google link, and the name and link of each specific image for every reference photo for future usage.

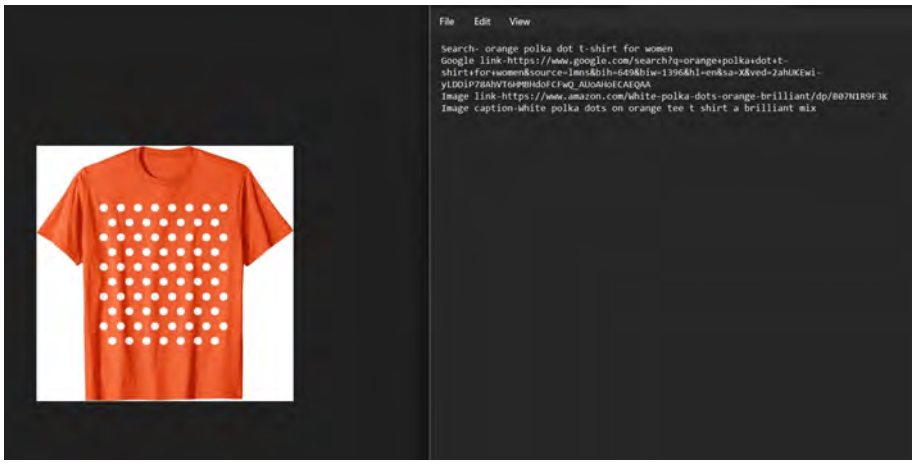

Figure 5.3: A sample reference picture with information

3. **Crowd-Sourcing:** We set up a crowd-sourcing website so that people could send us photos of the clothes. We give people reference pictures of the clothes and ask them to draw or paint them. We also draw some of the clothes ourselves. We get pictures of clothes in the RGB format. We use graphic screens and MSPaint to make our sample doodle drawing, which will be fed into the model as data.

Figure 5.4: A reference picture with corresponding doodle sample

4. **Image Revolver:** We use a platform like Image Revolver, which lets users show reference pictures to other users and get their written statements or captions, to gather textual data about the clothing items. We give users a link to the reference pictures and ask them to write detailed captions based on what they see. Here, we do these things:

    - Add the reference pictures for each type of clothing to the platform for crowd sourcing.
    - Give different people or contributors a link to these pictures.
    - Ask them what they think when they first see these images.
    - Collect their response for caption generation for our future usage.

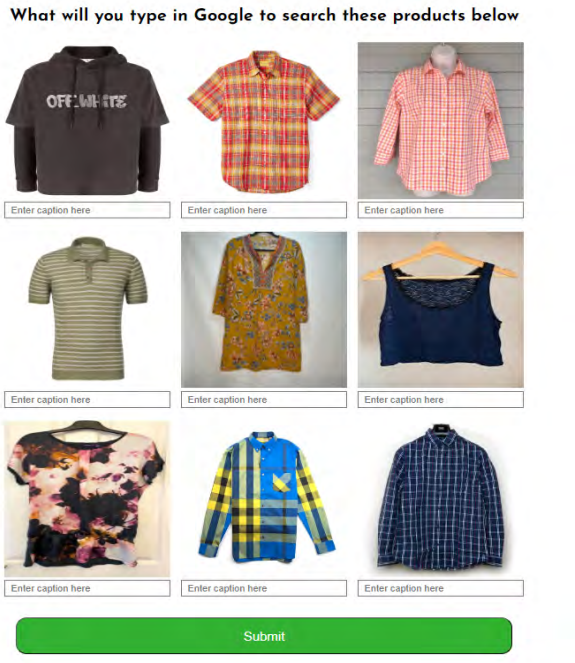

Figure 5.5: Image Revolver

5. **Organize Dataset Folders for Individual Models:** We collected reference images together with their corresponding information and samples, and established distinct folders for each of our suggested models. In one technique, we generate directories with the image caption of the reference picture and the accompanying Google query. We then store all samples of the reference picture in these directories. Conversely, we classified the patterns, colors, shapes, etc. based on their respective characteristics and organized images of the same color into one category, while images with similar pattern types were placed in another folder, and so on. At this stage, we generate directories and individually compare their attributes.

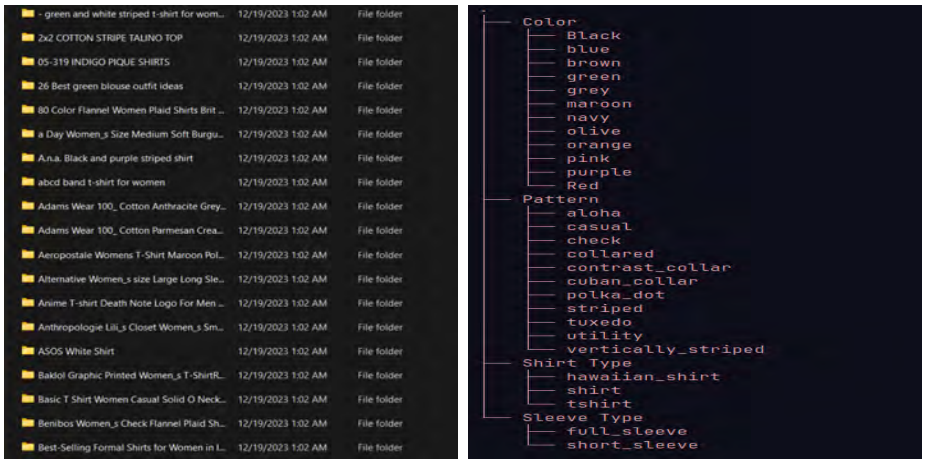

Figure 5.6: Folder Organization for Both Approaches

6. **Creating a Spreadsheet for Captions and Links:** We utilize the data gathered from our reference images to generate a spreadsheet. On the left side, we include the search prompts used to find the images and their corresponding captions. On the right side, we provide the links to the respective search results. We customize the links according to our requirements and intend to utilize them for enhancing our model's capability to comprehend the rationale behind each prompt through its created links.

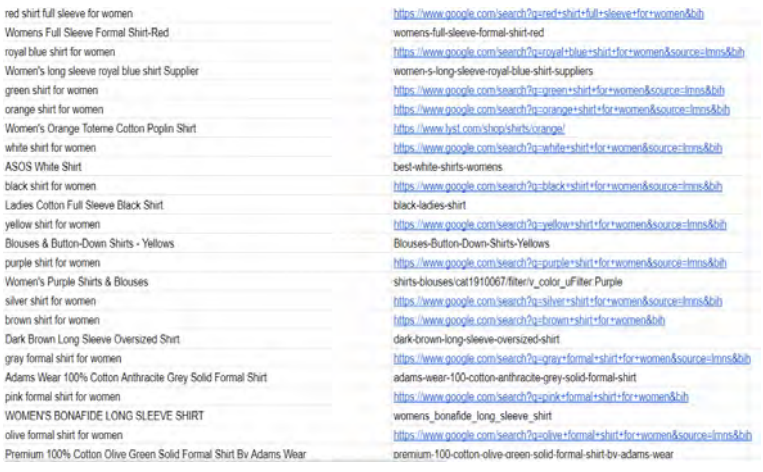

Figure 5.7: Spreadsheet Creation

5.2.2 Data Pre processing

In order to render our dataset appropriate for the research, it is necessary to adhere to certain protocols. These stages are crucial for both approaches as they must be compatible for further implementation.

Background Removal and cropping
Our dataset consists of images with a default white background, which was created using Microsoft Paint to design the doodle samples. For our initial approach, we employed YOLOv8x to assist the model in distinguishing between the clothing and the background. It eliminates the white background from all images. In the initial method, we utilized both cropped and uncropped data during our training process. The x1, x2, y1, and y2 coordinates were recorded in a text file. A Python script was employed to crop each image and store it in a new directory with the same original name. In contrast, the second strategy involved utilizing the mean and standard deviation values from the imagenet dataset. This allowed for the transformation of the data by cropping it to a size of 128 and normalised it by the help of imagenet status from the pretrained model.

Data Augmentation
Data augmentation is a technique that can be employed to enhance the quantity of data inside a given dataset. Data augmentation enhances the accuracy of machine learning and neural network models by generating substantial amounts of data, hence facilitating the training process. We have implemented many data augmentation strategies during our preprocessing phase. They are-

1. Rotating
2. Width Shifting
3. Height Shifting
4. Shearing
5. Flipping

We expanded our dataset using the ImageDataGenerator function from the tensorflow library, following a set of instructions. Initially, we randomly rotate our image by 20 degrees. Next, we establish a width shift range of 0.2, allowing for random horizontal shifting of images within this specified range. In addition, we establish vertical alignment by defining the height shift range as 0.2. Next, we establish our shearing range as 0.2 and our zoom range as 0.2. Finally, we enabled the horizontal flip by setting it to true. Upon completing the prescribed procedures, we have successfully executed the data augmentation technique, hence enhancing the precision of our models.

Resize
Within our dataset, there exists a range of data points that differ in size. Not all of the data shares the same dimension. To address this issue, we employed the RGB formatting. Initially, we convert the photos to the RGB color space and subsequently resize them to dimensions of 224 x 224. If the image is in array format,

it is divided by 255 in order to normalize it. We have to specify the shape as it is mandatory to indicate the shape when providing input. In the second approach, we utilized the mean and standard deviation values from the Imagenet dataset to normalize the form and shrink it to 128 × 128. Afterwards, we normalized it using the Imagenet standardization.

Rescale
Prior to feature extraction, the pixel values of the image have been rescaled to a range of 0 to 1 in order to standardize the data. Typically, the pixel values of photographs range from 0 to 255. The expanded range of pixels is unsuitable for deep learning due to its tendency to cause instability in the neural network and hinder the model's capacity to process such high values. Therefore, in order to achieve more optimal outcomes, it is necessary to rescale the pixels prior to feeding them into the model. Consequently, the pixels have undergone a division by 255 in order to adjust their scale.

Encoding
Once every image from the dataset is loaded, we proceed to generate arrays for both the features and labels. Additionally, we determine the overall count of distinct classes based on the labels. Next, we utilize the LabelBinarizer to convert the labels into a one-hot encoded format, allowing them to be compatible with our dataset for transformation.

Code Explanation
Firstly, The essential libraries are imported, encompassing Matplotlib for visualizing data, WordCloud for generating word clouds, Pandas for manipulating data, NumPy for doing numerical computations, and other auxiliary tools. The code utilizes the Pandas library to parse a CSV file that contains image captions. The programme manages character encoding and generates a DataFrame called dbset, which includes columns for image IDs, image names, and titles (captions). After that, The analysis of punctuation in captions leads to the definition of a function that eliminates punctuation and converts text to lowercase. Extraneous words and numerical values are also eliminated. Furthermore, the code examines the dataset for words consisting of one or two characters, while excluding stopwords. It generates a dictionary that pairs image names with the equivalent concise phrases found in captions. The code determines the amount of the vocabulary and tallies the occurrence of terms in captions. Additionally, it presents a visual representation of the 50 most commonly occurring terms, both at the highest and lowest frequencies. The dataset is divided into training, validation, and test sets. Records having captions that exceed a predefined word limit are eliminated. The processed data is stored as Parquet files to optimize storage efficiency. The Xception model is utilized to preprocess the images and extract features. The characteristics are stored as NumPy arrays. The process involves tokenizing image captions and generating sequences of input and output tokens. The process of tokenization is executed by utilizing the Tokenizer class. The tokenized data is stored as Parquet files for both the training and validation sets. And lastly, intermediate data structures and variables are deallocated to release memory resources.

5.3 Feature Extraction

Our research involved the utilization of several models and approaches to extract features from the image. Instead of re-training the weights, the pre-trained imagenet weights of these models were utilized for feature extraction. YOLOv8x was employed to eliminate the white background from the photos. Our collection consists of samples that have a default white backdrop, since they were made using Microsoft Paint. Consequently, we employed this model to eliminate the background and extract the primary dress image from the dataset. Subsequently, we trained it using our initial strategy. Xception was employed to extract the features of the images in our initial methodology. It will enhance our ability to extract the image with greater efficiency and speed. By undertaking this action, we will enhance its precision. The LabelBinarizer was employed to identify and encode all distinct classes into one-hot encoding, facilitating its utilization in our approach model. We also applied Blip-2, GPT-2, Vision Encoder-Decoder for our first approach as a model. Our second technique employed the ResNet18 model. This model utilizes an 18-layer structure, which significantly enhances our ability to do multilabel and multiclass classification tasks. We employed this approach to discern between numerous classes and various combinations of classes. We utilized the mean and standard deviation data from the Imagenet Dataset to standardize the augmented values in the second method. First, we randomly assigned the size and form of the features. Then, we adjusted them to match the standard values used in the imagenet dataset to ensure compatibility with our system.

5.4 Train-Test Split

Our research has utilized two distinct methodologies, each with its own unique train-test split for validation purposes. Our major strategy involved dividing the dataset into an 90:5:5 ratio, with 90% of the data utilized for training and 5% used for validation and 5% used for testing. Alternatively, our secondary method utilizes a dataset split of 85:15, with 85% of the data used for training and 15% reserved for validation testing.

Chapter 6

Implementation and Results Analysis

6.1 Model Implementation

1. **R-CNN with Xception:** Initially, we established the maximum length of the in-sequence, the size of the vocabulary, the number of training and validation records, and the batch size. Next, we imported the parquet files of the valid and train datasets that were generated as part of our data preprocessing. Following that, a class is created for the Hyperparametertuner to optimise the hyperparameters. Subsequently, the essential libraries are imported and the trial model is executed for a maximum of 10 trials and 25 epochs. The trial model includes essential parameters, such as dense units, image dropout, embedding units, embedding dropout, merging units, and learning rate, which are required for training our main model. Our model incorporates feature extraction from the Xception encoder and decoder, as well as dense and LSTM sequencing modelling. Additionally, we include additional output dense layer. In order to train the model, we provided the model with the extracted features of the images and the captions as input. We used Adam optimizers to optimise the model's performance, and the loss function was defined as Categorical CrossEntropy. Ultimately, we assembled the model and executed it for a total of 500 epochs.

2. **Blip-2:** Initially, we imported the requisite libraries and subsequently loaded the train, validate, and test datasets from their respective parquet files. Subsequently, we proceeded to carry out image processing, employing the pre-trained processor derived from the 'Salesforce/blip2-opt-2.7b'. Subsequently, we conducted image processing and developed a data loader to facilitate the loading of data into the model. Subsequently, we invoked the pre-trained model named "ybelkada/blip2-opt-2.7b-fp16-sharded". Prior to training, we utilised an accelerator and implemented LoralConfig to enhance the speed and efficiency of the training process. The learning rate employed was 6e-4. We employed the Adam optimizer. The model was trained for 10 epochs with a gradient_accumulation_steps value of 2. [35]

3. **GPT-2+Vision Encoder-Decoder:** At first, we import necessary libraries and modules for our GPT-2 Vision encoder and decoder. Our code implements the NLTK (Natural Language Tokenizer) for sentence splitting. After that, we configured our main pre-trained model which is the Vision Encoder and Decoder model. This model combines the vision encoder and GPT-2 for text decoder, which are crucial for understanding both textual and visual information. Then we initialized an image feature extractor (AutoFeatureExtractor) which loads the pre-trained weights for the Vision encoder model, and similarly, we used the AutoTokenizer module to load the pretrained weights for our GPT-2 model. Next, the model's configuration is adjusted to match with the tokenizer's parameters and saved the pretrained weights for both models. In the next stage of our code, we loaded our train, validation and tests datasets from the parquet files that we have created during data preprocessing. The data is then manipulated by removing unnecessary columns and mapping the image paths to a new type format. Next, we performed the text and image preprocessing. For our captions, a tokenization function is built to tokenize the words and an image feature extraction function is built and used to extract the image features, and finally the resulting dataset is prepared for training. After that, we set the arguments for our Sequence to sequence model, setting the total training epochs to seven, and the training and evaluate batch size to 4. Finally, we set our evaluate metric to Rouge metrics and trained our model.[26]

4. **ResNet50+LSTM (n-Layers):** This model comprises 50 residual blocks to acquire and map all the trainable parameters of the doodle artwork to their respective labels. In this model, we employed a pre-trained model and integrated it into our project by removing the output layer and modifying it to align with our specific work aim. The captions were kept in a dictionary and then tokenized using a tokenizer. Subsequently, a generator function was employed, which accepted the image and caption as input and generated a generator to be utilised for training. The LSTM model employed the softmax activation function in the output layer, utilising categorical cross entropy as the loss function and RMSprop as the optimizer. We executed the model for a total of 200 epochs and incorporated early stopping to guarantee that the model ceases training if there is no improvement in accuracy. [28]

5. **Multi Label Classification:** This method utilises a dataset composed of doodle drawings clustered into various folders based on distinct features: colour, dress type, pattern, and sleeve type. The classifier detects all the color classes the prompted doodle image belongs to. This classifier was trained on the colour dataset, consisting of 16 distinct colour classes. Before training, the labels of these doodle drawings were encoded in a manner where the presence of a color class is encoded to be 1, while all other colours are represented as 0. A class called MyDataset was utilised for loading and pre-processing the dataset. It also performs transformations such as converting the images into tensors and augmentations such as resizing to 128x128, cropping, horizontal flipping, and rotation. The doodle tensors were also normalized using the pre-trained weights and standard deviation of the ImageNet dataset. The entire dataset was split into training and validation sets in the ratio of 0.85

and 0.15 respectively. In order to fit and train the model on the dataset, we had to create DataLoaders with random shuffling and send it to 'cuda' using the DeviceDataLoader. The multi-label classification was performed using the ResNet18 architecture consisting of 18 residual blocks and using binary cross-entropy as the loss function for both training and validation stages. The model was run for 10 epochs with a max learning rate of 0.001, gradient clip of 0.1, weight decay of 1e-4 and the Adam optimizer. This classifier was able to learn the distinguishing features of the color, dress type, pattern, and sleeve type to predict out all the labels the prompted image belonged to. The predict_all function consolidates the predictions for colour, dress type, pattern, and sleeve type, resulting in a conclusive caption. The outputted caption was then fed as a parameter to Google Custom Search API in order to obtain the e-commerce site links of associated doodle and the classified caption.

6.2 Performance Evaluation

The models have been assessed using four performance metrics: accuracy, precision, recall, and f1 score.

1. **Accuracy:** Accuracy is a metric that quantifies the level of correctness in classification. It computes the proportion of accurately anticipated cases out of the total instances.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{6.1}$$

2. **Precision:** Precision is the quotient obtained by dividing the number of accurately anticipated positive observations by the total number of predicted positives. Its primary emphasis lies on the precision of the affirmative forecasts. Accuracy is crucial when the consequences of incorrect positive results are significant, and it aids in determining the percentage of accurately anticipated positive cases.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{6.2}$$

3. **Recall:** Recall is the proportion of accurately predicted positive observations out of all the observations in the actual class. The metric assesses the model's capacity to accurately represent and encompass all pertinent examples. Recall is essential in situations where the consequences of missing important instances are significant, as it guarantees that the model detects as many relevant examples as it can.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{6.3}$$

4. **F1 Score:** The F1 score is calculated as the reciprocal of the arithmetic mean of the reciprocals of precision and recall. It offers a trade-off between precision and recall, particularly in datasets with imbalanced classes. The F1 score is a numerical measure that falls within the range of 0 to 1. A higher value signifies a more optimal trade-off between precision and recall. It is especially beneficial in cases where there is an imbalanced distribution of classes.

$$F1 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{6.4}$$

   Here,
   TP = True Positive, FP = False Positive
   TN = True Negative, FN = False negative
   Where true means correctly predicted and false means incorrectly predicted and positive means positive instances and negative means negative instances.

Confusion Matrix: A confusion matrix is a concise representation of the outcome of predictions made on a classification issue. The count numbers summarize the number of accurate and inaccurate predictions, and they are further categorized by each individual class. Here is the essential component of the confusion matrix. The confusion matrix illustrates the instances in which your classification model encounters confusion while making predictions. It provides you with a deeper understanding of the mistakes made by your classifier, namely the specific categories of errors made. This breakdown effectively surpasses the constraint of relying just on classification accuracy.

		Predicted	
		Positive	**Negative**
Actual	**Positive**	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

Table 6.1: Confusion Matrix

6.3 Results Analysis

We conducted our research following two different approaches where in the first approach we implement 7 models and in our second approach we implement four identical models for our individual features such as pattern, dress type, color, sleeve type. Where all the feature classification followed ResNet18 Model structure with multi-class classifier and multi-label classifier. Here is a result analysis for each of the models of both approaches.

- **First Approach:**
  - RCNN with Xception
  - Blip-2
  - GPT-2 and Vision Encoder and Decoder
  - ResNet50+LSTM (n-Layers)
- **Multiple Feature Classification Approach:**
  - Multi label classification.

6.3.1 First Approach

In our first approach we intiated 7 different models for our research. Some of them have the same models with increasing number of layers such as ResNet+LSTM Model, CNN+LSTM Model. We used different layers to examine their loss and accuracy and find out which model works better in this approach.

1. **R-CNN with Xception:** Here is the loss and accuracy graph for sleeve type feature. We can see that the validation loss in decreasing and the validation accuracy is increasing and that is upto 70%

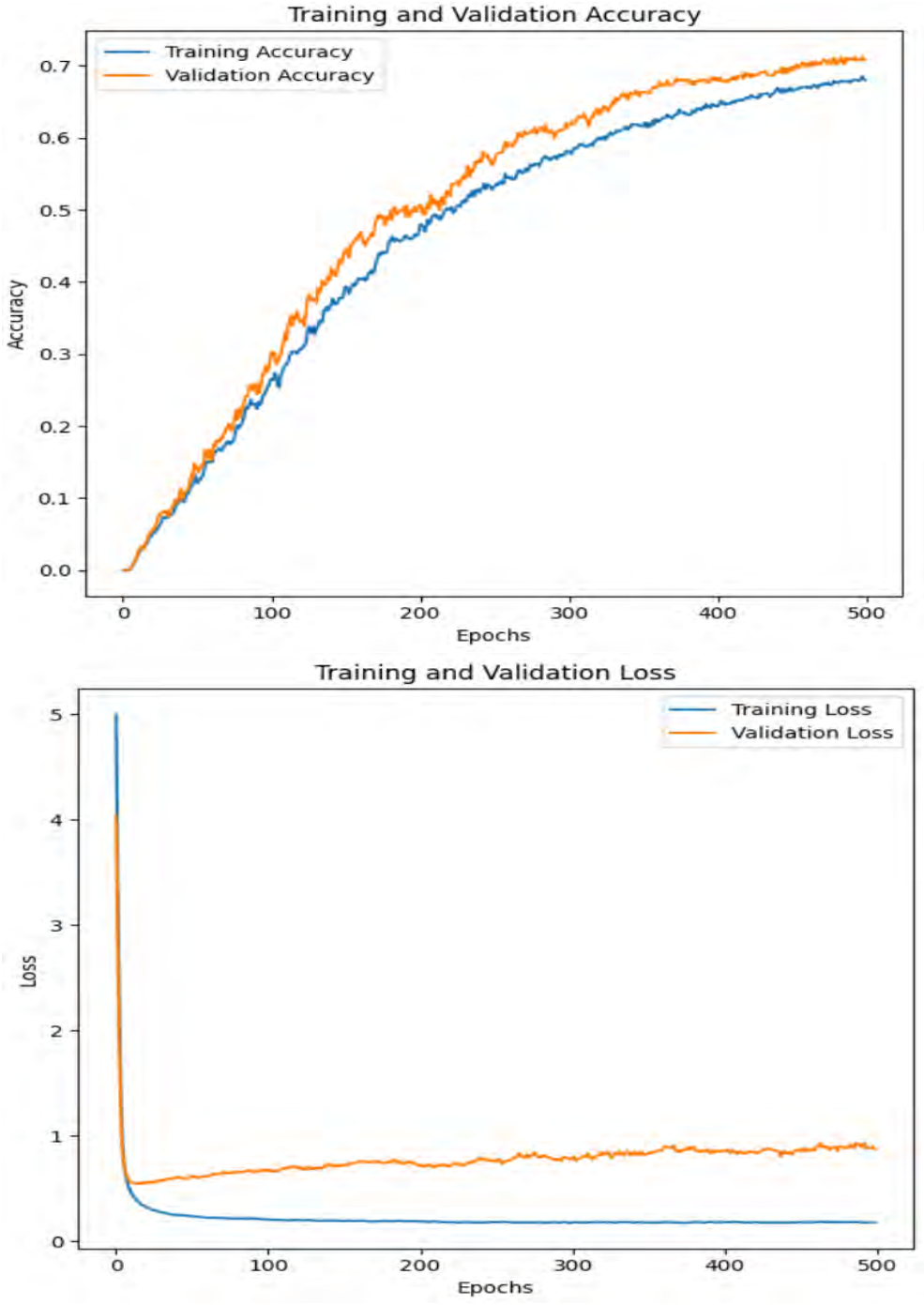

Figure 6.1: Loss and Accuracy Graph for R-CNN Model

2. **Blip-2:** Here is the loss and accuracy graph for sleeve type feature. We can see that the validation loss in decreasing after 1 epoch and 10 epochs.

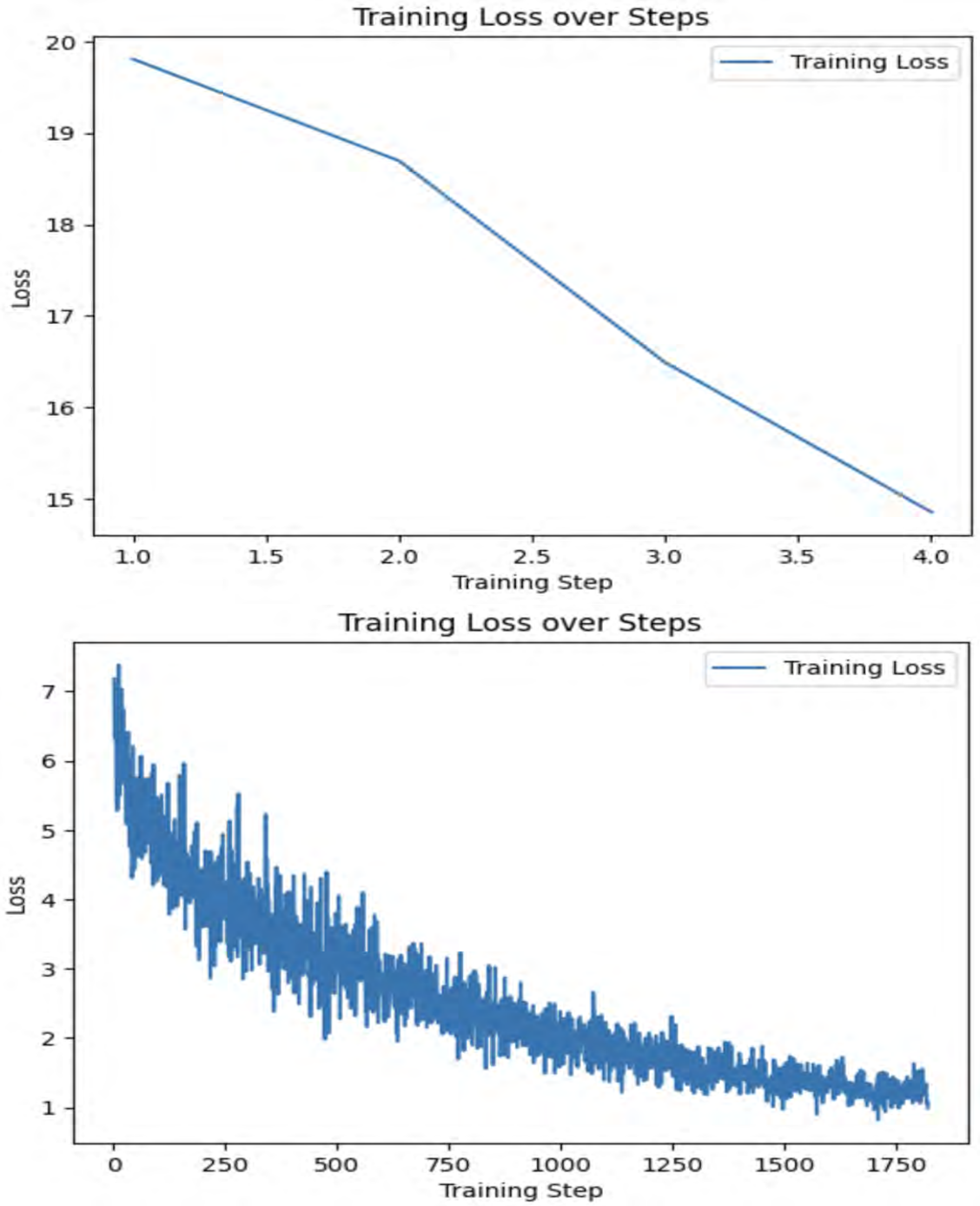

Figure 6.2: Loss Graph for Blip-2 Model after 1 & 10 Epoch

Here is the information of Blip-2 Model After 1 and 10 epoch. These values works as a performance metrics of the model.

	rouge1_fmeasure	rouge2_fmeasure	rougeL_fmeasure	bleu@1	bleu@2
Epoch 0:	0.03	0.0	0.03	0.03	0.0

	rouge1_fmeasure	rouge2_fmeasure	rougeL_fmeasure	bleu@1	bleu@2
Epoch 0:	0.40	0.11	0.37	0.38	0.20
Epoch 1:	0.41	0.13	0.38	0.40	0.22
Epoch 2:	0.43	0.15	0.40	0.40	0.24
Epoch 3:	0.44	0.16	0.41	0.41	0.24
Epoch 4:	0.46	0.19	0.43	0.41	0.26
Epoch 5:	0.46	0.20	0.44	0.42	0.27
Epoch 6:	0.48	0.22	0.46	0.44	0.29
Epoch 7:	0.48	0.22	0.45	0.44	0.29
Epoch 8:	0.50	0.24	0.47	0.46	0.32
Epoch 9:	0.50	0.25	0.48	0.47	0.33

Figure 6.3: Information for Blip-2 Model after 1 & 10 Epoch

3. **GPT-2 and Vision Encoder-Decoder:** Here is information for GPT-2 and Vision Encoder-Decoder Model. These values works as a performance metrics of the model. We can see that the validation loss in decreasing.

Epoch	Training Loss	Validation Loss	Rouge1	Rouge2	Rougel	Rougelsum	Gen Len
1	0.045800	0.049527	43.725800	16.158200	40.756300	40.753900	6.312958
2	0.039500	0.031512	52.158500	26.414300	48.773900	48.819800	6.701711
3	0.026900	0.024188	58.302600	32.885700	54.792800	54.855600	6.986145
4	0.020900	0.020150	61.484600	35.974600	57.748400	57.825400	7.165444
5	0.016100	0.017158	62.429000	36.744400	58.146800	58.152900	7.306438
6	0.012500	0.016251	59.622500	31.157600	54.915900	54.902900	7.568052
7	0.010500	0.016115	56.252300	24.871400	51.085600	51.048100	7.510187

Figure 6.4: Information for GPT-2+Vision Encoder-Decoder Model after 7 Epoch

4. **ResNet50+LSTM (n-Layers):** Here is the loss and accuracy graph before and after adding more dense layer to the resnet+lstm model. We can see that the validation loss in decreasing and the validation accuracy brfore adding more layers is increasing and that is upto 92%. This is the best models so far among all the models in the first approach and after adding layers it became 91%.

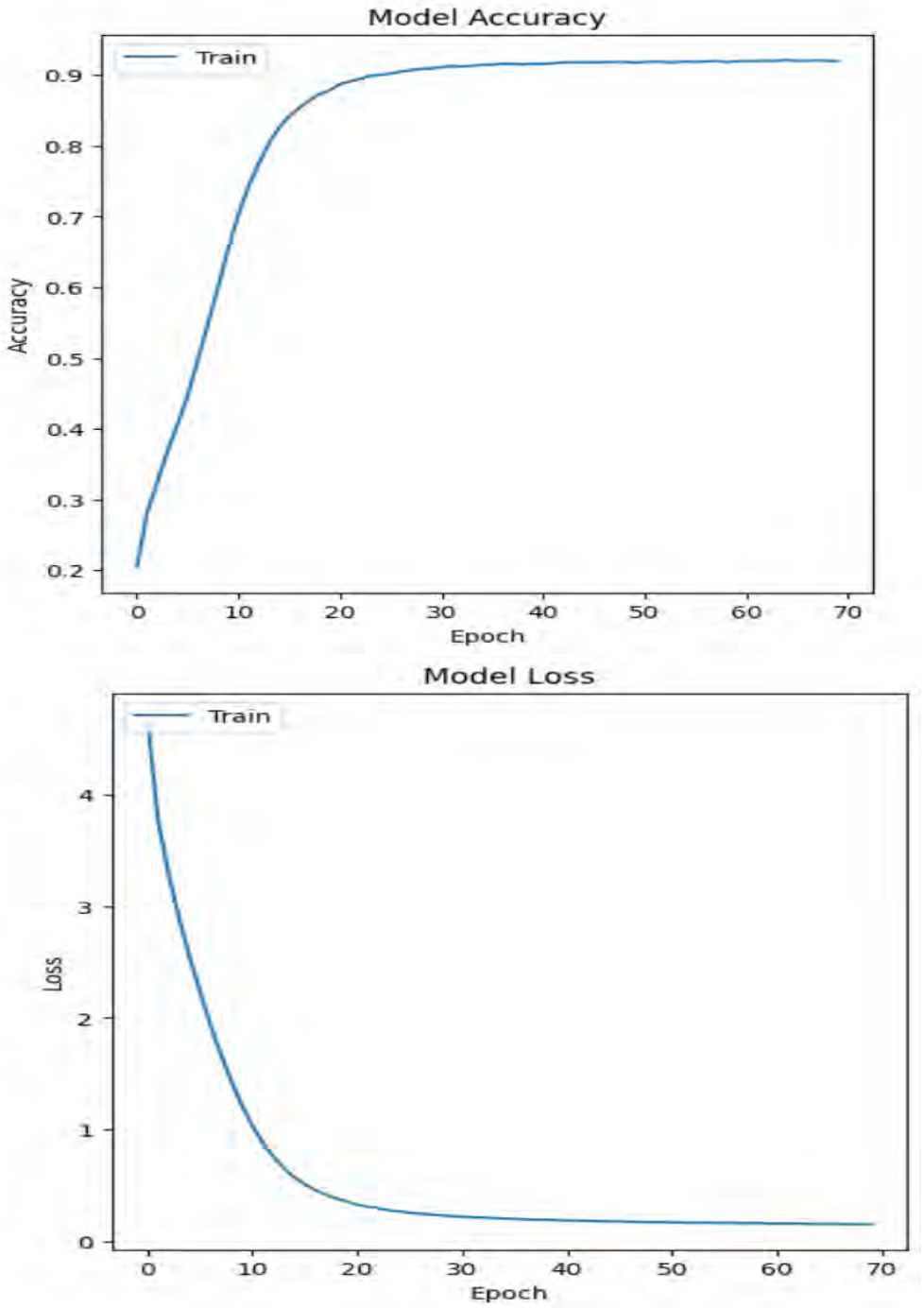

Figure 6.5: Loss and Accuracy Graph for ResNet50+LSTM Model

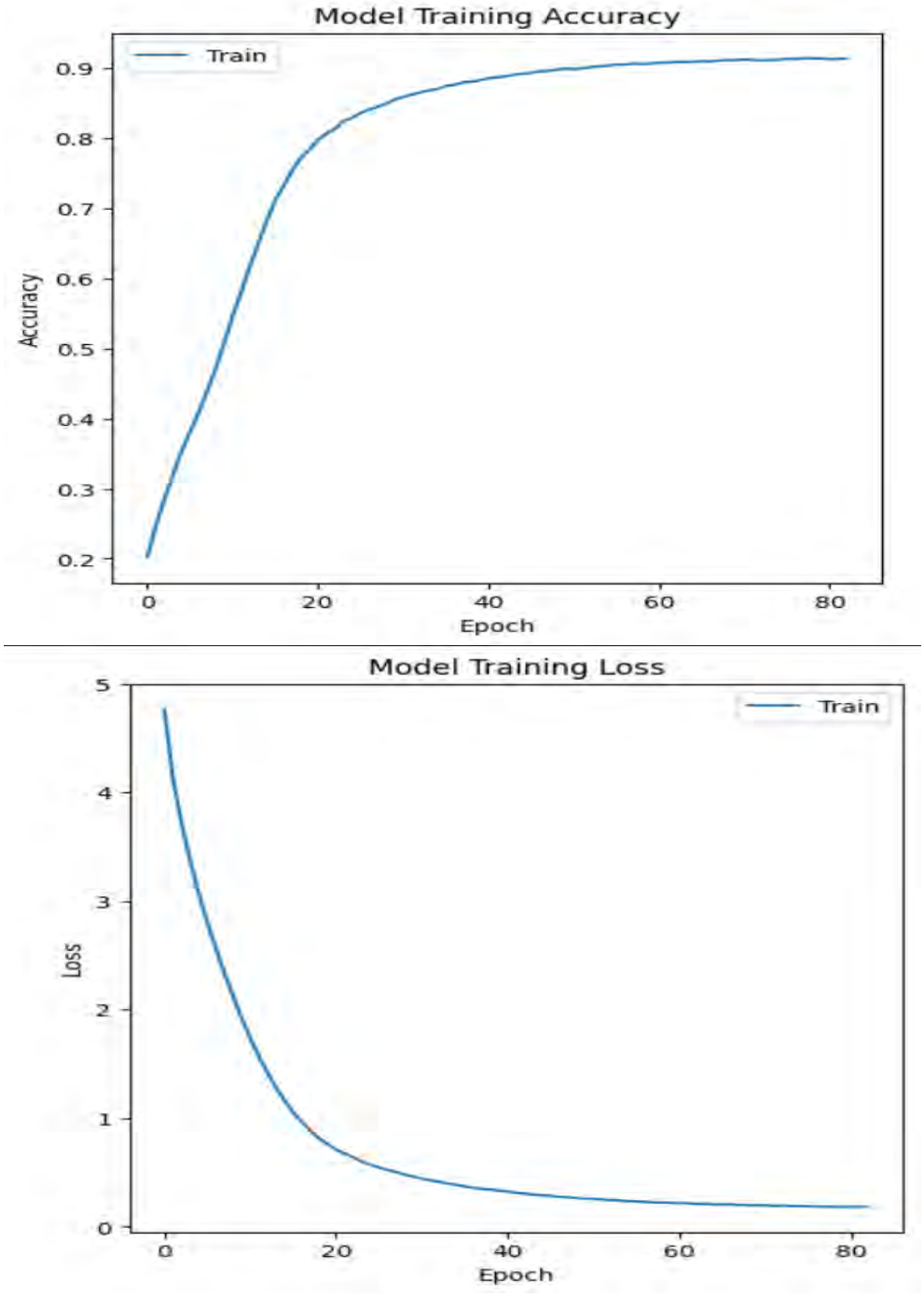

Figure 6.6: Loss and Accuracy Graph for ResNet50+LSTM Model with more Layers

6.3.2 Second Approach

This is the final accuracy and loss graph for each features in our Multiple features Classification Approach. The features includes sleeve type, pattern, dress type and color.

1. **Sleeve Type:** Here is the loss and accuracy graph for sleeve type feature. We can see that the validation loss in decreasing and the validation accuracy is increasing and that is upto 95%

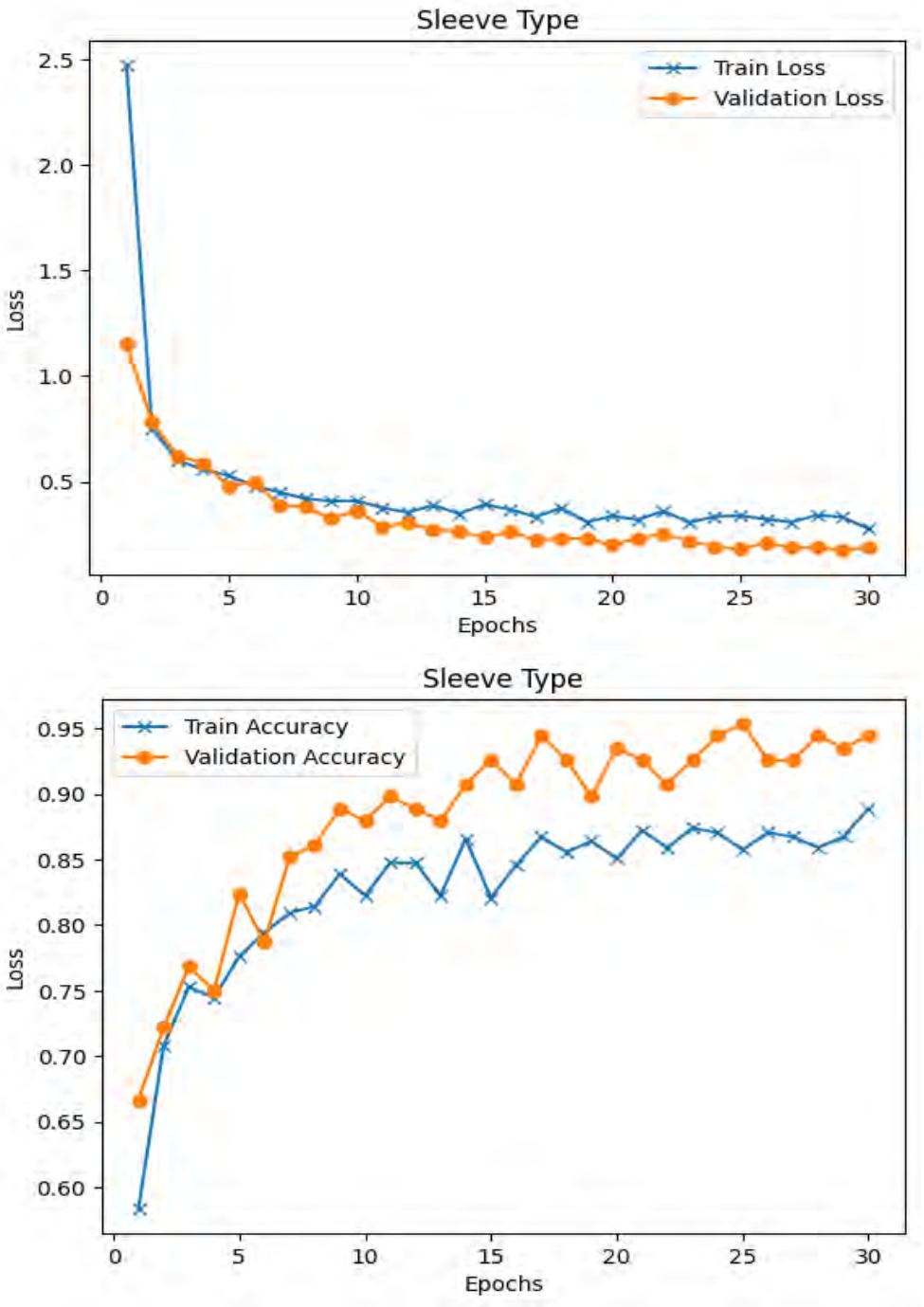

Figure 6.7: Loss and Accuracy Graph for Sleeve Type

2. **Pattern:** Here is the loss and accuracy graph for pattern feature. We can see that the validation loss in decreasing and the validation accuracy is increasing and that is upto 90%.

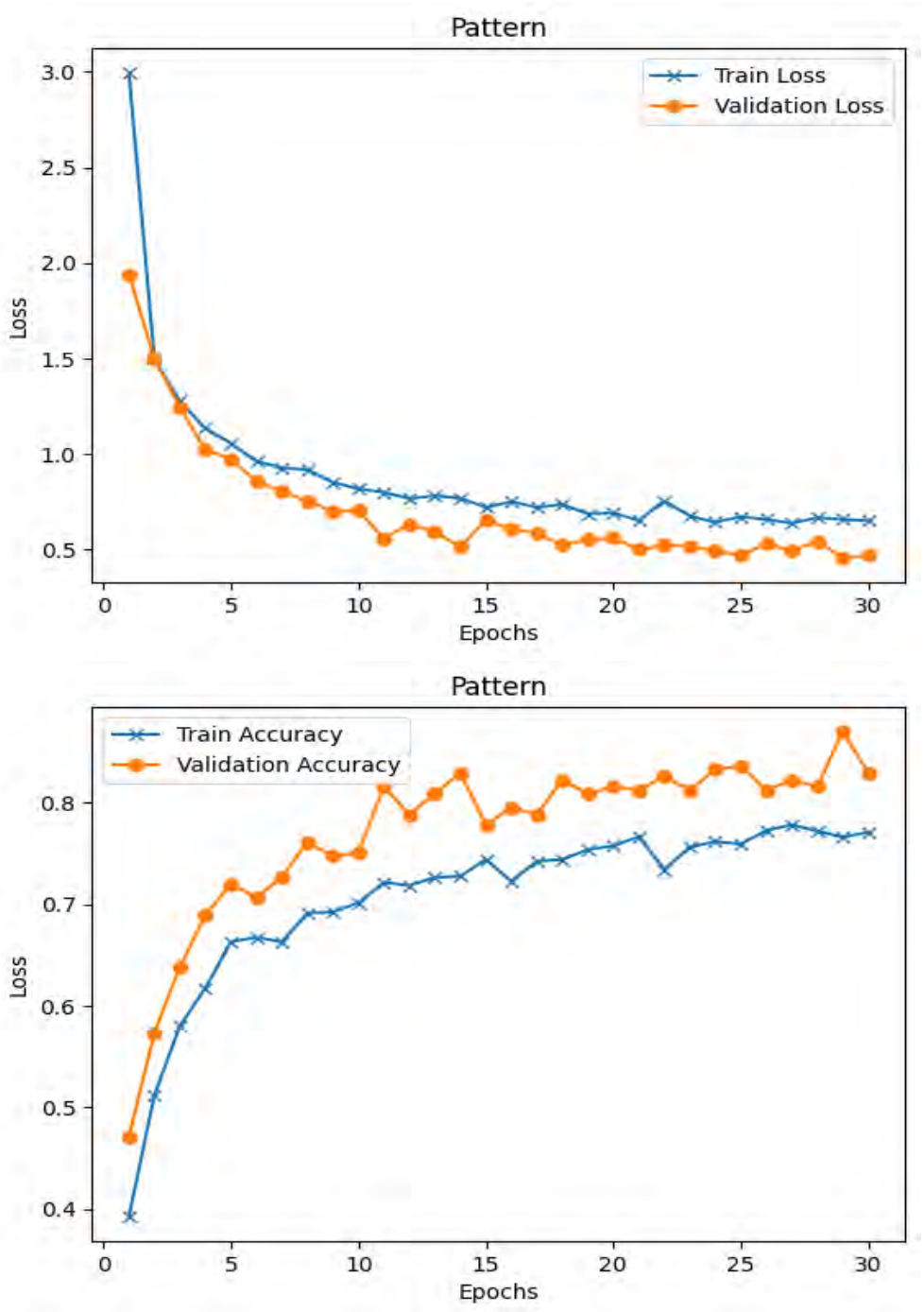

Figure 6.8: Loss and Accuracy Graph for Pattern

3. **Dress Type:** Here is the loss and accuracy graph for dress type feature. We can see that the validation loss in decreasing and the validation accuracy is increasing and that is upto 80%.

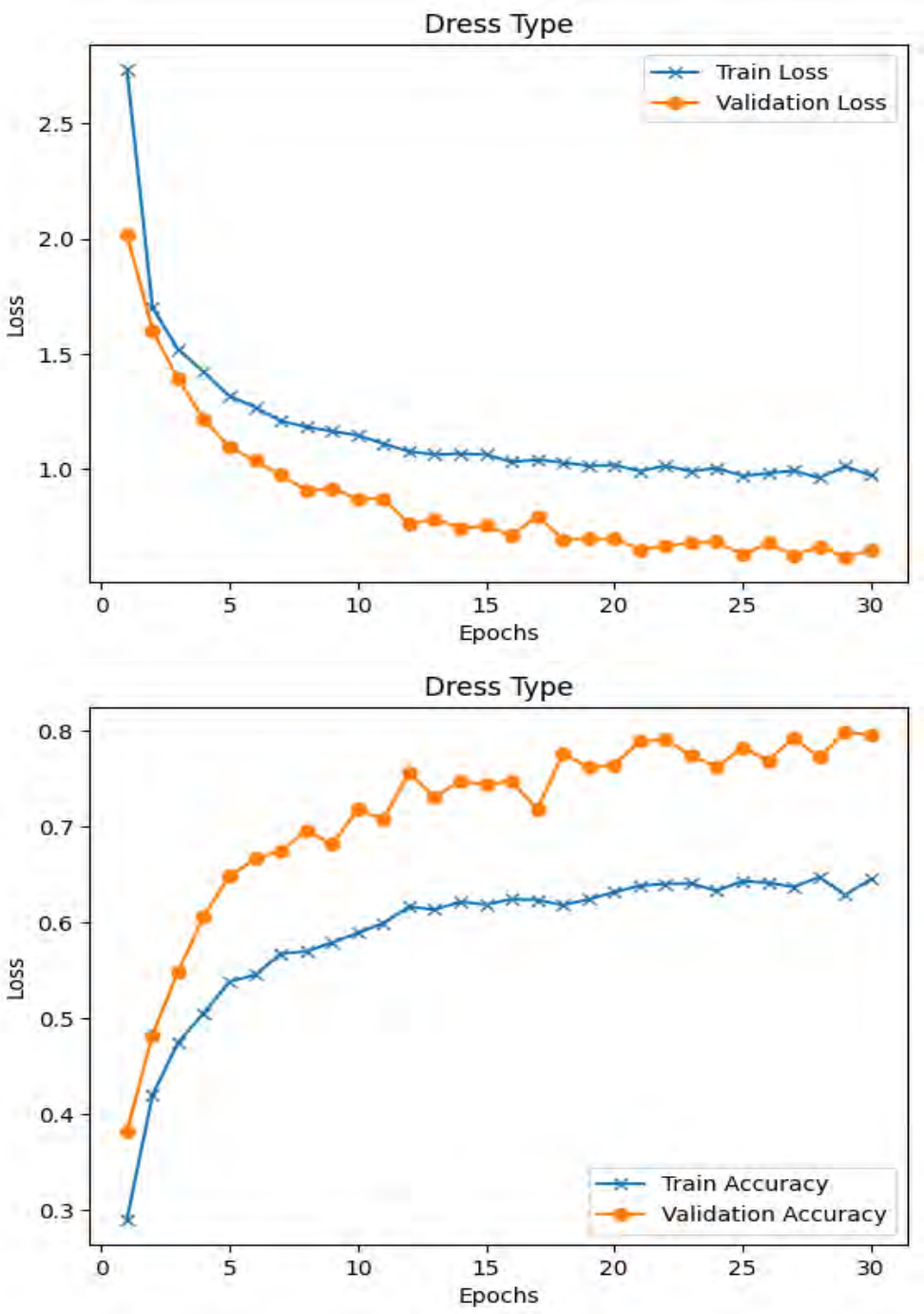

Figure 6.9: Loss and Accuracy Graph for Dress Type

4. **Color:** Here is the loss graph for color feature. The reason why we couldn't get the accuracy graph is because we directly get the F1 from the color model history. Because we used Multi-label classifier on color. We can see that the validation loss in decreasing.

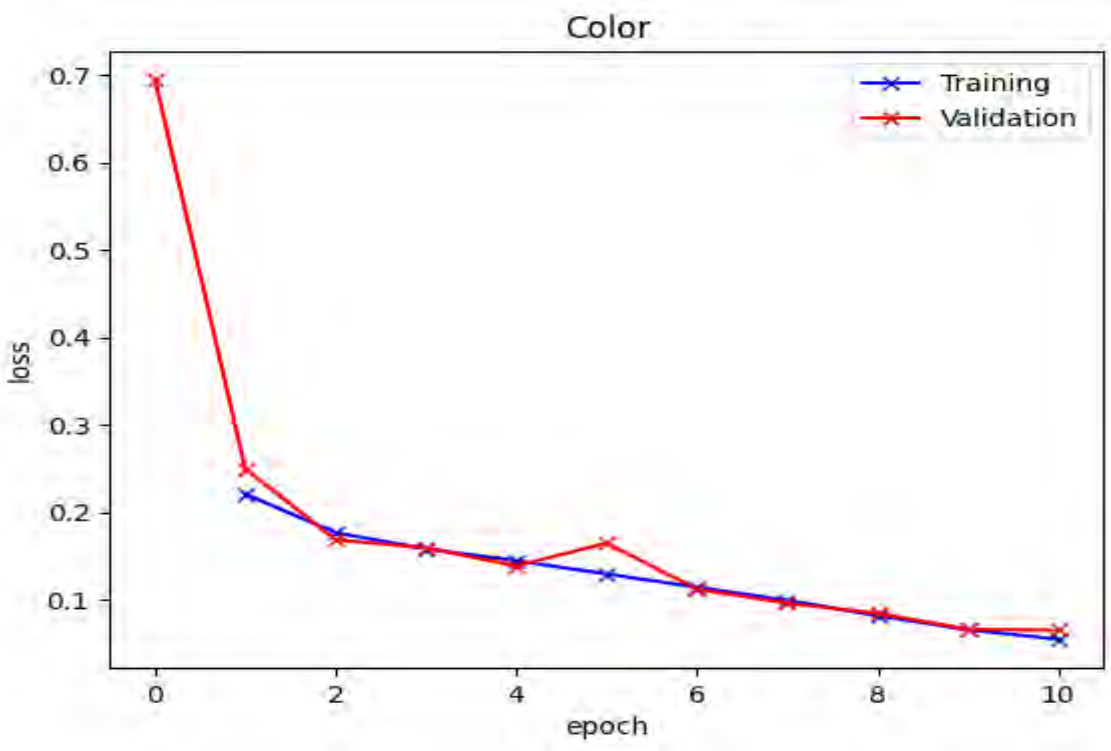

Figure 6.10: Loss and Accuracy Graph for Color

6.3.3 Comparison on the performance metrics for Second Approach

Here is a table and the Bar Graph of these performance matrices for each of the features. You can compare the values with each other and find that sleeve type has very high accuracy as there were only two class for sleeve which is half sleeve and full sleeve. Where other features have many classes. For example, Color has total 17 unique classes.

Table 6.2: Performance Metrics for Different Classes

Classifier	Loss	Accuracy	Precision	Recall	F1 Score
Sleeve Type	0.1881	0.9444	0.9444	0.9444	0.9444
Pattern	0.4703	0.8294	0.8331	0.8294	0.8246
Dress Type	0.6471	0.7950	0.8010	0.7950	0.7947
Color	0.0661	0.7593	0.9006	0.8117	0.8538

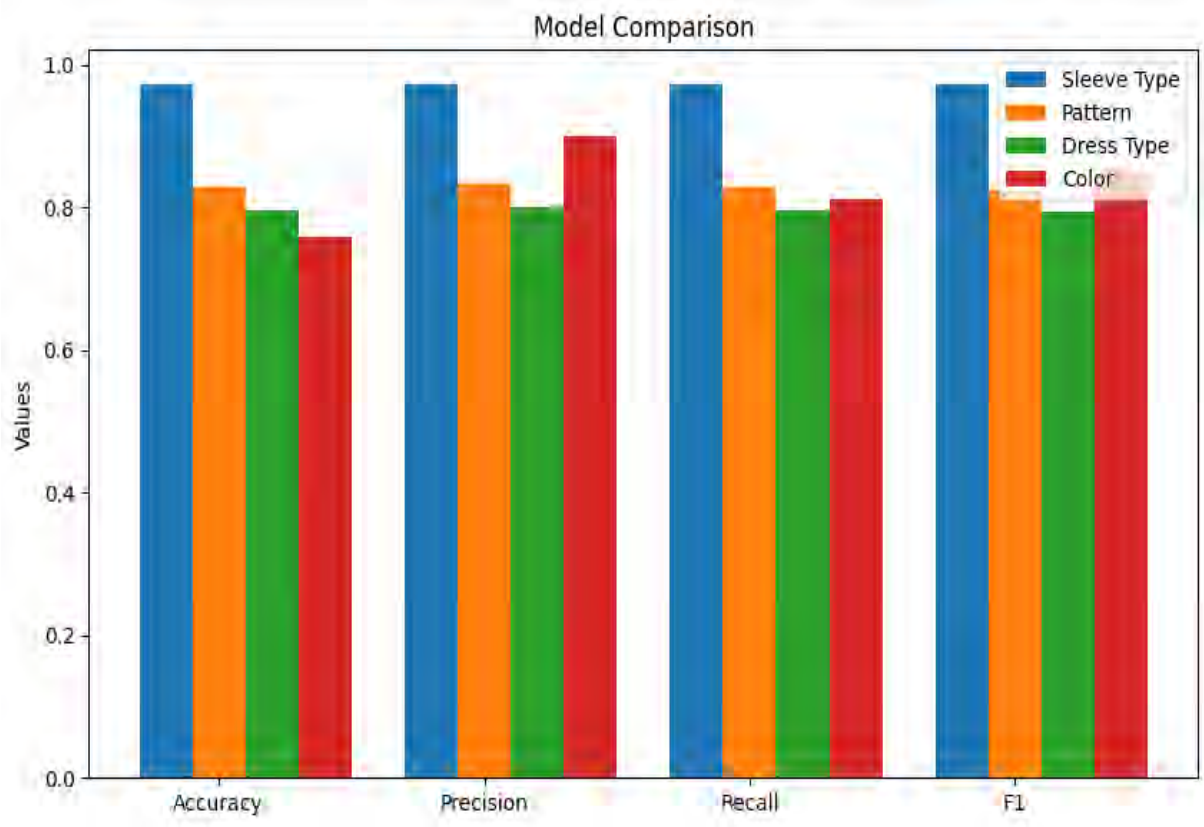

Figure 6.11: Multiple Feature Classification Model Performance Metrics Bar Graph

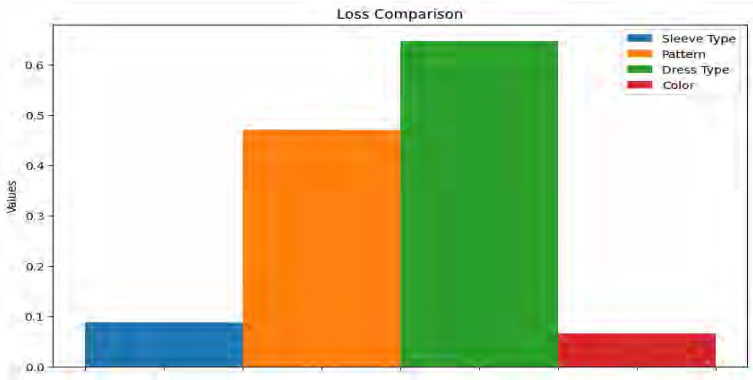

Figure 6.12: Loss comparison for the four features

We can see from the loss comparison that the sleeve type has low loss as that feature has only two distinct class. Where the dress type has the maximum loss because dress has many combination which will reduce from time to time. The color on the other hand, class being used in a multilbael resulted remarkable output and give us the minimum loss from all of these above.

6.3.4 Multiple Feature Classification Model's Overall Performance Analysis

Here is the graph for the overall performance of the four feature's models. We can see the overall score is quite good for such a novelty research like this where we made all the datasets from scratch.

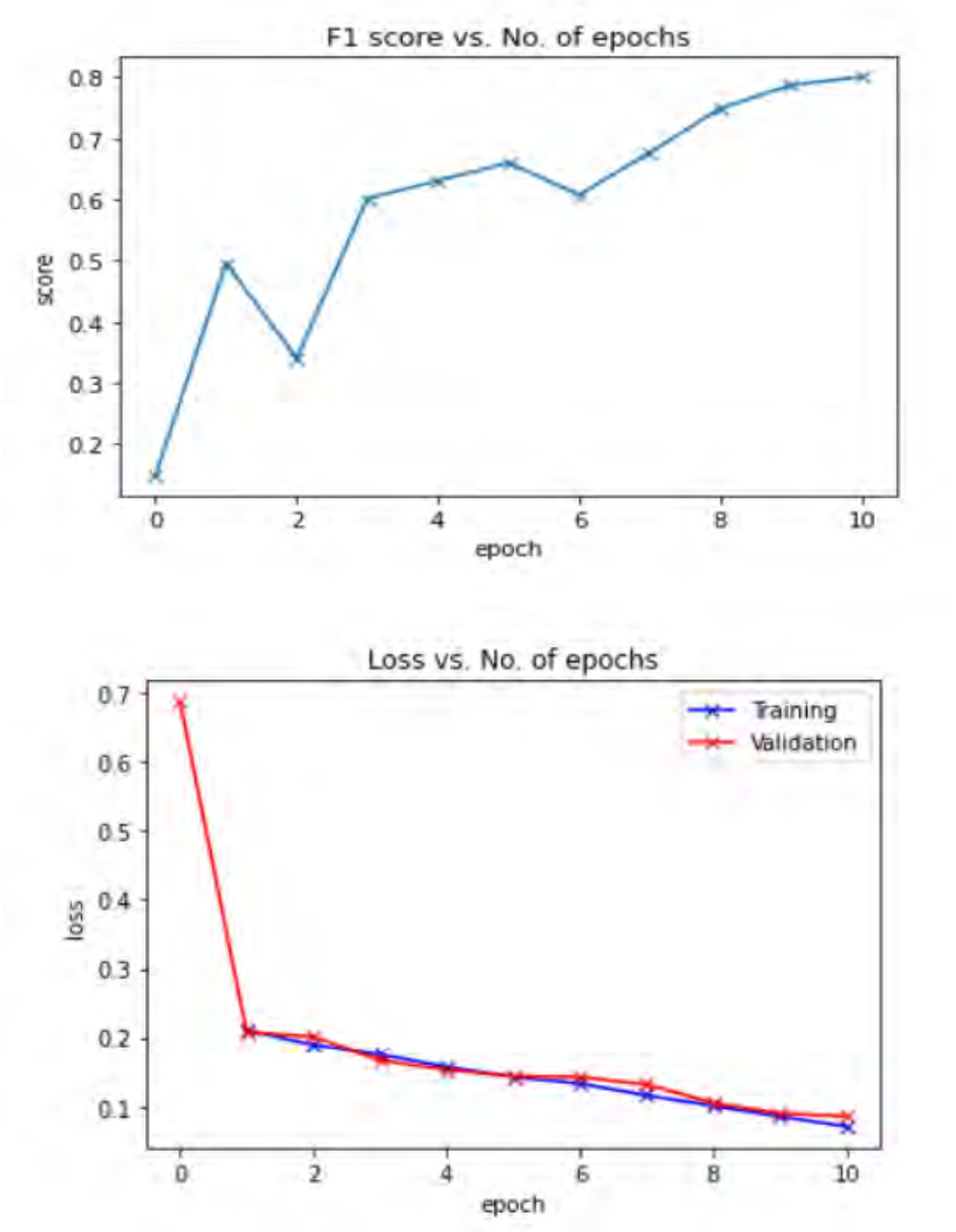

Figure 6.13: Multiple Features Classification Model Accuracy and Loss Graph

Chapter 7

Conclusion

7.1 Conclusion

The purpose of this work is to examine the various challenges and approaches that are related with achieving the desired results, namely those that are associated with generating drawings. We carried out a comprehensive examination, during which we examined the data obtained from free-hand doodling, and were able to clarify the various applications that were made possible by these doodles. We have devised a novel method, although a simple one, that is capable of recognising the boundaries of objects in sketches, classifying them, recognising complex patterns or designs, and generating a search query for a system that is connected to the internet.

We regard this survey as a stimulus for new researchers, offering vital perspectives for incorporating novel traits, allowing them to stay up to speed with the current condition of the field, and stimulating future progress in this interesting topic. This is based on the fact that this survey was conducted. As a result of the vast broadness of our discoveries, there is the possibility that the complexity consumers face while looking for solutions that precisely correspond to their requirements can be simplified. We have high hopes that the applications that are connected to this research will result in achievements that are truly amazing.

In conclusion, the work that we have done during the entirety of this project has not only illustrated the complexity of sketch-related difficulties and techniques of representation, but it has also paved the way for breakthroughs in both academic study and practical application throughout the entirety of this project. As a result of the survey, the foundation for future research and technological achievements in this fascinating topic will be strengthened, and it is predicted that the survey will have far-reaching effects in both the academic and practical fields.

7.2 Limitations and Future Work

Despite employing divergent methodologies, our models may not be universally applicable to all cases. Firstly, in the multitask classifier model, if we provide a completely new feature as input without pre-training it, the machine may fail to recognize it and perhaps crash.

Moreover, in regard to the alternative strategy, we can expect a result from the given input as this approach is not limited to specific features. Nevertheless, the accuracy of the output will not be absolute due to slight variations in the input compared to another. it will gradually accumulate information and ultimately yield a possibly comparable result. This model is adaptive, allowing it to acquire knowledge from new inputs.

Furthermore, a notable constraint of this study is the exclusive focus on a single category which is clothing. At this moment, it will not acknowledge or identify any other products that fall into different categories. Currently, it will exclusively be appropriate for clothing products.

Finally, to address this constraint, we will endeavor to adapt this approach to accommodate more categories, such as footwear, watches, eyewear, and so on. This will enhance our research and fields of work. Hence, expanding its applicability beyond clothing to encompass other products is one of our intended future endeavors.

Bibliography

[1] C. M. Eastman and B. J. Jansen, "Coverage, relevance, and ranking: The impact of query operators on web search engine results," *ACM Trans. Inf. Syst.*, vol. 21, pp. 383–411, 2003.

[2] M. P. Evans, "Analysing google rankings through search engine optimization data," *Internet Res.*, vol. 17, pp. 21–37, 2007.

[3] N. Duhan, A. K. Sharma, and K. K. Bhatia, "Page ranking algorithms: A survey," *2009 IEEE International Advance Computing Conference*, pp. 1530–1537, 2009.

[4] J. Krapac, M. Allan, J. J. Verbeek, and F. Jurie, "Improving web image search results using query-relative classifiers," *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1094–1101, 2010.

[5] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and tell: A neural image caption generator," *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3156–3164, 2014.

[6] X. Chen and C. L. Zitnick, "Mind's eye: A recurrent visual representation for image caption generation," *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2422–2431, 2015.

[7] S. Xie and Z. Tu, "Holistically-nested edge detection," *International Journal of Computer Vision*, vol. 125, pp. 3–18, 2015.

[8] K. Xu, J. Ba, R. Kiros, *et al.*, "Show, attend and tell: Neural image caption generation with visual attention," in *International Conference on Machine Learning*, 2015.

[9] Y. Liu, M.-M. Cheng, X. Hu, K. Wang, and X. Bai, "Richer convolutional features for edge detection," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5872–5881, 2016.

[10] C. Gan, Z. Gan, X. He, J. Gao, and L. Deng, "Stylenet: Generating attractive visual captions with styles," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 955–964, 2017.

[11] Kmader. "Mobilenet classification." Kaggle project. (2017), [Online]. Available: https://www.kaggle.com/code/kmader/mobilenet-classification.

[12] Z. Yu, C. Feng, M.-Y. Liu, and S. Ramalingam, "Casenet: Deep category-aware semantic edge detection," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1761–1770, 2017.

[13] D. Guinness, E. Cutrell, and M. R. Morris, "Caption crawler: Enabling reusable alternative text descriptions using reverse image search," *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, 2018.

[14] K. Guo, J. WoMa, and E. Xu, *Quick, draw! doodle recognition*, 2018.

[15] S. Jadon, "Introduction to different activation functions for deep learning," *Medium, Augmenting Humanity*, vol. 16, 2018.

[16] J. A. Lasserre, C. Bracher, and R. Vollgraf, "Street2fashion2shop: Enabling visual search in fashion e-commerce using studio images," in *ICPRAM*, 2018.

[17] J. A. Lasserre, K. Rasch, and R. Vollgraf, "Studio2shop: From studio photo shoots to fashion articles," *ArXiv*, vol. abs/1807.00556, 2018.

[18] M. Fabien. "Xception model and depthwise separable convolutions." Website. (Mar. 2019), [Online]. Available: https://maelfabien.github.io/deeplearning/xception/.

[19] C. Ziakis, M. Vlachopoulou, T. Kyrkoudis, and M. Karagkiozidou, "Important factors for improving google search rank," *Future Internet*, vol. 11, p. 32, 2019.

[20] Y. Bitirim, S. Bitirim, D. Ç. Ertugrul, and O. Toygar, "An evaluation of reverse image search performance of google," *2020 IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC)*, pp. 1368–1372, 2020.

[21] S. Chen, Q. Jin, P. Wang, and Q. Wu, "Say as you wish: Fine-grained control of image caption generation with abstract scene graphs," *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9959–9968, 2020.

[22] K. F. Mawoneke, X. Luo, Y. Shi, and K. Kita, "Reverse image search for the fashion industry using convolutional neural networks," *2020 IEEE 5th International Conference on Signal and Image Processing (ICSIP)*, pp. 483–489, 2020.

[23] A. K. Bhunia, S. Khan, H. Cholakkal, *et al.*, "Doodleformer: Creative sketch drawing with transformers," in *European Conference on Computer Vision*, 2021.

[24] A. Sain, A. K. Bhunia, Y. Yang, T. Xiang, and Y.-Z. Song, "Stylemeup: Towards style-agnostic sketch-based image retrieval," *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8500–8509, 2021.

[25] O. Calzone, "An intuitive explanation of lstm," *Medium*, Feb. 2022. [Online]. Available: https://example.com/link-to-the-article.

[26] A. Kumar. "The illustrated image captioning using transformers." Cited as: The Illustrated Image Captioning using transformers. (2022), [Online]. Available: https://ankur3107.github.io/blogs/the-illustrated-image-captioning-using-transformers/.

[27] S. Mukherjee. "The annotated resnet-50." Towards Data Science, Medium. (Aug. 2022), [Online]. Available: https://towardsdatascience.com/the-annotated-resnet-50-a6c536034758.

[28] SayamAlt, *Image caption generation using resnet and lstms*, https://github.com/SayamAlt/Image-Caption-Generation-using-ResNet-and-LSTMs, 2022.

[29] M. U. Hassan, "Resnet (34, 50, 101): Residual cnns for image classification tasks," *Neurohive / Neural Networks*, Jul. 2023, https://neurohive.io/en/popular-networks/resnet/.

[30] S. Iskandaryan. "Fine-tune t5 for news article summarization." Kaggle project. (2023), [Online]. Available: https://www.kaggle.com/code/sargisiskandaryan/fine-tune-t5-for-news-article-summarization?fbclid=IwAR2IEoCTJ71epYeazl-XYnCj0lsgDk_j5wX3OE-kFz0Y8ER4aZjqIV6y6NU.

[31] J. Lu, S. Lee, I. Kim, W. Kim, and M. S. Lee, "Small foreign object detection in automated sugar dispensing processes based on lightweight deep learning networks," *Electronics*, vol. 12, no. 22, p. 4621, Nov. 2023. DOI: 10.3390/electronics12224621.

[32] F. Oyerinde, "Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models," *Personal Blog*, Jun. 2023, blog_url.

[33] G. Research. "Mammut: A simple vision-encoder text-decoder architecture for multimodal tasks." Blog Post. (May 2023), [Online]. Available: https://blog.research.google/2023/05/mammut-simple-vision-encoder-text.html.

[34] N. Shahriar, "Convolutional neural network—cnn (deep learning)," *Medium*, Feb. 2023. [Online]. Available: insert_url_here.

[35] S. Upadhyaya. "Automate fashion image captioning using blip-2." GitHub repository. (2024), [Online]. Available: https://github.com/SmithaUpadhyaya/fashion_image_caption.

[36] J. Alammar. "The illustrated gpt-2 (visualizing transformer language models)." Website. (), [Online]. Available: https://jalammar.github.io/illustrated-gpt2/.

[37] "Architecture diagram of resnet-18." ResearchGate. (), [Online]. Available: https://www.researchgate.net/figure/Architecture-Diagram-of-ResNet-18-21_fig2_353655307.

[38] "Gpt-2 model architecture." ResearchGate. (), [Online]. Available: https://www.researchgate.net/figure/GPT-2-model-architecture-The-GPT-2-model-contains-N-Transformer-decoder-blocks-as-shown_fig1_373352176.

[39] M. G. Kavitha and M. E. Gurumoorthi, "Robust and radial image comparison using reverse image search,"

[40] N. M. Phu, C. Xiao, and J. Muindi, "Drawing: A new way to search,"

[41] A. Singla. "Image classification using vision transformer." GitHub repository. (n.d.), [Online]. Available: https://github.com/AarohiSingla/Image-Classification-Using-Vision-transformer/blob/main/image_classifier_from_scratch.ipynb.

[42] "Xception cnn architecture for the detection and classification." ResearchGate. (n.d.), [Online]. Available: https://www.researchgate.net/figure/Xception-CNN-architecture-for-the-detection-and-classification-of-powder-bed-defects-at_fig3_350319854.