DEEP LEARNING FOR SCENE VISUALIZATION AND SENTENCE-BASED IMAGE SYNTHESIS

BY

BEH TECK SIAN

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF INFORMATION SYSTEMS (HONOURS) INFORMATION SYSTEMS ENGINEERING

Faculty of Information and Communication Technology

(Kampar Campus)

MAY 2023

UNIVERSITI TUNKU ABDUL RAHMAN

REPORT STATUS DECLARATION FORM

Title: __**DEEP LEARNING FOR SCENE VISUALIZATION AND SENTENCE-**
__**BASED IMAGE SYNTHESIS**_________________________________

Academic Session: ___MAY 2023__________

I __BEH TECK SIAN____
(CAPITAL LETTER)

declare that I allow this Final Year Project Report to be kept in
Universiti Tunku Abdul Rahman Library subject to the regulations as follows:

1. The dissertation is a property of the Library.
2. The Library is allowed to make copies of this dissertation for academic purposes.

Verified by,

BEH ______ rameh ______

(Author's signature) (Supervisor's signature)

Address:
_51 TERES 2 TINGKAT, ______
_JALAN CORINA 11, TAMAN DESA
_CORINA KAMPUNG RAJA
_39010 CAMERON HIGHLANDS, PAHANG

____Dr.Ramesh Kumar Ayyasamy___
Supervisor's name

Date: _12/9/2023____________ **Date**: ____15/09/2023____

Universiti Tunku Abdul Rahman			
Form Title : **Sample of Submission Sheet for FYP/Dissertation/Thesis**			
Form Number: **FM-IAD-004**	Rev No.: **0**	Effective Date: **21 JUNE 2011**	Page No.: **1 of 1**

FACULTY/INSTITUTE* OF INFORMATION
AND COMMUNICATION TECHNOLOGY

UNIVERSITI TUNKU ABDUL RAHMAN

Date: _12/9/2023________

SUBMISSION OF FINAL YEAR PROJECT /DISSERTATION/THESIS

It is hereby certified that ______***BEH TECK SIAN***___________________________ (ID No: __***19ACB01560***___) has completed this final year project/ dissertation/ thesis* entitled "_***DEEP LEARNING FOR SCENE VISUALIZATION AND SENTENCE-BASED IMAGE SYNTHESIS*** " under the supervision of _ Dr Ramesh Kumar Ayyasamy __ (Supervisor) from the Department of _Information Systems____, Faculty/Institute* of Information and Communication Technology. , and ______________________ (Co-Supervisor)* from the Department of __________________________, Faculty/Institute* of ____________________________.

I understand that University will upload softcopy of my final year project / dissertation/ thesis* in pdf format into UTAR Institutional Repository, which may be made accessible to UTAR community and public.

Yours truly,

_BEH TECK SIAN______
(*Student Name*)

DECLARATION OF ORIGINALITY

I declare that this report entitled “**DEEP LEARNING FOR SCENE VISUALIZATION AND SENTENCE-BASED IMAGE SYNTHESIS”** is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree or other award.

Signature : ___BEH_____________

Name : _BEH TECK SIAN_________

Date : __12/9/2023_______________

ACKNOWLEDGEMENTS

I would like to express my sincere thanks and appreciation to my supervisors, Dr Ramesh Kumar Ayyasamy and my moderator, Ts Dr Khor Siak Wang who has given me this bright and golden opportunity to engage in a deep learning for scene visualization and sentence-based image synthesis project to involve in deep learning and data mining field study. Besides that, they have given me a lot of guidance to complete this project. When I was facing problems in this project, the advice from them always assists me in overcoming the problems. Again, a million thanks to my supervisor and moderator. A million thanks to you.

Other than that, I would like to thank my project teammate, Tan Kean Wei and Yavenraj, who has provided a lot of assistance to me when completing this project. Although both of us are having different project and task scope, he is still willing to support me when I faced difficulties in developing this project. For their patience, unconditional support, and love, and for standing by my side during hard times. Finally, I must say thanks to my parents and my family for their love, support, and continuous encouragement throughout the course.

ABSTRACT

Deep learning and data mining is a subset of machine learning. This project requires to study mainly in field of deep learning and data mining. The research question to be addressed is solve deep learning for scene visualization and sentence-based image synthesis through image classification and image captioning using language python and anaconda navigator. Image classification is a part of project that has many practical applications in different fields, ranging from object recognition, medical imaging, content moderation, and quality control. Image captioning generator is simple which take an image and try to generate a caption that matches the gist of that image closely as possible, which include whole meaning of one picture in just one sentence, which saves times. The image captioning between NLP and computer vision and work in coordination to make image captioning possible and the attention mechanism came to rescue. The methodology and techniques included in the project are research-based project, which in the research process. Research methods and tools to be used were language python and anaconda navigator to launch the jupyter notebook and google colab. Besides that, the dataset was gotten from Kaggle which is Flickr8k Dataset to launch the progress. The platform uses to run the datasets is Jupyter notebook and google colab to run the coding input and give output to judge validity and generality of results. The projects image processing contributes to computer vision applications, such as object detection, classification, and tracking. Scene visualization allows computer understand objects, environment and sentence-based image synthesis enabled computers generate images from textual descriptions. User can random insert picture and system will detect the images given with suitable text description. This project used to generate visual instructions for robots to perform tasks and create more realistic and immersive gaming environments. For advertising and marketing, these techniques can used to generate personalized ads or product recommendations based on customer preferences. For example, sentence-based image synthesis can used to create custom product images based on user input or social media data. These neural networks try to mimic how the human brain functions. Using a public dataset as training data, a deep learning method called CNN used to detect and segment multiple targets in two-dimensional (2D) elemental images for integral imaging system. A range of applications are embracing these techniques to build virtual scenes by verbal description in tandem with advancement of computer graphics, natural language processing, and computing power. Image captioning with start an image and pass it through a pre-trained ImageNet model like inception v3 and produce output feature vectors. Inception v3

is a large network with many pooling, convolution, and fully connected layers which have higher accuracy in the ImageNet dataset which knows as transfer learning for layer output.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

β	beta
Ω	Ohm (resistance)

LIST OF ABBREVIATIONS

5G	Fifth Generation
API	Application Programming Interface
CPU	Central Processing Unit
GPIO	General Purpose Input Output
IOT	Internet of Things
IP	Internet Protocol
RAM	Random Access Memory

CHAPTER 1 INTRODUCTION

In this chapter, I will present the background and motivation of research, contributions to the field, and the outline of the thesis. Deep learning and data mining techniques are effective and quickly evolving in the evaluation of visual aesthetics. It is still a new area, therefore there is plenty of room for more advancements. Using deep learning and data mining to the aesthetic assessment of photographs presents three significant problems with existing datasets aesthetics evaluation have limited amount and categories, aesthetic stimulus is varying in different scenes, and limitations of current image datasets. Sentence-to-image synthesis requires an agent to generate a photo-realistic image according to the given text description. Radiology and radiation oncology both make extensive use of image synthesis across and within medical imaging modalities. In this project, it will motivate to design a multi-scene deep learning framework and provide contribution of make network a strong adaptability to different scenes and to improve the project model performance that balance number of high aesthetic and low aesthetic quality images. In the field of computer vision, deep learning, and data mining, image synthesis from sentence has long been a significant issue. There are some techniques that enable GANs and Inception v3 to produce images related to a specific class. The application of NLP principles to aid the models in language comprehension has resulted in certain works that have achieved notable success. The novel mapping and sampling techniques used by VAEs have also yielded encouraging results.

The project synthesize images from sentence descriptions (known as sentence-to-image synthesis or image-to-sentence synthesis) is an important machine learning tasks requiring ambiguous and incomplete processing explanatory information and learning in natural language across visual and verbal modalities. All targets are detected and segmented from base images in 3D integrated imaging system using deep learning algorithms. Deep-slice images were reconstructed computationally using only segmented targets in the 2D underlying image. The 3D images were then reconstructed using the base images segmented with the detected target. The proposed method works well in the presence of partial occlusions.

The motivation research for visual understanding, with images contain of visual information that will contain visual information, image classification for develop algorithms and models to understand the message from visual data. It will help people for search engine to detect and find for image user want or caption description based on that image to ensure that user can get all the information that no understand based on images with user just need to see one sentence which represent the image whole meaning. For example, when user know that things through picture what is it but don’t know how to call it, this will help user to understand what is related of the picture through caption in one sentence and user now know the name of the things through the image uploaded. We can see that it has potential for many areas, and it can improve between digital and human understanding on it. The contribution can provide correctness information of image classification and image captioning, to enable system to detect and lower the error occur. It turns the images and text description for user to that they cannot see the images graphic. Through this, user can easily find images they want through text queries through the search bar and datasets database. So, it was saving time and money for include large datasets.

1.1 Problem Statement and Motivation

Nowadays, deep learning scene visualization and sentence-based image synthesis industry and more valued by society is something that be easier for user. State-of-the- art algorithms pose the view synthesis problem as the prediction of novel views from an unstructured set or arbitrarily sparse grid of input camera about the view sampling requirements of these methods and predict how their performance will be affected by the input view sampling pattern whether a set of sampled views will produce acceptable results for a virtual experience. Sometimes user have no ideas about it, user can use this long form paragraph to tell and share story. This is an opportunity to showcase a unique moment that captures the spirit of your brand The problem is obtaining object-level segmentation instances is not easy in cluttered and occluded scenes. Therefore, people in this category or those who are new looking to start up might require some guidance or recommendation system to help them solve the problem, it being the main mission for the project. Besides that, correctness to provide the accuracy for the output of sentences with images that can automatically detect the images with the objects and complexness information from the image will also affect the correctness through the output. The image feature will relate to the accuracy of the output and input provided.

The main motivation of developing this project is due to describe how densely a user must capture a given scene for reliable rendering performance. The aim of the thesis is to propose new efficient algorithms for scene visualization and sentence image synthesis. In this thesis, find that it is useful to categorize IBR algorithms by the extent to which they use explicit scene geometry. Hence, these provides motivation for the implementation of this project. Besides, the visual content in nowadays which technology era become important make user can more easily to get the visual content that relevant to requirements can increase user experience by just simple insert the input. Most importantly can improve the effectiveness of the user with just provide simple input and system can generate output for user that was connected to the datasets.

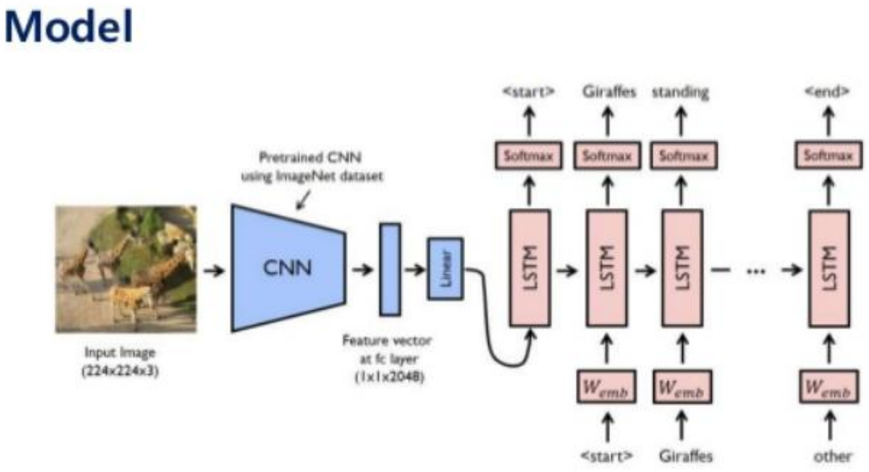


Figure 1.1 Images generated process with captioning.

1.2 Objectives

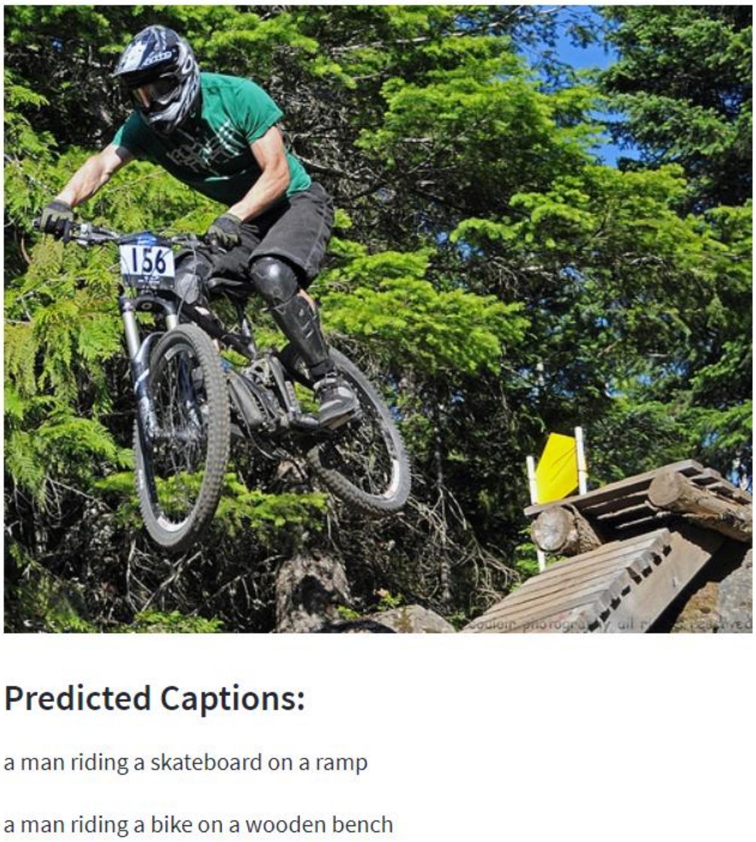
1. The first research objective is to classify the images deep learning through python.

2. The second research objective is to develop a website to increase effectiveness in scene visualization and sentence-based algorithm identification.

3. The third objective is to build a recommendation system that matches the opinion of users with correct result rather than incorrect unrelated results.

1.3 Project Scope and Direction

The scopes of the project include to develop an AI recommendation system which deep learning scene visualization and sentence-based image synthesis that will allow user to create their own scene visualization in graphics that provide with different type of examples to easiest user use it. The system provides user to generate their own synthesis which with sentence to image AI generator and then the AI will recommend user the best suitable and correct results based on user opinion that giving the tutorial provided and recommendation of output for user. For example, when user insert an input image to upload to system, system will provide the description caption for user understand what the image representation means in just one simple sentence. The system of the AI will provide the guidelines for user as references and generate their satisfied scene visualization and synthesis.



Predicted Captions:

a man riding a skateboard on a ramp

a man riding a bike on a wooden bench

Figure 1.3 Upload images to system with provide captions.

1.4 Contributions

Our experiment and analysis confirm the most commitment of this project is an investigation of diverse scene representations for see amalgamation, with the objective of enlightening the important characteristics that make scene representations viable fireproofed learning-based view union. The project visualizes the common issue definition we consider, where the input may be a set of pictures with comparing camera postures, and objective is to recuperate a scene representation that underpins rendering novel sees of the scene. Rather than employing a discrete sampled volume, user speak to the scene as a ceaseless volumetric work, parameterized by a fully connected neural arrange that takes in a 3D facilitate and 2D viewing direction, and yields the volume density and view-dependent color at that location. In this way, the whole scene is encoded within the weights of this profound organize. The project demonstrate that this will be much more effective than an inspected volumetric representation whereas still empowering us to render photorealistic novel sees of the scene.

1.5 Report Organization

This detail of this research report is organized into 6 chapters: Chapter 1 Introduction, Chapter 2 Literature Review, Chapter 3 System Design, Chapter 4 System Implementation and Testing, Chapter 5 System Outcome and Discussion, Chapter 6 Conclusion. The first chapter is the introduction of this project which includes problem statement, project background and motivation, project scope, project objectives, project contribution, highlights of project achievements, and report organization. The second chapter is the literature review carried out on several existing solving method or background study related to deep learning scene visualization and image captioning or image classification theory. The third chapter is discussing the overall system design of this project. The fourth chapter is regarding the details on how to implement the design of the system. Furthermore, the fifth chapter reports the systems outcome that image caption that upload file to run and generate output. The last chapter output related to conclusion of the overall project.

CHAPTER 2 LITERATURE REVIEW

2.1 Google colab

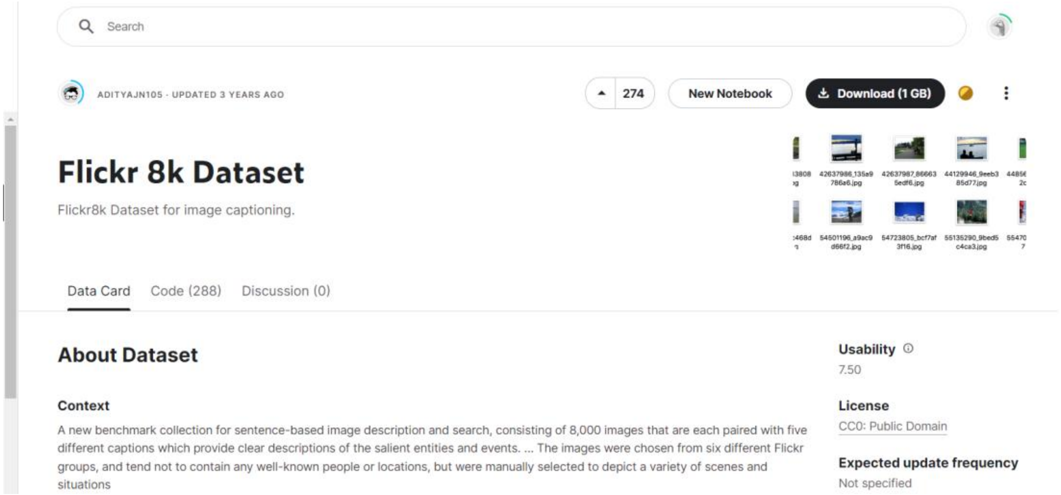
Google colab is a platform based on cloud platform for connect with anaconda to launch the jupyter notebook to run my python program which is a program that with ipynb file. I use it to write and launch my code for image classification and image captioning code project. I insert Kaggle dataset into Goggle colab to run my program with the dataset provide with images and captions.txt file. With these datasets, it provides opportunities for the output accuracy.

Table 2.1 Specifications of laptop

Description	**Specifications**
Model	Asus A510U series
Processor	Intel Core i5-8250U
Operating System	Windows 11
Graphic	NVIDIA GeForce GT 930MX 2GB DDR3
Memory	12GB DDR4 RAM
Storage	128GB SATA HDD

2.2 Kaggle Datasets

This is datasets I used for launching my program that contains with picture and captions.



Flickr 8k Dataset

Flickr8k Dataset for image captioning.

About Dataset

Context

A new benchmark collection for sentence-based image description and search, consisting of 8,000 images that are each paired with five different captions which provide clear descriptions of the salient entities and events. ... The images were chosen from six different Flickr groups, and tend not to contain any well-known people or locations, but were manually selected to depict a variety of scenes and situations

Usability
7.50

License
CC0: Public Domain

Expected update frequency
Not specified

Figure 2.2 Kaggle Dataset

2.3 Visual Studio code

I also use visual studio code for launching my program in python language with connect to streamlit website.

2.4 Previous works on Deep learning and Data Mining

2.4.1 U-Net: Convolutional Networks for Biomedical Image Segmentation

In the field of biomedical image segmentation, the U-Net architecture, introduced by Ronneberger, Fischer, and Brox in 2015, has emerged as a significant advancement. Traditional convolutional networks faced limitations due to the scarcity of annotated training samples and the need for pixel-level localization in biomedical tasks. U-Net addresses these challenges by employing a unique architecture that combines a contracting path for context capture with an expansive path for precise localization. Notably, U-Net demonstrates its effectiveness in training with limited data through extensive data augmentation, particularly elastic deformations. This innovation enables the network to learn invariance and robustness properties essential for biomedical image processing. Moreover, U-Net's seamless tiling strategy facilitates the segmentation of large images. It has found success in various applications, from segmenting neuronal structures in electron microscopy to cell tracking in light microscopy, consistently outperforming previous methods, and showcasing its versatility in biomedical image segmentation tasks.

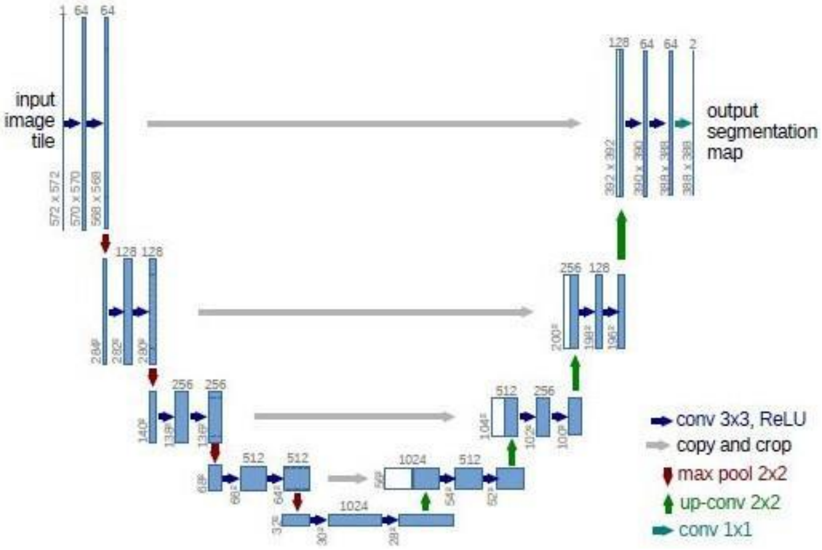


Figure 2.4 U-net architecture

2.4.2 Strengths and Weakness

The U-Net architecture offers several notable strengths. Firstly, its ability to achieve exceptional segmentation accuracy with limited annotated training data sets it apart. This is made possible through the effective use of data augmentation, such as elastic deformations, which allows the network to learn critical invariance and robustness properties, a crucial requirement in biomedical image processing where datasets are often small and varied. Secondly, the U-Net's architecture is uniquely designed to combine a contracting path for context capture with an expansive path for precise localization, enabling it to provide highly accurate pixel-level localization, a fundamental requirement in biomedical segmentation tasks. Additionally, its seamless tiling strategy allows for the efficient segmentation of large images, overcoming GPU memory limitations. Lastly, its rapid execution, with the ability to segment a 512x512 image in less than a second on a modern GPU, contributes to its practical usability.

Despite its many strengths, U-Net does have some limitations. One significant drawback is its susceptibility to overfitting, particularly when dealing with extremely limited training data. Although data augmentation helps mitigate this issue to some extent, acquiring more annotated data remains a challenge in many biomedical applications. Additionally, while the architecture's emphasis on precise localization is advantageous, it may compromise on computational efficiency when compared to faster, less accurate segmentation methods. Lastly, U-Net's applicability may be subject to the specific nature of the biomedical segmentation task at hand; its performance can vary based on the characteristics of the dataset and the complexity of the objects being segmented. Therefore, it is essential for researchers to consider the trade-off between accuracy and computational efficiency when selecting U-Net for a given biomedical image segmentation task.

2.5 Image Processing Deep Learning

2.5.1 Effect of Attention Mechanism in Deep Learning-Based Remote Sensing Image Processing: A Systematic Literature Review

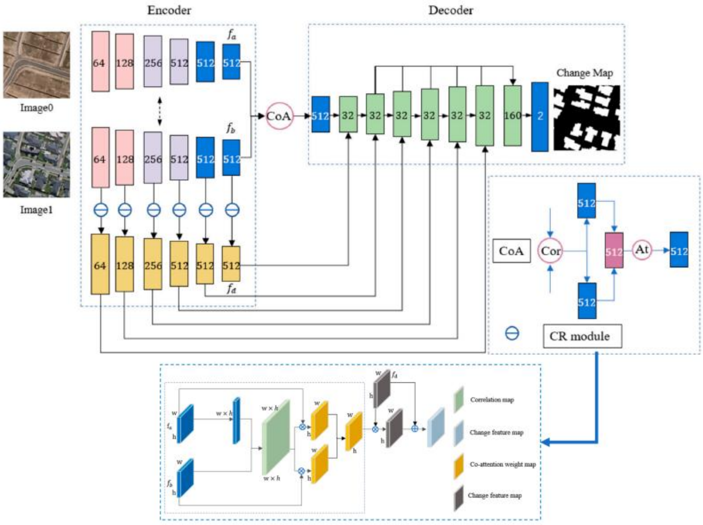


Figure 2.5 CNN Structure

In summary, the reviewed literature showcases the remarkable progress achieved at the intersection of deep learning and remote sensing. Researchers have devised innovative solutions employing attention mechanisms, convolutional and recurrent neural networks, and novel architectures to address a wide range of remote sensing challenges. These applications encompass image segmentation, object detection, image super-resolution, change detection, and classification, among others. Beyond traditional tasks, the literature explores hyperspectral imagery, transfer learning, and generative adversarial networks. It also emphasizes the importance of handling small sample sizes and cloud removal. Overall, this body of work highlights the transformative potential of AI-driven geospatial analysis for diverse scientific and practical applications in the field of remote sensing.

2.5.2 Strengths and Weakness

The reviewed literature on deep learning applications in remote sensing highlights a compelling array of strengths and weaknesses in this rapidly evolving field. On the positive side, these studies underscore the transformative potential of deep learning techniques, showcasing significant enhancements in the accuracy and automation of remote sensing tasks. Innovative approaches like attention mechanisms, advanced network architectures, and state-of-the-art techniques have collectively contributed to impressive progress in tasks such as object detection, image segmentation, and classification. These advancements hold great promise for practical applications in diverse domains, including disaster management, agriculture, and environmental monitoring.

Nevertheless, several noteworthy weaknesses are also discernible in this body of research. The most prominent among them are the substantial data and computational requirements, along with a reliance on domain-specific expertise. The deep learning models discussed in these studies often necessitate large volumes of labeled data for training, making them less accessible for applications with limited access to such resources. Moreover, the computational demands of these models can be prohibitive for some users or regions. Furthermore, the interpretability and explain ability of complex deep learning models pose significant challenges, especially in critical applications where decision-making transparency is crucial. Additionally, the generalization of deep learning models across different environmental and geographical conditions remains a pressing concern that necessitates further exploration. In conclusion, while deep learning holds immense promise for remote sensing applications, addressing these challenges is essential to fully harness its potential while ensuring broader accessibility and reliability.

2.6 Text-to-Image Synthesis

2.6.1 Object-driven Text-to-Image Synthesis via Adversarial Training

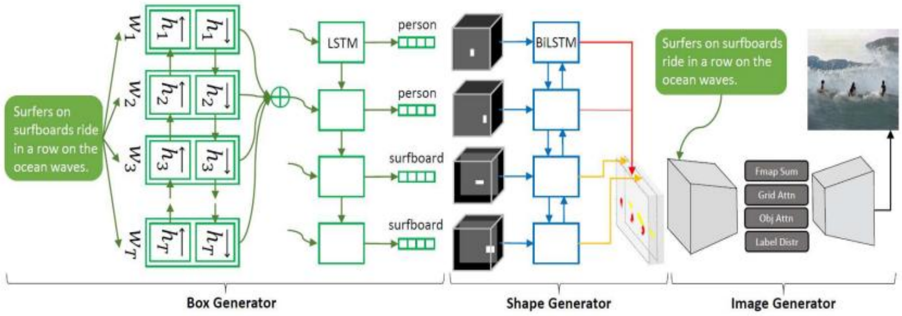


Figure 2.6 Obj-GAN network

This literature review provides an in-depth examination of the recent progress in text-to-image synthesis, with a primary focus on the pioneering Object-Driven Attentive Generative Adversarial Networks (Obj-GAN). Obj-GAN represents a significant advancement in the field by introducing a multi-stage architecture that leverages object-driven attention mechanisms. This innovation enables the model to generate images with an unprecedented level of detail and fidelity, driven by fine-grained information extracted from both text descriptions and object-level features. The inclusion of object-wise discriminators, built on the foundation of Fast R-CNN, further enhances the model's capability to generate images conditioned on complex textual inputs. The empirical evaluation of Obj-GAN against previous state-of-the-art models, particularly on the notoriously challenging COCO dataset, consistently demonstrates its superiority in terms of various metrics, including Inception scores, R-precision, and FID. Notably, Obj-GAN's robust generalization ability, as exemplified by its successful generation of images for novel and unconventional textual descriptions, underscores its potential for real-world applications in creative content generation and visual storytelling. In summary, Obj-GAN stands out as a remarkable contribution to the field of text-to-image synthesis, pushing the boundaries of what is achievable in generating visually rich and contextually coherent images from textual descriptions.

2.6.2 The strengths and weaknesses

Obj-GAN presents several notable strengths in the context of text-to-image synthesis. Firstly, its innovative object-driven attention mechanism allows the model to capture fine-grained details and context in both textual descriptions and visual features, resulting in images that are not only visually appealing but also semantically coherent with the input text. This fine-grained attention enables Obj-GAN to excel in generating complex scenes with multiple objects, making it highly suitable for a wide range of applications such as content generation, creative design, and more. Secondly, the incorporation of object-wise discriminators, based on Fast R-CNN, provides a robust means to condition image generation on object-level information, enhancing the model's ability to generate images that align with the content and layout specified in the input text. Additionally, Obj-GAN's generalization capability, as evidenced by its successful generation of images for novel textual descriptions, highlights its potential for practical applications where creative content generation is essential.

Despite its strengths, Obj-GAN does have some weaknesses. Firstly, its computational demands can be substantial, particularly when generating high-resolution images or when training on large datasets. This computational complexity may limit its accessibility to researchers and practitioners with limited resources. Secondly, like many generative models, Obj-GAN's performance can be sensitive to the quality and diversity of the training data. Biases or limitations in the training dataset can lead to undesirable artifacts or biases in the generated images. Lastly, while Obj-GAN has demonstrated remarkable capabilities in text-to-image synthesis, there is always room for further improvement in terms of achieving even higher levels of realism and fine-grained details. As the field continues to evolve, addressing these computational and data-related challenges and pushing the boundaries of image synthesis will be crucial for Obj-GAN to reach its full potential.

2.7 Attention Mechanism Deep Learning

2.7.1 A review on the attention mechanism of deep learning

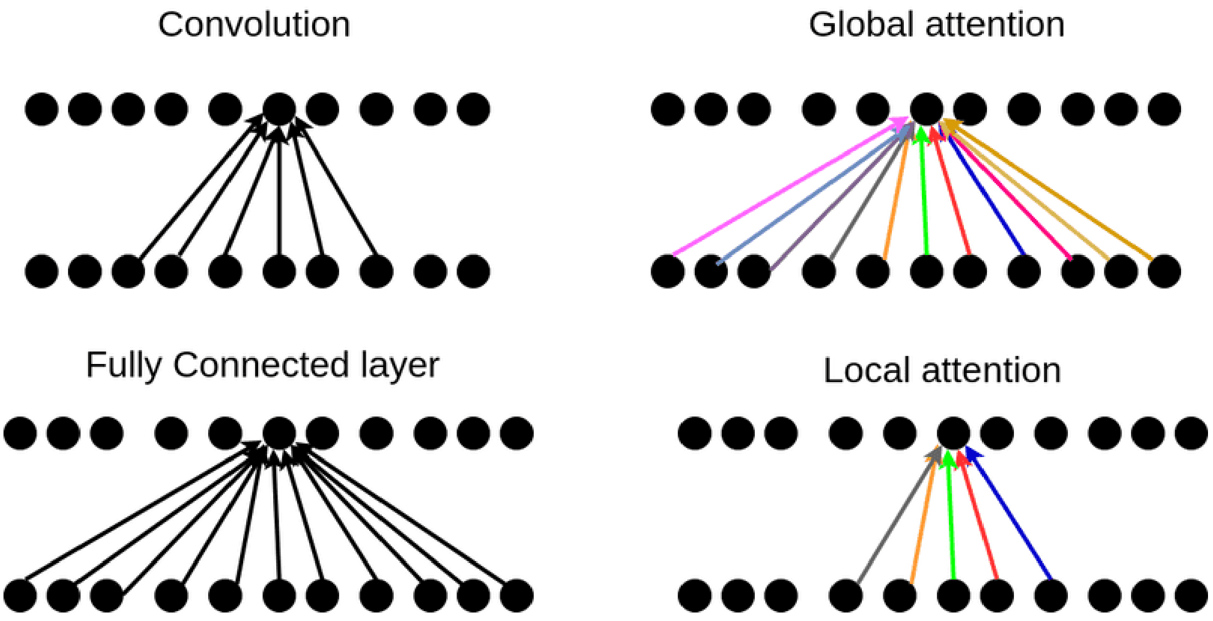


Figure 2.7 Attention Mechanism Deep Learning

In the realm of neural networks, attention mechanisms have rapidly evolved since their pioneering application in machine translation. A unified attention model, as illustrated in this literature review, encompasses the core components shared by most attention models. This model involves two fundamental steps: computing the attention distribution on input data, achieved by encoding source data as keys and introducing task-related queries, and subsequently computing context vectors based on this distribution. The choice of attention type can be characterized as either soft (deterministic), such as Bahdanau's weighted average approach, which is differentiable and suitable for standard back-propagation training, or hard (stochastic), as exemplified by Xu's method involving stochastically sampled keys, introducing an element of randomness in key selection. The flexibility and adaptability of attention mechanisms make them a vital tool in diverse neural network applications.

The model's implementation of the attention mechanism includes detailed descriptions of each phase. Additionally, we categorize existing attention models using the following four parameters: the softness of the attention, the types of input feature, the input representation, and the output representation.

2.7.2 The strength and weakness

Attention mechanisms significantly enhance the performance of deep learning models in various tasks, such as machine translation, image captioning, and speech recognition. They enable models to focus on relevant parts of the input data with improve model performance. At multimodal integration, they enable the integration of information from different modalities with text and images, making it possible to build models that can process and generate content across various data types. With Efficiency, Attention mechanisms can improve the efficiency of model training and inference by reducing the need to process the entire input sequence at every step, leading to faster convergence.

The weakness includes computational complexity which include attention mechanisms can be computationally expensive, especially when dealing with large sequences or complex models. This can hinder their scalability. With the second data efficiency, attention mechanisms often require a large amount of data for training to generalize well, which can be challenging in domains with limited data availability. For Interpretability Complexity, attention mechanisms offer interpretability, understanding the exact decision-making process of complex models with attention can be challenging, especially when dealing with deep neural networks.

2.8 Proceedings of Machine Learning Research

2.8.1 Show, Attend and Tell: Neural Image Caption Generation with Visual Attention

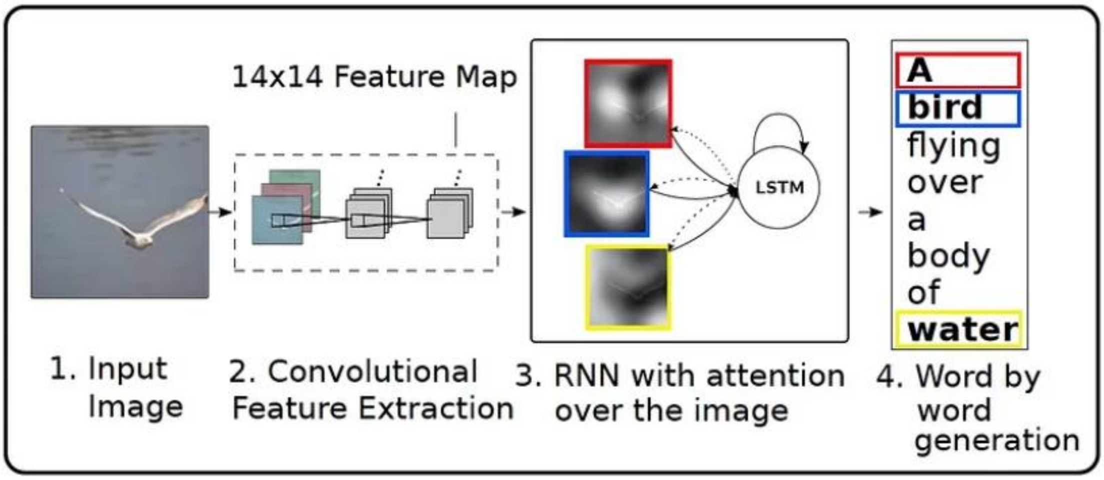

14x14 Feature Map
LSTM
A
bird
flying
over
a
body
of
water
1. Input Image
2. Convolutional Feature Extraction
3. RNN with attention over the image
4. Word by word generation

Figure 2.8 Neural Image Caption

It offers an attention-based model that automatically learns to describe the content of images. This model was inspired by previous work in machine translation and object detection. They demonstrate how they can train this model both stochastically by maximizing a variational lower limit and deterministically by utilizing conventional backpropagation approaches. Through visualization, they also demonstrate how the model can automatically develop a fixation on salient things while producing the words for those objects in the output sequence. With cutting-edge performance on three benchmark datasets—Flickr8k, Flickr30k, and MS COCO—we validate the application of attention.

2.8.2 The strength and weakness

The first strength is improved caption quality to models with visual attention typically produce higher-quality captions that are more aligned with human perception compared to non-attention-based models. Next with interpretability that Visual attention provides a degree of interpretability by showing where the model is focusing within the image when generating each word in the caption. This can help users understand why a particular word or phrase was chosen.

The weakness which includes computational complexity that use of visual attention adds computational overhead, as the model needs to attend to different parts of the image at each time step. This complexity can slow down training and inference, requiring powerful hardware resources. It faces alignment challenges to getting the alignment between the visual and textual information just right can be a challenging problem, especially when dealing with complex images with multiple objects or intricate scenes. The robustness to image variations effect model performance when faced with variations in image quality, lighting conditions, or unusual perspectives that were not well-represented in the training data.

2.9 Image Captioning

2.9.1 Image Captioning Using Inception V3 Transfer Learning Model

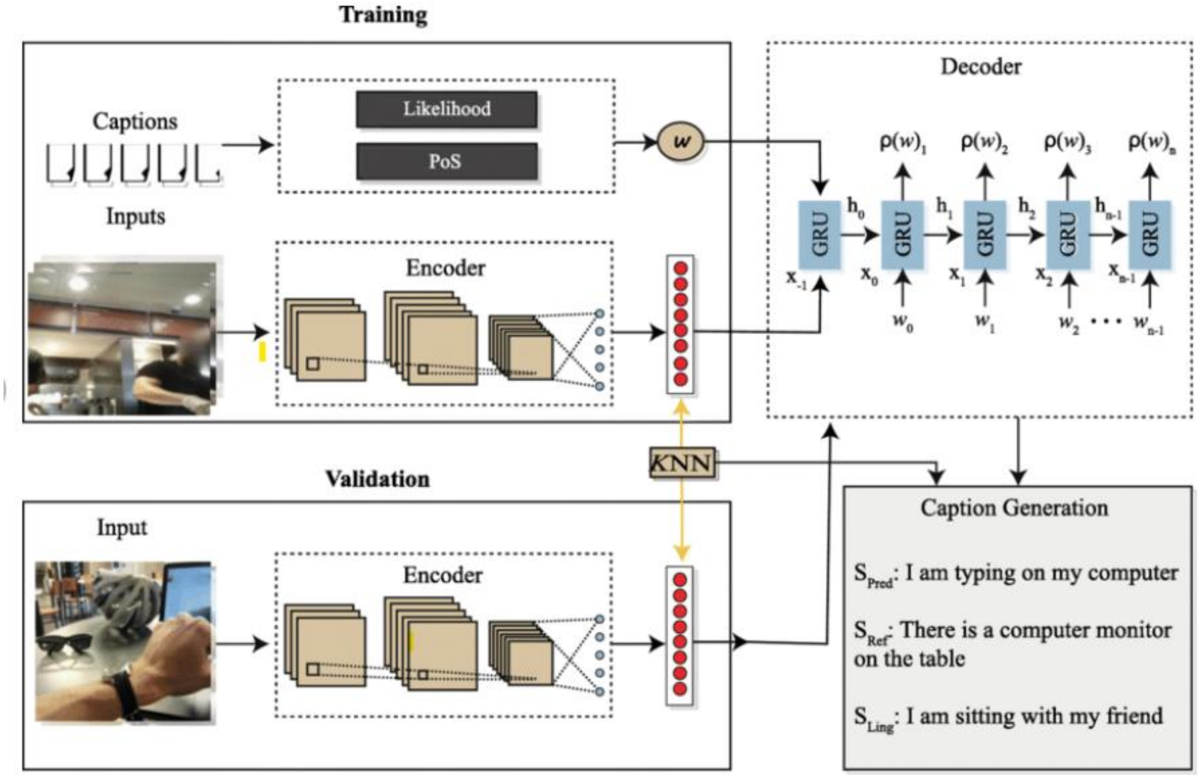


Figure 2.9 Image Captioning

As artificial intelligence has increased in popularity in recent years, photo captioning has piqued the interest of many experts, making it an interesting and hard problem. Visual subtitles, which automatically generate natural language interpretations based on image data, are an important component of scene analysis, which combines machine vision and natural language processing capabilities. This study employs various NLP methodologies for perceiving and explaining the meaning of an image in a natural language such as English. CNN (Coevolutionary Neural Networks) and LSTM (Long Short-Term Memory) units are used in the proposed Inception V3 picture caption generating model. On an ImageNet dataset, the InceptionV3 model was trained in 1000 different classes. The model was imported straight from the Keras application module. Remove the last classification layer for the dimension (1343,) vector from the InceptionV3model. The embedded matrix is used to connect vocabulary. A building matrix is the linear transformation of a real-life space from an original space with crucial relationships. Image captions are widely utilized and, for example, play a vital role in developing human-computer interaction.

2.9.2 The strength and weakness

The first strength is excellent quality of the visual features because of Inception V3 is deep convolutional neural network (CNN) use to train at big datasets and it can train low and high-level of visual features from image efficiency, it provides strong basic for generate the output for images. For state-of-the-art performance that on various picture classification benchmarks, Inception V3 demonstrated cutting-edge performance. When applied to image captioning, it frequently yields high-quality captions.

The weakness includes it lacks understanding on the image features context knowledge required to generate meaningful and contextually suitable captions. It may have difficulty comprehending the relationships between things in an image. It also has complexity on implementation on system that learning models for picture captioning, such as Inception V3, can be difficult, especially for those with little familiarity with deep learning and model adaption. Inception V3 is fixed visual features that may not adapt effectively to visuals with varied levels of complexity or viewpoints. It may also have difficulty comprehending dynamic scenes or video frames.

2.10 Mobile App for Text-to-Image Synthesis

2.10.1 Mobile App for Text-to-Image Synthesis

The creation of visual representations of textual data is a difficult but fascinating issue with a wide range of potential applications. To improve language teaching, we provide a novel method for visualizing natural language sentences using ImageNet in this study. The current emphasis is on helping English language learners expand their knowledge of common nouns and have a thorough understanding of the numerous prepositions of location. To accomplish this, real-world photographs of nouns are taken from ImageNet, then using image segmentation, their foreground items of interest are retrieved. Then, based on the spatial relationship indicated in the text, the objects are rearranged on a canvas. They created a mobile application that uses the RESTful API to receive the photographs from the web service that runs the image production software to show the viability of the suggested strategy. To aid in the learning of new vocabulary and spatial prepositions during language education, the prototype mobile application may produce visual representations of natural language sentences and a written description of the spatial relationship of objects.

Low-level qualities are the focus of common forms of picture altering techniques. In this thesis, they use machine learning to enable more conceptually advanced image manipulation. The fundamental goal of the suggested techniques is to separate the information that can be changed by incorporating the general visual knowledge from the information that must be maintained in the editing process. The new techniques can thereby alter photos in ways that are understandable to humans, such as changing one thing into another, stylizing photographs into the works of a certain artist, or including a sunset in a daytime photograph. Per-pixel labels, per-image labels, and no labels are some of the numerous scenarios in which we investigate the design of such procedures with differing degrees of supervision. First, they suggest a new deep neural network architecture that can create realistic images from scene layouts and optional target styles using per-pixel supervision. Second, investigate the domain translation job, which involves converting an input image of one class into another, employing per-image supervision. Finally, provide a framework that can still identify texture and structural alteration from a set of unlabeled photos. In a variety of applications, including interactive photo painting tools, object transformation, bridging the gap between the virtual and physical worlds, and realistic texture manipulation, we deliver visually compelling results.

2.10.2 The strength and weakness

The advantage includes enhanced learning that related of visual representations can enhance language learning, especially for English language learners, by providing concrete images that help students understand and remember vocabulary and spatial prepositions. Visual content can make language learning more engaging and interactive, appealing to a wider range of learners through engagement. Using real-world photographs from ImageNet provides learners with context and examples from everyday life through real-world context is one of the strengths. The method can be customized to suit different learning levels and objectives, making it adaptable for various educational contexts with customization. The mobile application makes it accessible to learners on their smartphones or tablets, allowing for on-the-go learning with accessibility app.

The weakness were complexity and limited coverage which developing the technology to automatically generate meaningful visual representations from text can be complex and resource intensive. The method may not cover all aspects of language learning, and some nuances of language may not translate well into images. The effectiveness of the method depends on the availability and reliability of technology, including internet connectivity. In this case, without internet connectivity, the system cannot function. Lastly, the use of manipulated images can raise legal issues related to copyright, intellectual property, and misrepresentation.

2.11 Deep Learning for Scene Classification

2.11.1 Deep Learning for Scene Classification: A Survey

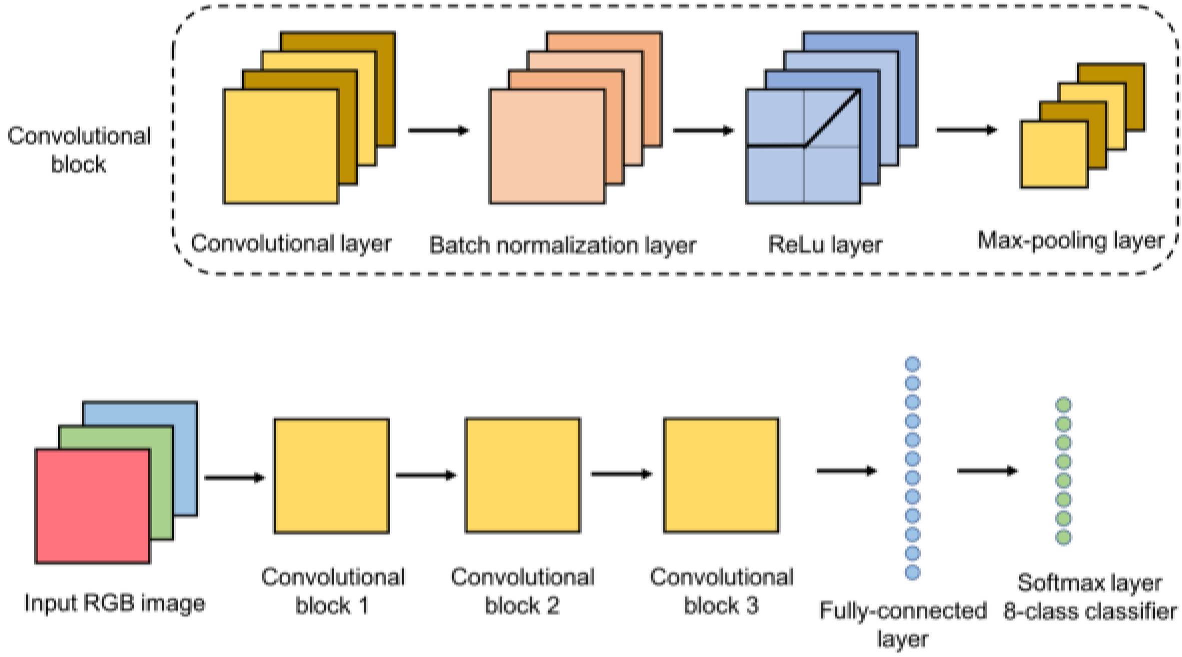


Figure 2.11 A Baseline CNN used.

Scene classification is a long-standing, essential, and difficult subject in computer vision. It aims to categorize a scene image into one of the specified scene categories by understanding the full image. Scene representation and classification have advanced significantly because of the emergence of large-scale datasets, which serve as a corresponding dense sampling of a variety of real-world scenes, and the renaissance of deep learning techniques, which learn potent feature representations directly from big raw data. The purpose of this work is to present a thorough assessment of current developments in deep learning-based scene categorization to aid researchers in understanding the necessary advancements in this field. This study includes more than 200 significant papers that discuss many facets of scene categorization, such as difficulties, benchmark datasets, taxonomy, and quantitative performance evaluations of the algorithms under consideration. This report also includes a list of prospective areas for future research that are discussed in the context of what has already been accomplished.

2.11.2 The strength and weakness

The strength as we can see was high accuracy potential which means that deep learning models have shown remarkable success in various computer vision tasks, including scene classification. They can learn complex features and patterns from large datasets, potentially leading to high classification accuracy. Deep learning models can adapt to a wide range of scene categories and variations in lighting, perspective, and object composition, making them versatile for scene classification tasks which have good compatibility. Deep learning models can handle large-scale datasets, allowing for the training of robust classifiers on extensive collections of scene images for scalability.

The weakness includes deep learning models often require large volumes of labeled datasets for training. Gathering and annotating such datasets can be time-consuming and expensive, particularly for niche or uncommon scene categories. Training deep learning models can be computationally intensive, requiring access to powerful GPUs or specialized hardware. This can be a barrier for smaller research groups or organizations with limited resources. Deep learning models are often viewed as black boxes, making it challenging to interpret their decision-making processes. This lack of interpretability can be a drawback, especially in applications where transparency and accountability are essential.

2.12 Image Classification: A Literature Review

2.12.1 Scene Level Image Classification: A Literature Review

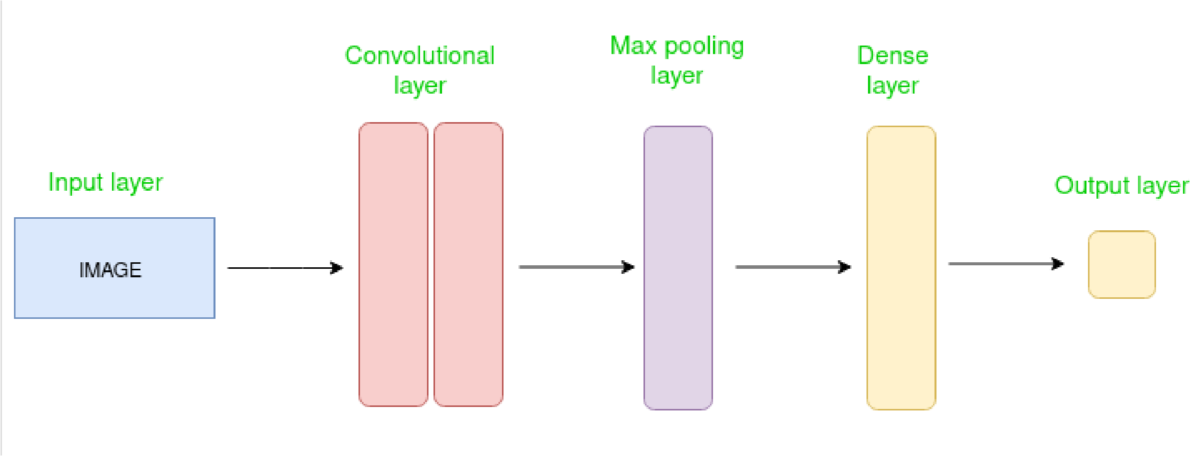


Figure 2.12 CNN Network diagram

Since the advent of deep learning, convolutional neural networks (CNNs) have made important advances to natural and remote sensing imaging. Scene-level picture classification is a challenge with diverse applications that affects both the natural and remote sensing realms. The focus is on the amount of probable scene items in the image content that could match the dataset images. Because of unresolved difficulties such as intraclass heterogeneity, interclass homogeneity, background cluttering, high spatial resolution, and variable imaging settings, scene-level categorization is significant and exciting. Furthermore, the imbalance, lack of preservation of complex semantic linkages, and greater label-to-label correlation are all visible in the multi-label scene dataset. The paper presents a meta-analysis of current scene classification literature approaches.

CNNs, attention mechanisms, capsule networks, and generative adversarial networks are all discussed. The paper also provides a summary of the scene domain's numerous activations, losses, optimization strategies, and regularization schemes. The standard benchmark datasets are compiled based on single- and multi-label themes. The performance measures for scene classification are also discussed. The paper also discusses the implementation of multi-label scene categorization using multiple CNN models on the UC Merced multi-label dataset. The suggested Mobile Net-based model outperforms the established cutting-edge techniques.

2.12.2 The strength and weakness

CNNs are very good at automatically learning hierarchical features from raw data. They can detect complex patterns and objects in photos by capturing low-level information such as edges and textures and combining them. CNNs are spatially invariant, which means they can recognize features regardless of where they are in an image. This characteristic is required for tasks such as object detection and scene comprehension. CNNs can handle larger and more complicated information, making them useful for a wide range of applications and sectors. CNNs have attained state-of-the-art performance in numerous computer vision benchmarks and contests, even surpassing human-level performance in some circumstances. CNNs can process pictures in parallel in an efficient manner, making them appropriate for real-time applications such as video analysis and autonomous driving.

The shortcoming CNNs sometimes require large, labeled datasets for training, which can be costly and time-consuming to produce, particularly in specialized fields. Deep CNN training can be computationally demanding, involving strong GPUs or specialized hardware. This may make them inaccessible to researchers with minimal resources. CNNs are sometimes regarded as black-box models, making it difficult to understand how they make judgments. In situations where transparency is critical, this lack of interpretability can be a disadvantage. While CNNs excel at spotting patterns in images, they may lack a comprehensive knowledge of context or semantic meaning. This can be a disadvantage in tasks that need high-level reasoning. CNNs can perpetuate biases inherent in training data, raising questions about fairness and ethics in applications such as facial recognition and criminal justice.

2.13 A comprehensive Review on Image Synthesis with Adversarial Networks

2.13.1 A Comprehensive Review on Image Synthesis with Adversarial Networks: Theory, Literature, and Applications

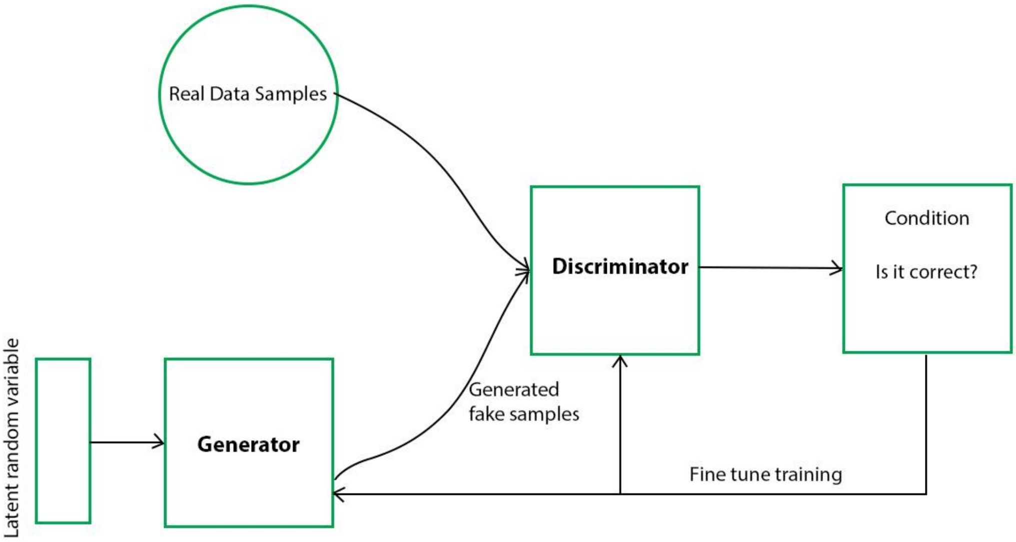


Figure 2.13 Generative Adversarial Networks (GANs)

Deep learning has had a significant impact in engineering and science in recent years. One of the most appropriate fields is the synthesis and editing of images. Image synthesis is a branch of computer vision and expert systems. GANs (generative adversarial networks) have received a lot of interest since they outperform traditional adversarial networks the customary ways. They can also be utilized in numerous picture synthesis and editing applications, such as human image synthesis and face recognition, aging, text-to-image synthesis, and 3D image synthesis are some of the techniques used. Several cutting-edge image synthesis and editing techniques are featured in this survey.

Techniques for creating fake images using convolutional neural networks are described. We also explore the benefits, drawbacks, and features of such methods, as well as how image quality varies with the amount of the dataset used for learning. Finally, we will look at some methods for detecting fraudulent photos created by image synthesis techniques.

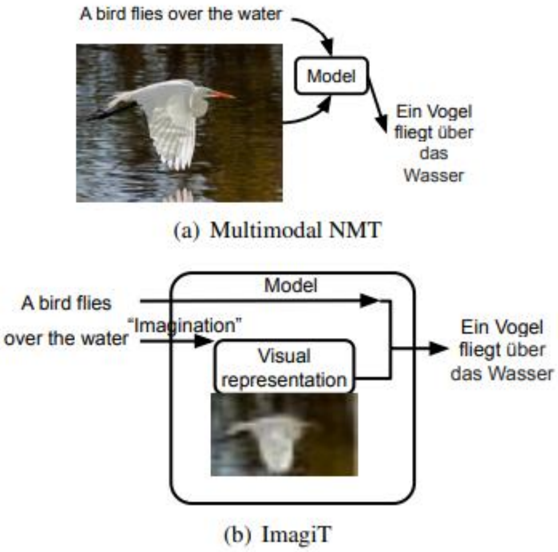
2.13.2 The strength and weakness

GANS's strength is high-quality data production, which includes the capacity to generate high-quality, realistic data such as images, audio, and text. This can be used to create realistic visuals for art, design, and entertainment. GANs have produced cutting-edge outcomes in a variety of picture production and manipulation tasks, paving the way for advances in computer vision and image processing. GANs have been employed in creative applications like as painting, music, and poem generation, exhibiting their capacity for artistic expression and innovation. GANs are adaptable to a variety of data generating tasks, such as image-to-image translation, style transfer, super-resolution, and text-to-image synthesis, making them useful for a wide range of applications.

The weakness include GANs are often viewed as black-box models, making it challenging to understand the decision-making process behind generated content which lack of interpretability. Evaluating the quality of GAN-generated data is a challenging problem, as traditional metrics like mean squared error may not capture the perceptual quality of the generated content accurately. GAN training can be notoriously unstable and sensitive to hyperparameters. Achieving convergence and avoiding mode collapse where the generator only produces a limited set of samples can be challenging.

2.14 Generative Imagination Elevates Machine Translation

2.14.1 Generative Imagination Elevates Machine Translation


A bird flies over the water
Model
Ein Vogel fliegt über das Wasser
A bird flies over the water
Model
"Imagination"
Visual representation
Ein Vogel fliegt über das Wasser

(a) Multimodal NMT

(b) ImagiT

Figure 2.14 ImagiT

ImagiT stands as a groundbreaking advancement in machine translation, harnessing visual cues to enhance the translation process without the need for annotated images. This model seamlessly intertwines text-to-image synthesis, image captioning, and neural machine translation to construct semantic-consistent visual representations, which serve as invaluable guides during translation. Extensive evaluations across diverse datasets underscore ImagiT's remarkable performance gains when compared to traditional text-only neural machine translation models, while also positioning it as a competitive contender in the realm of multimodal translation systems. One of ImagiT's most distinctive features is its ability to imaginatively reconstruct missing textual information, offering the potential to address complexities in sentence comprehension and decipher unfamiliar terminology. Additionally, ImagiT displays adaptability by incorporating external data sources, promising further enhancements as it encounters a wider array of data types and domains. However, its applicability may be contingent on the nature of the task, particularly in scenarios with less visually interpretable content, emphasizing the importance of thoughtful evaluation and understanding of its behavior in varying contexts for practical utilization.

2.14.2 The strength and weakness

ImagiT boasts several notable strengths that distinguish it in the field of machine translation. Firstly, it introduces a novel approach to multimodal translation, bridging the gap between text and images through imaginative generation. This unique ability to create semantic-consistent visual representations during the translation process is a significant strength, enabling ImagiT to enhance translation quality, particularly when dealing with complex or domain-specific content where images could provide crucial context. Furthermore, ImagiT offers versatility by not relying on annotated images, making it applicable to a broader range of translation tasks, even in low-resource scenarios. It also showcases adaptability through its capability to incorporate external data sources, potentially improving performance as it encounters a more extensive array of data types and domains. Additionally, ImagiT demonstrates the potential to recover missing textual information, which can be beneficial in addressing sentence comprehension difficulties and deciphering unfamiliar terminology.

However, ImagiT also exhibits certain limitations. One notable weakness is its sensitivity to the availability of visually interpretable content. In tasks where textual descriptions do not readily translate into visual representations, such as topics involving economics or politics, ImagiT may not perform optimally. Additionally, while ImagiT reduces the need for annotated images, its performance hinges on the quality of the generated visual representations. The model may struggle when faced with challenges like generating highly realistic or detailed images, potentially impacting translation quality. Furthermore, the successful implementation of ImagiT relies on finding the right balance between text and visual information, as well as effectively managing ambiguity and bias in the training data to avoid undesired model behaviors. Careful evaluation and understanding of ImagiT's behavior in different contexts are essential for practical usage.

2.15 Image classification.

2.15.1 In defense of Nearest-Neighbor based image classification.

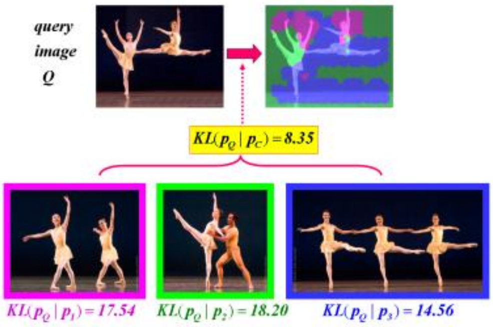


Figure 2.15 Image to image

In conclusion section of the paper, the authors summarized their findings from experimental evaluations of the Non-Parametric Binary Nearest Neighbor (NBNN) classifier. Their experiments were conducted on the challenging Graz-01 dataset, which involves complex object classification tasks with high intra-class variations and background clutter. The results revealed that despite its simplicity and the absence of a learning phase, NBNN performed remarkably well, competing favorably with more sophisticated learning-based classifiers, including Boosting-based and SVM-based approaches. This demonstrated the robustness and competitiveness of NBNN in handling challenging object recognition tasks, particularly characterized by complex background clutter and significant intra-class variations.

The literature review section of the paper cited several key references that contextualize the research within the broader field of image classification and recognition. Notable references include works by Berg on shape matching and object recognition, Lazebnik et al. on spatial pyramid matching, Opelt et al. on weak hypotheses and boosting for object detection and recognition, and Zhang et al. on local features and kernels for texture and object category classification. Additionally, references to datasets like Caltech-256 and papers discussing generative visual models and feature extraction trade-offs underscore the relevance and significance of NBNN's performance evaluations in comparison to various state-of-the-art approaches in image classification. This comprehensive literature review serves to position the research within the broader academic landscape and highlights the contributions and competitive advantages of the NBNN classifier.

2.15.2 The strength and weakness

One of the notable strengths of the Non-Parametric Binary Nearest Neighbor (NBNN) classifier, as demonstrated in the paper's experimental results, is its impressive performance in object classification tasks. Despite its simplicity and the absence of a learning or training phase, NBNN consistently achieved competitive accuracy rates on challenging datasets such as Caltech-101 and Caltech-256, even outperforming some complex learning-based classifiers. This highlights NBNN's ability to handle image classification tasks effectively, particularly when dealing with high intra-class variations and complex background clutter. Its robustness and efficiency, along with its non-parametric nature, make it a valuable tool for various computer vision applications.

While NBNN showcases notable strengths, it also has some inherent limitations. One key weakness is its sensitivity to the choice of descriptors. The classifier's performance heavily relies on the quality and relevance of the chosen descriptors, and selecting inappropriate or insufficient descriptors can lead to suboptimal results. Additionally, NBNN might not be the ideal choice for tasks that require real-time processing or extremely large datasets, as it relies on exhaustive nearest-neighbor searches, which can be computationally expensive for extensive collections. Thus, its suitability depends on the specific requirements of the computer vision task at hand, and users should carefully consider the choice of descriptors and computational resources when employing NBNN for image classification.

2.16 Residual Attention Network

2.16.1 Residual Attention Network for Image Classification

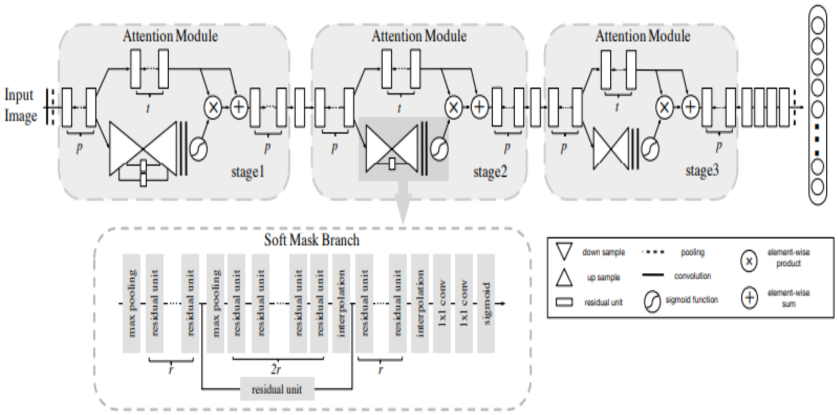


Figure 2.16 Network ImageNet

The experiments conducted on the ImageNet dataset have yielded compelling evidence of the efficacy of Residual Attention Networks, specifically the Attention-56 and Attention-92 variants, in significantly improving image classification performance. These networks have outperformed well-established models like ResNet-152 and ResNet-200 while managing to substantially reduce the number of parameters and overall computational complexity. This achievement underscores the remarkable efficiency of Residual Attention Networks in handling large-scale image classification tasks. Moreover, the experiments have demonstrated the remarkable adaptability of these networks to different basic units, including ResNeXt and Inception, further highlighting their versatility within various network architectures. This adaptability holds great promise for their utilization in diverse computer vision tasks beyond image classification, such as object detection and segmentation. In summary, these findings underscore the pivotal role of attention mechanisms within convolutional neural networks and their potential to redefine the landscape of deep learning in computer vision applications.

2.16.2 The strength and weakness

Residual Attention Networks (RANs) offer several notable strengths in the realm of computer vision. First and foremost, RANs excel in image classification tasks, consistently outperforming state-of-the-art models while maintaining a reduced model size and computational complexity. This efficiency makes them highly practical for real-world applications where computational resources are limited. Additionally, RANs exhibit an impressive level of versatility, as they can be seamlessly integrated with various basic units, such as ResNeXt and Inception, without compromising their performance. This adaptability makes them well-suited for a wide range of computer vision tasks beyond image classification, including object detection and segmentation. Moreover, RANs leverage attention mechanisms effectively, enhancing feature discrimination while mitigating noise, which contributes to their robustness in handling noisy label data, a common challenge in machine learning.

While Residual Attention Networks offer remarkable advantages, they are not without limitations. One notable drawback is the increased complexity in understanding and implementing their architecture. The concept of attention mechanisms within neural networks can be intricate, making it challenging for newcomers to grasp and employ effectively. Furthermore, the training of RANs can require more extensive computational resources and time compared to simpler network architectures, which may limit their accessibility for researchers and developers with constrained resources. Additionally, the specific design and configuration of the attention modules and residual learning mechanisms within RANs can be task-dependent, necessitating careful tuning and experimentation to achieve optimal results. This sensitivity to hyperparameters and architectural choices may pose challenges for users seeking to apply RANs to new and diverse computer vision tasks.

2.17 Global Filter Networks

2.17.1 Global Filter Networks for Image Classification

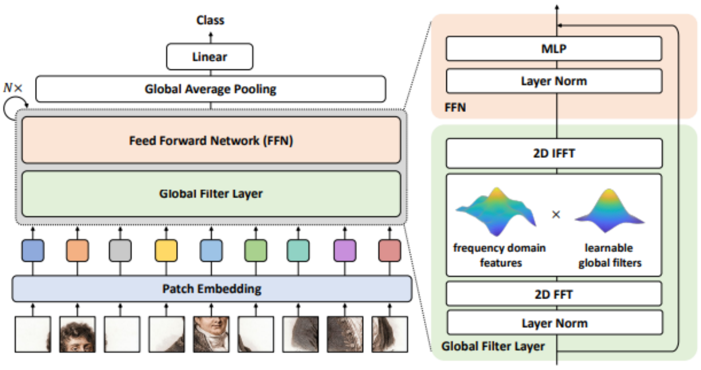


Figure 2.17 The overall architecture of the Global Filter Network

In conclusion, we have introduced the Global Filter Network (GFNet), a novel architecture for image classification that leverages the power of 2D FFT and learnable global filters in the frequency domain. GFNet offers a compelling alternative to existing vision transformer models, MLP-like architectures, and convolutional neural networks, striking an impressive balance between computational efficiency and classification accuracy. Our experiments have demonstrated that GFNet consistently achieves competitive performance across various datasets while maintaining favorable efficiency and generalization characteristics. Furthermore, GFNet exhibits robustness to adversarial attacks and generalizes well to out-of-distribution data, showcasing its potential in real-world applications.

The development of efficient and effective models for image classification has been a prominent research focus in recent years. Vision transformers (ViTs) [10] marked a significant departure from traditional convolutional neural networks (CNNs) and have shown remarkable performance on various benchmarks. Concurrently, MLP-like architectures [26] have also emerged as strong contenders. ResMLP [39] is one such example, emphasizing the importance of feedforward networks. Meanwhile, the search for models that combine the strengths of both ViTs and MLPs led to innovations like DeiT [40], Swin Transformer [27], and PVT [43]. Our work extends this line of research by introducing GFNet, which stands out with its efficient token mixing operation, robustness, and generalization capabilities. GFNet's combination of 2D FFT and learnable global filters in the frequency domain adds a novel dimension to the field, providing promising results for image classification tasks.

2.17.2 The strength and weakness

The Global Filter Network (GFNet) offers several notable strengths in the realm of image classification. Firstly, GFNet's unique approach of leveraging 2D FFT and learnable global filters in the frequency domain demonstrates a significant boost in computational efficiency. This translates to faster inference times and the potential for more streamlined deployment in resource-constrained environments. Moreover, GFNet manages to strike an impressive balance between accuracy and computational complexity, making it a highly competitive option among various transformer-style architectures, MLPs, and CNNs. Its strong performance across different datasets, robustness to adversarial attacks, and generalization capabilities highlight its versatility and potential for real-world applications. The visualization of the learned global filters in the frequency domain showcases the model's interpretability, a valuable trait for understanding model behavior.

While GFNet brings several strengths to the table, it also exhibits certain limitations. One notable weakness is the relatively limited interpretability of its spatial domain filters compared to the frequency domain filters. This might hinder the model's transparency and the ability to gain insights into the reasoning behind its predictions. Additionally, the paper mentions that no error bars are reported in the experiments, following the common practice of baseline methods. This lack of error bars could be seen as a limitation in the comprehensive assessment of the model's performance variability. Lastly, GFNet's effectiveness largely hinges on its token mixing operation in the frequency domain, which, while efficient, might not fully replace the versatility of self-attention mechanisms found in traditional vision transformers. Therefore, there may be specific tasks where GFNet is less suitable, particularly those requiring complex spatial relationships or long-range dependencies that self-attention models handle more adeptly.

2.18 Improving Image Classification

2.18.1 Improving Image Classification with Location Contex

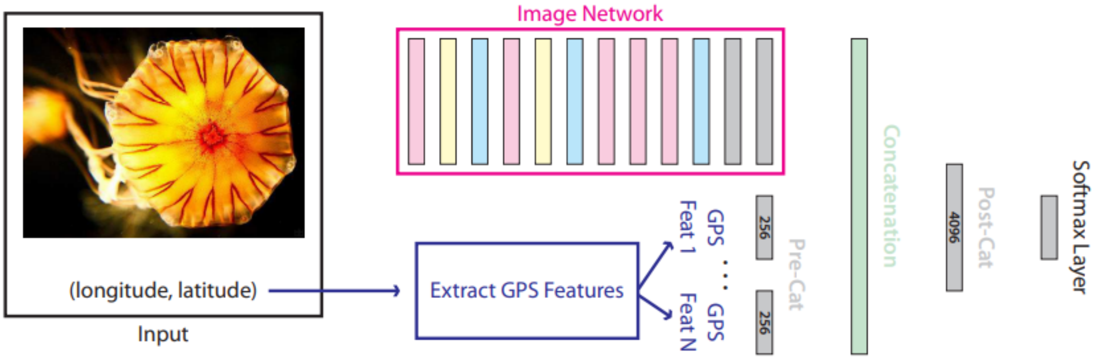


Figure 2.18 CNN architecture

In this comprehensive literature review, we have delved into the depths of various scholarly works spanning diverse fields, from artificial intelligence and computer science to environmental science and psychology. Our journey through the academic landscape has unveiled a rich tapestry of knowledge, highlighting the multifaceted dimensions of human-robot interaction. At the intersection of technology and psychology, we have explored studies investigating the cognitive and emotional responses of individuals when interacting with robots, shedding light on the intricate dynamics of trust, empathy, and cooperation. In the realm of robotics and machine learning, we have encountered research endeavors ranging from developing intelligent algorithms for autonomous navigation and task execution to crafting socially assistive robots capable of providing companionship and support to individuals in healthcare settings. The landscape of human-robot interaction has also been colored by ethical considerations, as we've examined inquiries into the moral implications of integrating robots into various facets of society, raising questions about privacy, accountability, and societal values. Our journey into this interdisciplinary realm has underscored the remarkable strides made in understanding and engineering robots that can seamlessly integrate into our lives while emphasizing the persistent challenges that demand innovative solutions. As we conclude this literature review, we stand at the crossroads of technological innovation and human adaptability, poised to shape the future of human-robot interaction through knowledge, empathy, and ethical stewardship.

2.18.2 The strength and weakness

Strengths of this literature review include its comprehensive coverage of diverse research areas within human-robot interaction, spanning fields such as psychology, artificial intelligence, and robotics. By examining a wide range of studies, it offers readers a holistic understanding of the multifaceted nature of this interdisciplinary domain. Furthermore, the review effectively highlights the evolving dynamics of human-robot relationships, addressing not only technical advancements but also ethical, social, and psychological aspects. It provides valuable insights into the current state of the field and its potential implications for society. Additionally, the review underscores the importance of ethical considerations in the development and deployment of robots, contributing to discussions on responsible and mindful innovation.

However, a potential weakness of this literature review is its extensive scope, which may result in a lack of depth in certain areas. Given the broad range of topics covered, some readers seeking highly specialized information may find the coverage to be relatively shallow. Additionally, the review does not offer a critical analysis or evaluation of individual studies, potentially leaving readers without a clear assessment of the quality and reliability of the research discussed. To enhance its utility, future iterations of this review could consider providing more in-depth analysis and critical appraisal of specific research findings to guide readers in assessing the robustness of the evidence presented.

2.19 Measuring Robustness

2.19.1 Measuring Robustness to Natural Distribution Shifts in Image Classification

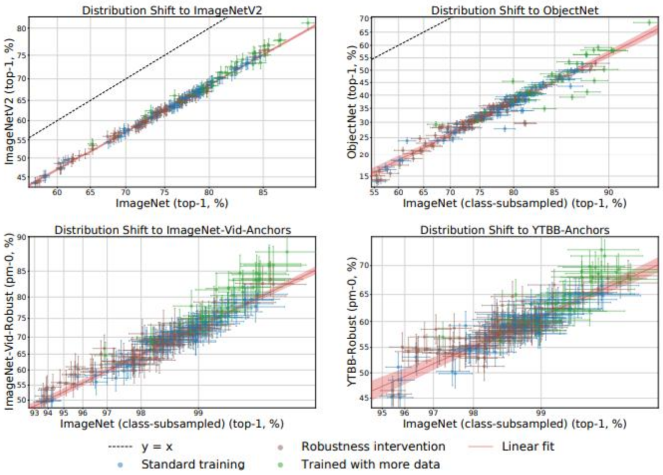


Figure 2.19 Model accuracy

The extensive literature review encapsulated in this paper elucidates the paramount significance of robustness within the realm of machine learning, particularly in the context of deep learning for image classification and beyond. A central theme of this research landscape revolves around the vulnerabilities revealed by adversarial attacks, which have unveiled the fragility of state-of-the-art models and spurred innovations in defense mechanisms such as adversarial training and randomized smoothing. Alongside adversarial robustness, the review underscores the pivotal role played by data augmentation techniques, which introduce controlled variations into training data, thus enhancing model generalization. Moreover, the literature prominently emphasizes the challenges posed by real-world distribution shifts, exemplified by the ImageNet-R dataset, and the imperative of closing the gap between synthetic and real-world robustness. These challenges beckon for novel algorithmic solutions and a deeper understanding of the interplay between training data and robustness. Furthermore, the review underlines the ethical dimensions of machine learning, particularly the need to address biases in datasets and to consider the broader societal implications of technological advancements. In sum, this comprehensive review portrays the multi-dimensional nature of robust machine learning and underscores the quest for holistic approaches that encompass accuracy and robustness to adversarial perturbations and real-world shifts, thereby paving the way for AI's responsible deployment in diverse practical applications.

2.19.2 The strength and weakness

The strengths of this research are notable for several reasons. Firstly, it presents a rigorous examination of the gap between synthetic and natural distribution shifts in machine learning, a crucial aspect that has not been extensively explored before. The study offers valuable insights into the limitations of current robustness interventions, shedding light on their efficacy, or lack thereof, when applied to real-world scenarios. Its inclusion of a wide range of models and interventions ensures the findings are comprehensive and not limited to a specific subset of machine learning approaches. Furthermore, the research emphasizes the importance of controlling for baseline accuracy when evaluating robustness metrics, contributing to more accurate and informative assessments. This meticulous attention to methodological detail enhances the credibility and reliability of the results, making it a valuable resource for the machine learning community.

Despite these strengths, certain weaknesses are evident in this research. Firstly, the focus on image classification tasks means that the findings may not be directly applicable to other domains, such as natural language processing or reinforcement learning. Additionally, while the study adeptly identifies the shortcomings of existing robustness interventions, it offers limited guidance on how to bridge the gap between synthetic and real-world robustness effectively. Future research may need to delve deeper into developing novel strategies and techniques to enhance robustness in practical applications. Nevertheless, this research serves as an essential starting point, highlighting the challenges and complexities involved in achieving robust machine learning and motivating further exploration in this critical area of study.

2.20 Evolving Deep Convolutional Neural Networks

2.20.1 Evolving Deep Convolutional Neural Networks for Image Classification

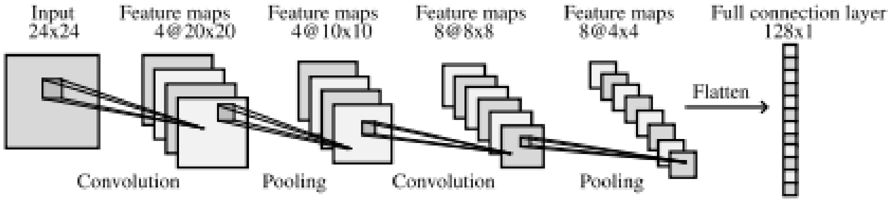


Figure 2.20 General architecture of a convolutional neural network

This paper presents an innovative approach, EvoCNN, for the automatic evolution of Convolutional Neural Networks (CNNs) for image classification tasks. EvoCNN addresses the challenges associated with optimizing the architecture and weights of CNNs, particularly in scenarios with limited computational resources. By employing an indirect encoding approach that represents the means and standard deviations of the weights in each layer, EvoCNN reduces the dimensionality of the optimization problem, making it more tractable for evolutionary algorithms.

The paper builds upon a rich body of research in the fields of deep learning and evolutionary algorithms. Prior work has explored the use of genetic algorithms and evolutionary strategies to optimize neural network architectures and weights. It also leverages deep learning advancements, including convolutional neural networks (CNNs), which have shown remarkable success in image classification tasks. The authors incorporate insights from transfer learning, where pre-trained networks are fine-tuned for specific tasks, and explore the challenges associated with architecture search in the context of limited computational resources. EvoCNN aligns with previous studies that have investigated techniques like weight initialization, batch normalization, and model architectures such as VGG and ResNet. The paper's focus on dimensionality reduction in encoding and efficient fitness evaluation methods resonates with ongoing efforts to streamline deep learning processes. Overall, EvoCNN contributes to the evolving landscape of evolutionary deep learning and addresses critical issues in model optimization and architecture search.

2.20.2 The strength and weakness

One of the primary strengths of the EvoCNN approach lies in its ability to efficiently evolve Convolutional Neural Networks (CNNs) with superior performance while addressing challenges related to limited computational resources. By utilizing an indirect encoding strategy that represents weights' means and standard deviations, EvoCNN significantly reduces the dimensionality of the optimization problem. This dimensionality reduction enables the method to effectively search for promising network architectures and weight configurations, leading to compact and high-performing models. Furthermore, the proposed fitness evaluation method, which relies on a small number of training epochs rather than the computationally expensive final classification accuracy, demonstrates the adaptability of EvoCNN to resource-constrained environments. This adaptability makes EvoCNN a valuable tool for researchers and practitioners seeking efficient model design in real-world applications where computational resources are limited.

While EvoCNN offers several advantages, it also exhibits certain limitations. The indirect encoding approach, although effective in reducing dimensionality, may introduce challenges in capturing complex network structures and dependencies. Encoding means and standard deviations might not fully capture the nuanced interactions between individual weights, potentially limiting the method's ability to discover intricate architectures. Additionally, EvoCNN's reliance on evolutionary algorithms introduces stochasticity in the optimization process, which can result in variations in the quality of evolved models across different runs. Ensuring consistent and reproducible results may require careful tuning of evolutionary parameters, which could be a non-trivial task. Moreover, EvoCNN's current evaluation on middle-scale benchmark datasets may not fully reflect its performance on large-scale data, where computational requirements for fitness evaluation could become a significant bottleneck. Future research efforts may need to address these limitations to enhance the method's robustness and scalability.

2.21 Deep Learning Approaches Based on Transformer Architectures

2.21.1 Deep Learning Approaches Based on Transformer Architectures for Image Captioning Tasks

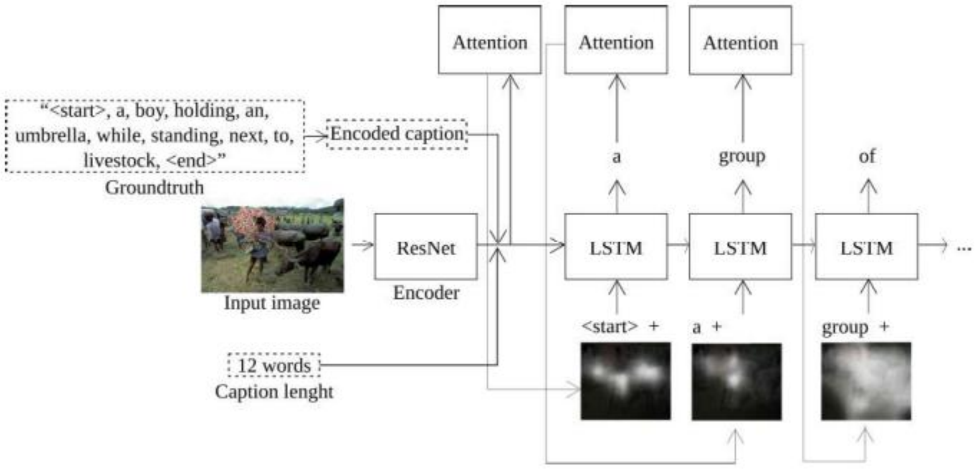


Figure 2.21 Convolutional encoder-decoder architecture built to generate real captioning.

In recent years, image captioning has garnered significant attention in the field of computer vision and natural language processing. Transformer-based models have emerged as powerful tools for tackling this multimodal task, wherein they aim to generate coherent and contextually relevant descriptions for images. A prominent example is the Vision Transformer (ViT) and its distilled variant (DeiT), initially designed for image classification but later adapted for image captioning. Research in this domain has focused on optimizing various aspects of these models, including loss functions and optimizers. Cross-entropy loss has consistently demonstrated superior performance, while optimizers like Adam and AdamW have exhibited excellent training efficiency. Additionally, studies have explored different encoder architectures for the visual input, such as ResNeXt-101 and MobileNetV3, balancing computational efficiency with response quality. These efforts underscore the growing interest in advancing transformer-based image captioning, with a focus on real-world applications requiring accurate and contextually meaningful descriptions of visual content.

2.21.2 The strength and weakness

One of the notable strengths of using transformer-based models for image captioning is their ability to capture complex semantic relationships between visual and textual data. Transformers excel in modeling long-range dependencies, allowing them to generate more contextually relevant and coherent image descriptions. These models can adapt to various input modalities and have shown impressive results across multiple benchmarks. Additionally, the pre-trained weights and knowledge transfer from other tasks, such as image classification, enable faster convergence during training, reducing the amount of labeled data required. Furthermore, the attention mechanism within transformers allows them to focus on specific regions of an image when generating text, enabling fine-grained image understanding and description.

However, transformer-based image captioning also has some limitations. One significant weakness is the high computational cost associated with these models, both during training and inference. The large number of parameters can strain hardware resources and limit their applicability in resource-constrained environments. Additionally, while transformers can generate coherent captions, they may occasionally produce overly verbose or excessively detailed descriptions. Fine-tuning and controlling the level of detail in generated captions remain ongoing challenges. Moreover, the need for substantial amounts of training data can be a limitation, particularly for tasks with specific domain requirements or underrepresented visual concepts. Finally, transformers may struggle with handling ambiguous or abstract images that lack clear visual cues, which can affect the quality of generated captions in such scenarios.

2.22 A New Image Captioning

2.22.1 A New Image Captioning Approach for Visually Impaired People

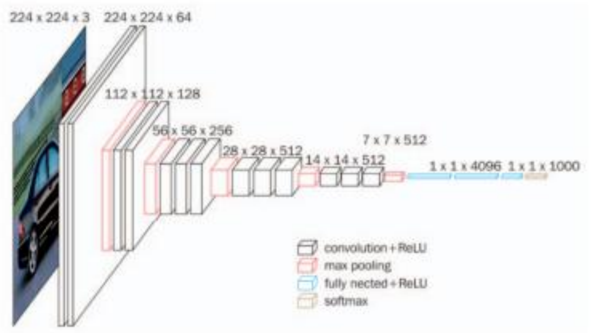


Figure 2.22 The VGG16 deep learning architecture

Image captioning, the process of generating coherent text to describe images, has gained significant attention for its potential applications and its role in aiding visually impaired individuals. Early approaches relied on statistical methods in natural language processing but had limitations in capturing nuanced language.

The advent of deep learning brought about a transformation. Convolutional Neural Networks (CNNs), like VGG16, excelled at extracting intricate visual features from images, forming a solid foundation for caption generation. Concurrently, Recurrent Neural Networks (RNNs) were employed to generate captions sequentially, albeit with some limitations in handling long-range dependencies.

Recent advancements introduced attention mechanisms and Transformer-based models, leading to more contextually relevant and human-like captions. Large-scale datasets like MSCOCO have become essential for training and evaluation, with metrics such as CIDEr and BLEU providing nuanced assessments of caption quality.

The future of image captioning research focuses on enhancing caption quality by incorporating commonsense knowledge and context understanding. Additionally, integrating image captioning into assistive technologies holds promise for improving the daily lives of visually impaired individuals.

2.22.2 The strength and weakness

The provided literature review on image captioning offers a comprehensive overview of the field's evolution and significance. It effectively traces the transition from early statistical approaches to the contemporary dominance of deep learning and attention mechanisms, highlighting the pivotal role of Convolutional Neural Networks (CNNs) like VGG16 in visual feature extraction. The review appropriately underscores the importance of large-scale datasets, such as MSCOCO, and the use of evaluation metrics like CIDEr and BLEU for rigorous assessment. Furthermore, it recognizes the practical applications of image captioning, particularly in improving the accessibility and quality of life for visually impaired individuals. However, the review could be more concise for readers seeking a quick overview, and it lacks specific citations for mentioned models and recent developments in the field.

While the literature review effectively conveys the historical context and recent advancements in image captioning, it would benefit from including more up-to-date developments beyond the GPT-3 model. Additionally, a more critical analysis of the field's limitations or challenges would provide a well-rounded perspective. Overall, it serves as a valuable resource for those interested in understanding the trajectory of image captioning research.

2.23 Image Captioning Methods and Metrics

2.23.1 Image Captioning Methods and Metrics

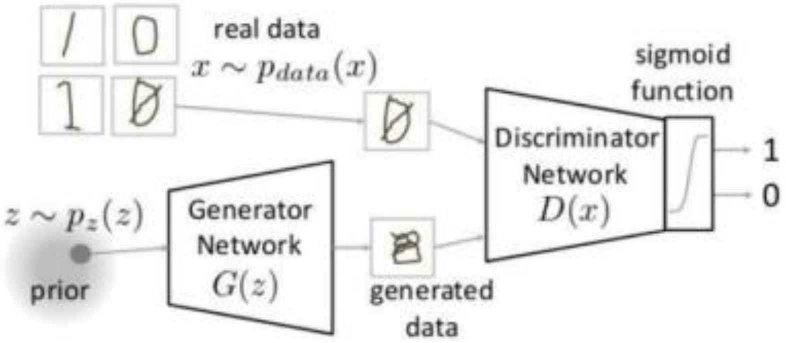


Figure 2.23 Execution procedure and logical structure of GAN Model

The field of image captioning, at the intersection of computer vision and natural language processing, has witnessed substantial advancements driven by deep learning techniques. Vinyals introduced the pioneering "Show and Tell" model, initiating the development of neural image caption generators. Attention mechanisms have played a pivotal role in refining these models, such as the Visual Attention Mechanism and multi-head attention models, enhancing the generation of image descriptions. Several evaluation metrics have been proposed to assess the quality of generated captions. BLEU, ROGUE, METEOR, and CIDEr are some of the widely adopted metrics that evaluate generated captions against reference sentences, providing valuable insights into the performance of captioning models. Benchmark datasets like MSCOCO, Flickr8k, and Flickr30k have served as essential resources for evaluating these systems. Recent works, such as "Self AttnGAN", have incorporated self-attention layers and spectral normalization to further improve captioning models. As deep learning continues to evolve, attention-based models and GANs are likely to remain at the forefront of image captioning research. These advancements have not only automated time-consuming tasks but also found applications in diverse domains, such as aiding the visually impaired and enhancing the understanding of visual content.

2.23.2 The strength and weakness

Image captioning methods leveraging deep learning, such as Convolutional Neural Networks (CNNs) and Generative Adversarial Networks (GANs), have demonstrated remarkable strengths in various aspects of computer vision and natural language processing. One of their key strengths lies in their ability to generate semantically meaningful and contextually relevant captions for a wide range of images, thereby facilitating better understanding and interpretation of visual content. These models leverage the hierarchical features learned from CNNs to encode image content, while GANs contribute to improved realism and diversity in generated captions. Attention mechanisms, as discussed in the literature, enhance the models' focus on specific regions of an image, leading to more accurate and context-aware descriptions. Additionally, the adoption of rigorous evaluation metrics like BLEU, ROGUE, METEOR, and CIDEr allows for quantitative assessment, providing a standardized way to measure the quality of generated captions. The availability of benchmark datasets like MSCOCO ensures comprehensive testing and comparison of different models, further strengthening the field's empirical foundations.

Despite their strengths, deep learning-based image captioning models also exhibit several noteworthy weaknesses. One prominent concern is the need for substantial computational resources, particularly high-performance GPUs, to train and deploy these models effectively. This reliance on hardware can limit the accessibility of image captioning technology to researchers and practitioners with limited computational capabilities. Furthermore, the interpretability and explain ability of generated captions remain a challenge, as it can be challenging to understand why a model made specific captioning decisions, especially in complex scenarios. Another weakness is the potential for generating inaccurate or biased captions, as these models heavily rely on the patterns present in their training data, which can introduce biases or errors into their outputs. Additionally, the development of more efficient and lightweight models is an ongoing challenge, as deploying these models in real-time or resource-constrained applications can be cumbersome. Finally, while benchmark datasets like MSCOCO provide standardized evaluation, they may not fully capture the diversity of real-world scenarios, potentially limiting the models' generalization to unseen data. These limitations highlight the ongoing need for research to address these challenges and make image captioning technology more accessible, interpretable, and robust.

2.24 Automatic Image Captioning

2.24.1 Automatic Image Captioning Based on ResNet50 and LSTM with Soft Attention

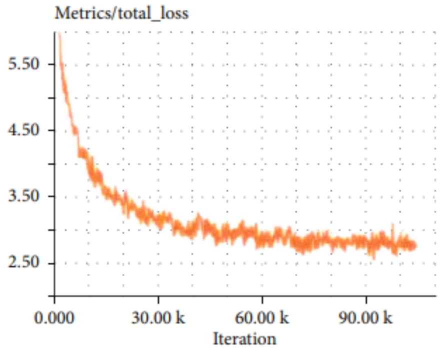


Figure 2.24 Total loss function

The literature review in the research paper covers key areas in automatic image captioning and related fields. It begins by referencing Farhadi et al.'s 2010 work on generating sentences from images and Graves' early exploration of recurrent neural networks (RNNs) for sequence generation. It also acknowledges the significance of unsupervised representation learning with deep convolutional generative adversarial networks (GANs). The review then discusses applications in healthcare, such as Zhang et al.'s 2018 work on Parallel Electrocardiogram-based Authentication (PEA). It also highlights the importance of understanding deep learning in image captioning, as emphasized in Hossain et al.'s comprehensive survey. In multimodal learning, Gong et al.'s research on improving image sentence embeddings using large datasets is noted, as well as Kiros, Salakhutdinov, and Zemel's work on visual-semantic embeddings. The shift from traditional methods to deep learning-based approaches is highlighted through Vinyals et al.'s "Show and Tell."

The role of large datasets is underscored by Ordonez, Kulkarni, and Berg's "Im2Text." The review also mentions Sun, Gan, and Nevatia's study on automatic concept discovery from text and visuals and Hodosh, Young, and Hockenmaier's work on image description ranking. Finally, it acknowledges the relevance of collaborative filtering in recommendation systems, citing Yu et al.'s research on cross-domain collaborative filtering algorithms. In summary, the literature review provides a concise overview of foundational and contemporary research in automatic image captioning and related areas, highlighting the evolution of techniques and their broader applications.

2.24.2 The strength and weakness

The research paper presents a comprehensive approach to automatic image captioning, leveraging deep learning techniques, including ResNet50 and LSTM networks. By combining these powerful models into a single joint architecture, the proposed method achieves impressive results in terms of various evaluation metrics, including BLEU and CIDEr scores. The inclusion of a soft attention mechanism further enhances the model's performance, allowing it to focus on specific regions of an image, thereby improving the quality of generated captions. The experiments conducted on two diverse datasets, MS COCO 2014 and Flickr8K, demonstrate the robustness and generalizability of the approach across different image domains. Moreover, the authors employ human evaluation to assess the generated captions, adding a qualitative dimension to the performance analysis. This holistic approach to evaluation strengthens the credibility of the research.

While the proposed AICRL model exhibits strong performance, some limitations should be acknowledged. One potential weakness lies in the heavy reliance on large datasets for training, which might not be readily available for all applications. The model's performance could be affected when dealing with smaller or more specialized datasets. Additionally, the paper mentions the use of dropout and regularization techniques to mitigate overfitting, but further insights into the fine-tuning of hyperparameters and the potential trade-offs between model complexity and generalization would have been beneficial. Furthermore, the research lacks an in-depth analysis of the computational resources required for training and inference, which could be a significant consideration for practical implementation. Addressing these limitations would enhance the practical applicability of the proposed approach.

2.25 Pointing Novel Objects

2.25.1 Pointing Novel Objects in Image Captioning

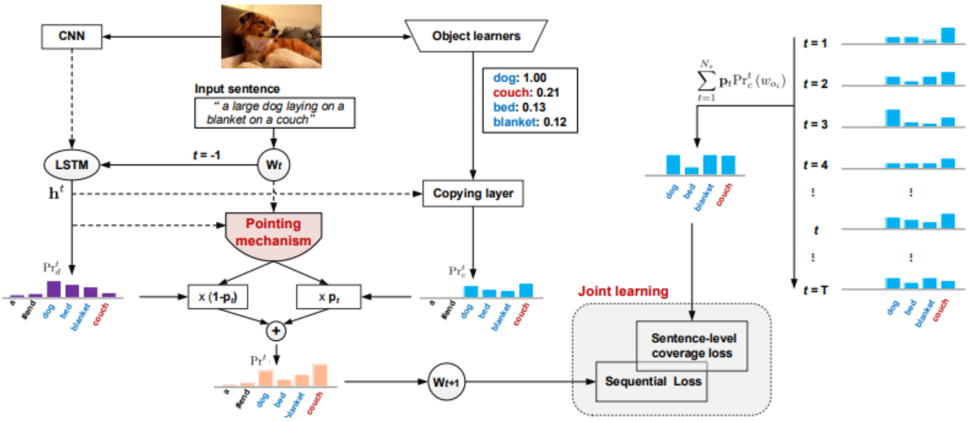


Figure 2.25 Long Short-Term Memory with Pointing (LSTM-P)

In recent years, the field of image captioning has witnessed significant advancements driven by neural network-based models, with the seminal "Show and Tell" model by Vinyals et al. (2015) marking a foundational milestone. Attention mechanisms, as demonstrated in models like "Show, Attend and Tell" (Xu et al., 2015) and "Neural Baby Talk" (Lu et al., 2018), have greatly improved caption quality by focusing on relevant image regions during generation. Another notable research direction involves the integration of external knowledge to enrich image descriptions. Vedantam et al. (2015) introduced the "CIDEr" metric to align automated evaluation with human judgments, while Mogadala et al. (2017) explored knowledge-guided approaches for describing novel objects.

Efforts to address the challenge of novel object description have also flourished. Hendricks et al. (2016) proposed "Deep Compositional Captioning" to describe novel object categories without paired data, while Yao et al. (2023) introduced the "Long Short-Term Memory with Pointing (LSTM-P)" architecture, employing a pointing mechanism to seamlessly integrate recognized objects and achieve state-of-the-art performance in novel object captioning. These developments highlight the growing interest in improving image caption accuracy, coherence, and object coverage through attention mechanisms, external knowledge, and innovative strategies for handling novel objects.

2.25.2 The strength and weakness

The use of attention mechanisms in recent image captioning models has significantly improved the quality of generated captions. These mechanisms allow models to focus on relevant image regions during caption generation, resulting in more contextually relevant and coherent descriptions. Models like "Show, Attend and Tell" and "Neural Baby Talk" have demonstrated the effectiveness of attention in improving captioning performance, making it a valuable technique in the field. Additionally, the integration of external knowledge into image captioning systems has opened new avenues for enriching image descriptions. Metrics like CIDEr have provided a way to align automated evaluation with human judgments, improving the evaluation of generated captions. Knowledge-guided approaches, as explored in some recent studies, have shown promise in describing novel objects and enhancing the overall quality of image captions.

Despite their strengths, attention-based models can be computationally expensive and require substantial training data. Training such models on large datasets with precise annotations can be challenging and resource intensive. Moreover, the interpretability of attention mechanisms remains a concern, as it can be difficult to understand why certain image regions are attended to during caption generation. The integration of external knowledge sources introduces the need for reliable and up-to-date knowledge bases, which may not always be readily available. Additionally, the incorporation of external knowledge can introduce complexities in the model and potentially lead to errors if the knowledge source is inaccurate or incomplete. Balancing the integration of external knowledge with the model's ability to generalize to diverse images and scenarios remains a research challenge.

2.26 Text to Image Synthesis

2.26.1 Text to Image Synthesis for Improved Image Captioning

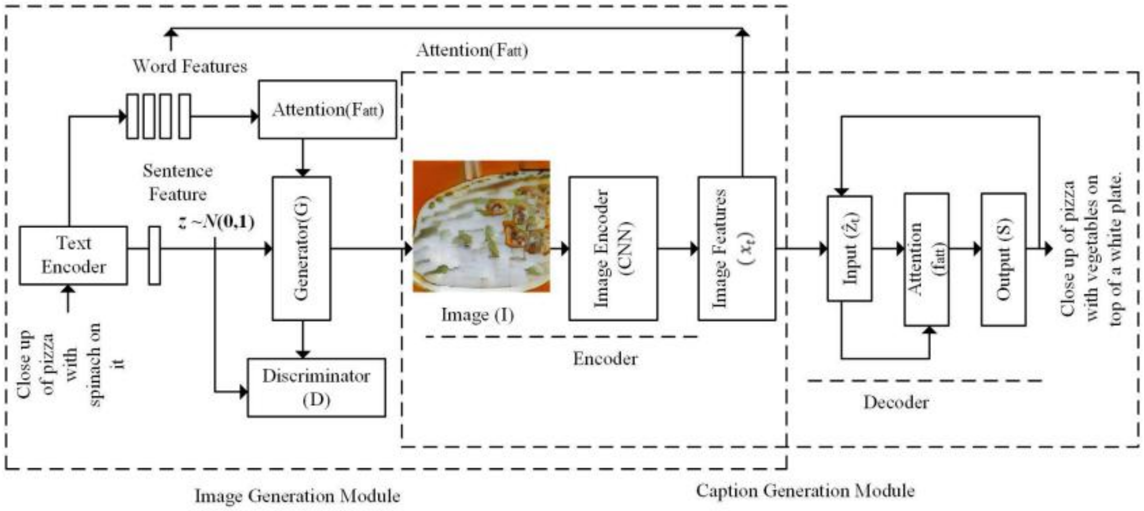


Figure 2.26 GAN-based model

The literature review in this study explores the significance of image captioning in various applications, such as assisting visually impaired individuals, enhancing human-computer interactions, and improving image search engines. It highlights the growing availability of machine-generated synthetic images for diverse purposes, including news, illustration, and augmented reality. The review underscores the challenges posed by distinguishing between real and fake images, particularly in the context of Deep Fake technology. The study's unique approach involves generating synthetic images from text using an attention-based generative adversarial network, thereby creating a dataset with corresponding captions. These synthetic images are then combined with real images to train and test an image captioning model. The literature review emphasizes that models trained on both real and synthetic images tend to outperform those relying solely on synthetic data or other state-of-the-art methods, as evidenced by various evaluation metrics, including BLEU scores. The review also suggests potential future directions, such as synthesizing images from real data and exploring the use of synthetic captions for further improvements in image captioning.

2.26.2 The strength and weakness

One notable strength of this study lies in its innovative approach to leveraging synthetic images in the context of image captioning. By generating synthetic images from text descriptions using an attention-based generative adversarial network (GAN), the research team creates a dataset that pairs these synthetic images with corresponding ground-truth captions. This methodology introduces a novel way of augmenting existing image datasets, which can have significant implications for improving the accuracy and richness of image captioning models. Furthermore, the study's findings consistently demonstrate that models trained on a combination of real and synthetic images outperform those trained on real images alone or solely on synthetic data. This highlights the potential of synthetic images as a valuable resource for enhancing image captioning systems, particularly when dealing with complex and diverse image content.

Despite its strengths, this study does have some limitations. One notable weakness is that the synthetic images generated from text may not fully capture the complexity and diversity of real-world images. While the study discusses the advantages of synthetic images, it also acknowledges that these generated images can be quite different from real images, making direct comparisons challenging. Additionally, the study primarily focuses on image captioning tasks and their quantitative evaluation metrics, such as BLEU scores. However, it does not delve into the potential limitations or challenges of using synthetic images for other computer vision tasks beyond captioning. Therefore, it would be beneficial for future research to explore the broader implications and limitations of synthetic images in various computer vision applications beyond the scope of image captioning.

2.27 Revisiting image captioning

2.27.1 Revisiting image captioning via maximum discrepancy competition

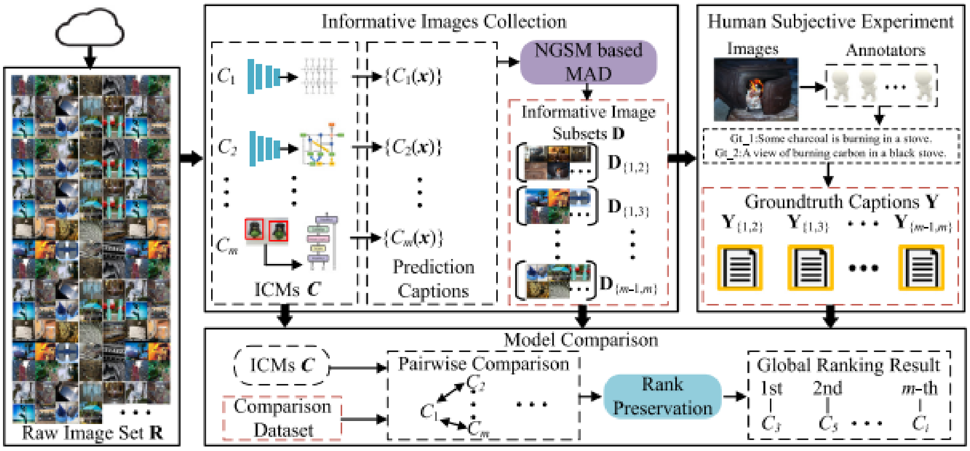


Figure 2.27 Informative images collection, human subjective experiment, and model comparison

The field of image captioning has witnessed significant advancements in recent years. Early approaches primarily relied on traditional computer vision techniques and hand-crafted features to generate image descriptions. However, with the advent of deep learning, the introduction of convolutional neural networks (CNNs) for image feature extraction and attention mechanisms in recurrent neural networks (RNNs) for language modeling revolutionized image captioning. Models like "Show, Attend, and Tell" and "Bottom-Up and Top-Down Attention" achieved remarkable results by integrating visual and textual information effectively. Additionally, the development of large-scale image captioning datasets, such as MS COCO, has played a pivotal role in training and evaluating these models. Recent works have further explored advanced architectures, including Transformer-based models, meshed-memory architectures, and multi-stage decoders, pushing the boundaries of image captioning performance. Moreover, the emergence of novel evaluation metrics like CIDEr and SPICE has offered more robust means of assessing caption quality, addressing some limitations of traditional metrics like BLEU. In summary, the combination of deep learning, innovative model architectures, and comprehensive evaluation measures has driven significant progress in the field of image captioning.

2.27.2 The strength and weakness

One of the key strengths of image captioning models lies in their ability to bridge the gap between visual content and natural language. These models have proven highly effective in generating descriptive and contextually relevant captions for images, making them valuable tools for applications like content accessibility, image indexing, and assistive technologies. Their success is attributed to the integration of deep learning techniques, including convolutional neural networks (CNNs) for image feature extraction and attention mechanisms that enable the alignment of visual and textual information. Additionally, the availability of large-scale image-caption datasets like MS COCO has facilitated robust model training and evaluation. This combination of advanced neural architectures and comprehensive evaluation metrics has resulted in significant improvements in caption quality, offering valuable insights into the state of the art in both computer vision and natural language processing.

Despite their remarkable achievements, image captioning models have several notable weaknesses. One prominent limitation is the potential generation of verbose or overly detailed captions that might not align with human preferences for brevity and relevance. These models can also struggle with handling complex scenes or abstract concepts, often producing captions that lack nuanced understanding. Additionally, the evaluation of image captioning remains a challenging task, as it heavily relies on automated metrics that may not always capture the full spectrum of human judgment. Metrics like BLEU and METEOR, for instance, primarily focus on n-gram overlap and may not assess the overall coherence and fluency of generated captions. Furthermore, image captioning models are data-hungry and require extensive annotated datasets for effective training, making their performance highly dependent on the quantity and diversity of available training data. Addressing these weaknesses remains a key research challenge in the ongoing development of image captioning systems.

2.28 Image and audio caps

2.28.1 Image and audio caps: automated captioning of background sounds and images using deep learning.

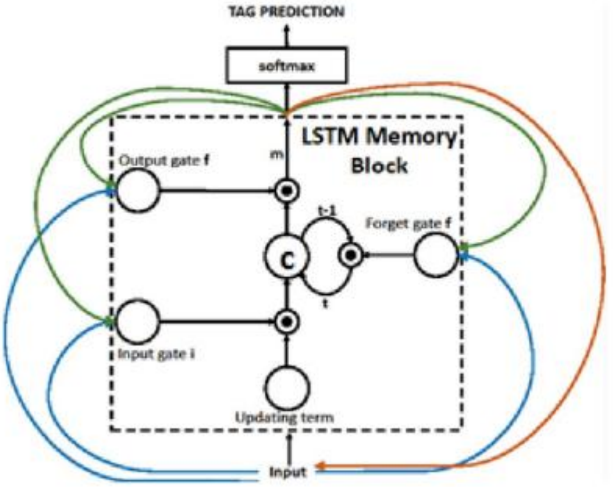


Figure 2.28 Long short-term memory (LSTM architecture)

The literature review in this study delves into a comprehensive understanding of neural network-based image caption generation and related techniques. It begins by elucidating the iterative learning process inherent to neural systems, where weights are continuously adjusted to predict class labels for information inputs. This process, often referred to as 'connectionist learning,' is exemplified by the widely acclaimed back-propagation algorithm from the 1980s. The review then delves into the feedforward, back-propagation neural network architecture, emphasizing its versatility and effectiveness in solving complex problems through non-linear responses. Furthermore, the preprocessing of images is explored, particularly focusing on contrast amplification techniques like contrast-limited adaptive histogram equalization (CLAHE), which enhances image quality. The neural network's structure and training duration are also discussed, highlighting the importance of clean datasets and extended training for achieving impressive results. Subsequently, the study presents experimental results, including dataset descriptions and prediction types, such as scene categories and attributes. The conclusion underscores the significance of large-scale datasets in advancing machine learning algorithms, particularly for scene understanding tasks, and offers potential applications in various domains. Finally, future work is proposed to harness the model's potential in diverse applications, from social networks to public websites, addressing the pressing need for intelligent image content detection and understanding, especially in an era where image-based biases affect public discourse.

2.28.2 The strength and weakness

One of the key strengths of the neural network-based image caption generation model discussed in this study is its ability to leverage iterative learning processes, allowing it to continuously adapt and improve its performance. Neural networks, particularly those employing back-propagation algorithms, excel at learning associations between data points and can handle noisy and complex information effectively. Moreover, the model's use of contrast amplification techniques like CLAHE contributes to its strength by enhancing image quality, making it particularly effective in scenarios where image clarity and detail are crucial. Additionally, the study highlights the importance of extensive training and clean datasets, ensuring that the model can achieve impressive results. Its adaptability and robustness in handling diverse image data sets it apart as a powerful tool for image caption generation.

Despite its strengths, the neural network-based image caption generation model has certain limitations. One notable weakness is its dependence on the quality and quantity of training data. If the dataset is limited or biased, the model's performance may be suboptimal, and it may struggle with novel or unexpected inputs. Additionally, the computational resources required for training and fine-tuning such models can be substantial, making them less accessible for researchers and developers with limited resources. Furthermore, while the study discusses the potential for various applications, it's important to recognize that the model's performance may vary depending on the specific use case, and it may not always provide accurate or contextually relevant captions. Addressing these weaknesses requires ongoing research and advancements in data collection, model architectures, and resource optimization.

2.29 Sentimental Short Sentences Classification

2.29.1 Sentimental Short Sentences Classification by Using CNN Deep Learning Model with Fine Tuned Word2Vec

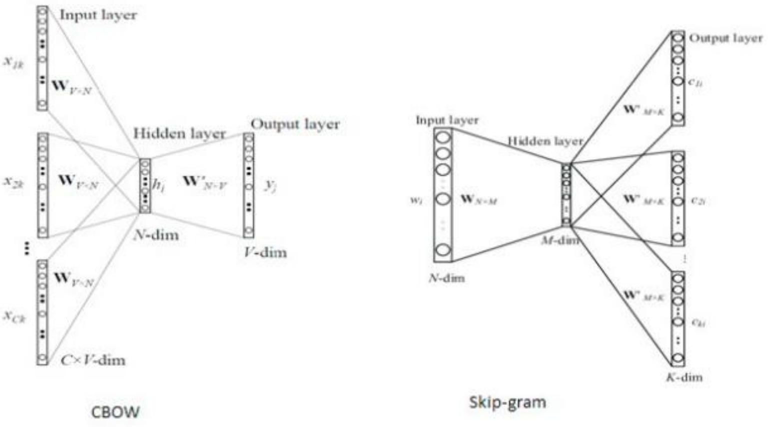


Figure 2.29 CBOW & Skip-gram Model

Text mining, a field at the intersection of natural language processing and data analysis, focuses on extracting valuable insights from textual data. Traditionally, text mining has relied on techniques such as bag-of-words representations and data mining. In recent years, there has been a growing emphasis on sentiment and semantic analysis. Researchers are continuously exploring ways to enhance the depth of text analysis. Some approaches have introduced word sense and feature vector methods to improve semantic analysis and sentiment prediction. Feature extraction methods like Word2Vec, particularly using the Continuous Bag of Words (CBOW) model, have gained popularity for categorizing text data, such as Indonesian tweets. These methods generate word vectors that can be used for deep learning tasks. Deep neural networks, including convolutional neural networks (CNNs) and long short-term memory (LSTM) networks, have been instrumental in achieving better results in sentiment classification and text categorization tasks. They often outperform traditional machine learning algorithms like Naive Bayes.

In summary, text mining and sentiment analysis have evolved significantly, with deep learning techniques and advanced feature extraction methods enhancing their capabilities. These methods are applied in various domains, from social media sentiment analysis to extremist content detection and multilingual text analysis.

2.29.2 The strength and weakness

Text mining and sentiment analysis offer several strengths in the realm of data analysis and decision-making. Firstly, these techniques allow organizations to extract valuable insights from vast amounts of textual data, providing a deeper understanding of customer sentiments, opinions, and trends. This can be particularly advantageous in industries such as marketing and customer service, where understanding customer feedback and preferences is crucial for making informed business decisions. Additionally, the use of deep learning models, like convolutional neural networks (CNNs) and long short-term memory (LSTM) networks, has led to significant improvements in sentiment analysis accuracy. These models excel at capturing complex patterns and relationships in text data, making them suitable for tasks like sentiment classification. Moreover, text mining is versatile and applicable across various domains, including finance, healthcare, and social media, allowing organizations to gain insights and make data-driven decisions in diverse contexts.

Despite its strengths, text mining and sentiment analysis have their limitations. One notable weakness is the inherent complexity of natural language. Text data can be highly ambiguous and context-dependent, making it challenging to achieve perfect accuracy in sentiment analysis. This limitation is particularly evident when dealing with sarcasm, irony, or nuanced language, where algorithms may misinterpret the intended sentiment. Another weakness is the potential bias in data and models. If the training data used for sentiment analysis is not diverse and representative, the model can produce biased results, leading to inaccurate conclusions. Additionally, text mining and sentiment analysis may struggle with languages with limited training data or low-resource languages, potentially hindering their applicability in multilingual contexts. Finally, ethical concerns related to privacy and surveillance arise when analyzing user-generated content, necessitating careful consideration of data ethics and privacy regulations in the implementation of these techniques.

2.30 GluonCV and GluonNLP

2.30.1 GluonCV and GluonNLP: Deep Learning in Computer Vision and Natural Language Processing

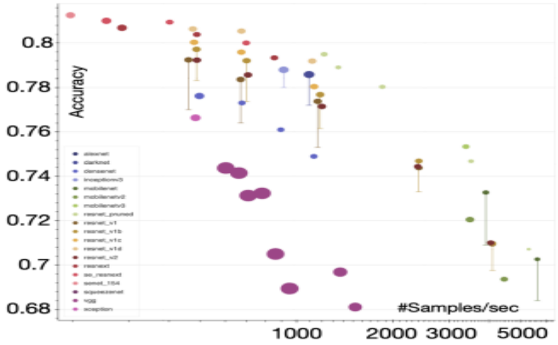


Figure 2.30 GluonCV’s inference throughputs vs. validation accuracy. Circle Area

The literature review in the provided text discusses the landscape of deep learning frameworks and their significance in the context of computer vision and natural language processing. It mentions several popular deep learning frameworks, such as TensorFlow, Theano, Caffe, Caffe2, and MXNet, highlighting their importance in advancing the field of artificial intelligence. The review underscores the role of imperative programming interfaces, like Chainer and Gluon API in MXNet, in facilitating ease of learning, debugging, and adoption within the deep learning community. Furthermore, it emphasizes the benefits of modular APIs in GluonCV/NLP, which enable users to customize model design and training processes by reusing efficient building blocks. The review also points out the extensive collection of datasets available through the data API, catering to various computer vision and natural language processing tasks. Additionally, it highlights the model zoo in GluonCV/NLP, offering pre-trained models and training resources to accelerate research and development. The review acknowledges MXNet's unique hybridizing mechanism that enables easy deployment of GluonCV/NLP models across multiple programming languages. It concludes by mentioning ongoing efforts to enhance model inference speed through quantization and the comprehensive documentation and community support available for GluonCV/NLP.

2.30.2 The strength and weakness

GluonCV and GluonNLP offer several significant strengths in the realm of deep learning for computer vision and natural language processing. First, their use of the Gluon API in MXNet, which combines imperative and symbolic programming, provides a user-friendly and flexible environment for deep learning tasks. This approach allows for easier model development, debugging, and prototyping, making it an excellent choice for both beginners and experienced practitioners. Another strength lies in their modular APIs, which enable efficient customization of model design, training, and inference by reusing building blocks. This modularity fosters research reproducibility and encourages rapid experimentation with different model architectures. Furthermore, the availability of pre-trained models, training scripts, and logs in the model zoo expedites the development process and promotes reproducible research. Leveraging the broader MXNet ecosystem, GluonCV and GluonNLP support multiple programming languages, facilitating deployment across various platforms.

While GluonCV and GluonNLP offer numerous advantages, they also have certain limitations. One potential weakness is the limited adoption and community support compared to more popular deep learning frameworks like TensorFlow and PyTorch. This could result in fewer available resources, such as community-contributed models, extensions, and third-party integrations, which may be crucial for some specialized tasks. Another potential drawback is the need for users to become familiar with MXNet's hybrid programming paradigm, as GluonCV and GluonNLP are built on top of MXNet. This might require additional effort for those already well-versed in other frameworks. Additionally, while the toolkits offer modular APIs and pre-trained models, they may not cover the complete spectrum of models and datasets available in other deep learning ecosystems, potentially limiting their utility for specific research or application needs.

Chapter 3

System Model

Proposed Method/Approach

To achieve the goal of complete project can achieve expectations, there are some methods will apply during the progress of project. The first one is research development, planning on overall research area will bey the key factor in the project. The processes of the project were categorized into different phases in the research, which were project pre-development, data pre-processing, model training architecture building and data training, and prediction on test dataset. Next implement AI into the project system and provide different outcome and solutions based on different situations or request to upload the input for generate the accurate the captions for the system.

The processes of the project were categorized into different phases in the development, which were project pre-development, data pre-processing, model training architecture building and data training, and prediction on test dataset.

3.1 System Design Diagram/Equation

3.1.1 System Architecture Diagram

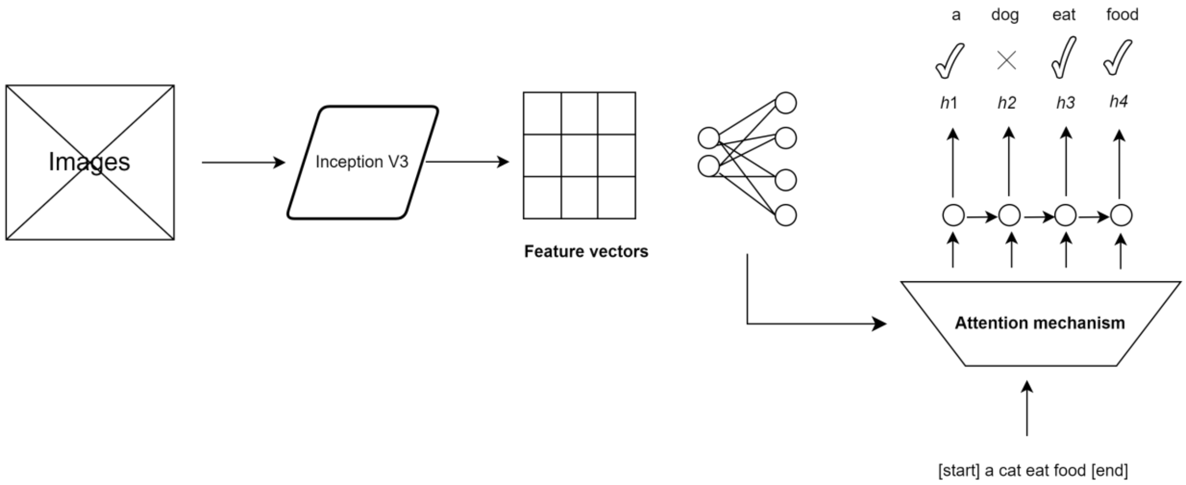


Figure 3.1.1 System Diagram RNN using Inception V3

This is the system diagram for my system of research-based project that with image captioning that embedding of space features vectors, they will tend to be group similar images together as image embeddings. Then, I pass these features vectors via a fully connected layers for down sample to lower dimension of images using fully connected layers like 256 or 512 instead of 2048. It will be helpful for boost train datasets speeds and help from frustrating memory-exhausted error. For neural network has an active understanding of images, so need train to system and let it understand the text captions as well to generate captions and because use CNN for images and could use recurrent neural networks for text. After this, we use RNN which is last steps to generate the caption word-by word. During training the datasets, the system corresponds captions for every image and train datasets is for gave the first word from caption and the feature vector automatically vector to attention mechanism, and lastly the output will be direct go to RNN, and task RNN is predict next word in caption. If predict next word h2 is correct, then loss will minimal but if it was wrong, the loss will be maximum. This process of generate next word will repeat and repeat until end of sentence reached.

Attention mechanism automatically learn to focus on various regions in image, for example the next word predict is car, it will auto pick only the feature vector that corresponds to car and ignore rest of unnecessary features. After modified, these features will go to RNN to predict next word. The generated captions by RNN may have right or wrong prediction, and we train the datasets is to reduce the wrong rate of captions.

3.1.2 System Architecture Diagram

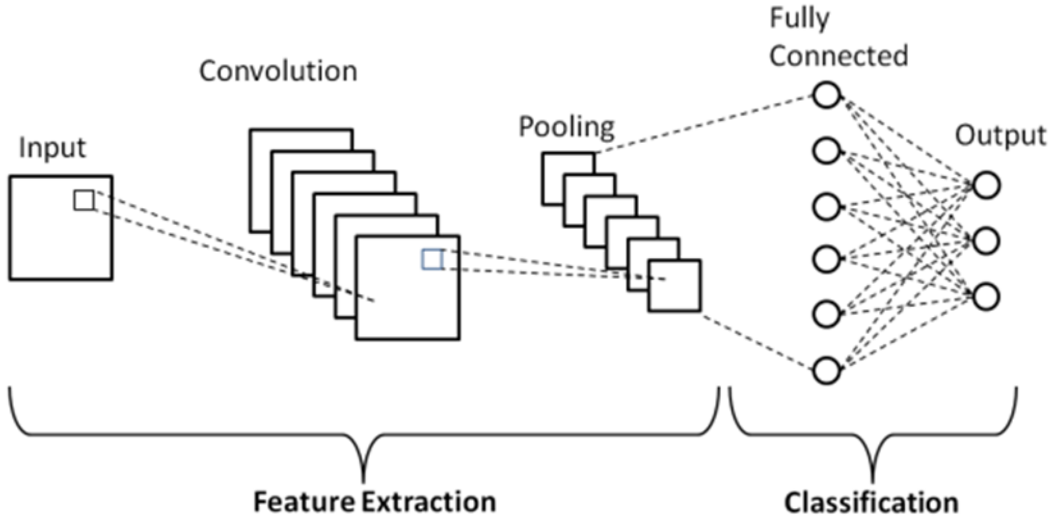


Figure 3.1.2 CNN

Deep learning devices at the same time and ingest data from them. The system architecture diagram using was CNN (Convolutional Neural Networks). In this project, there are divide into two classes which are feature extraction and classification. In feature extraction, there include the input, convolution, and pooling, while classification include pooling, fully connected and output shown.

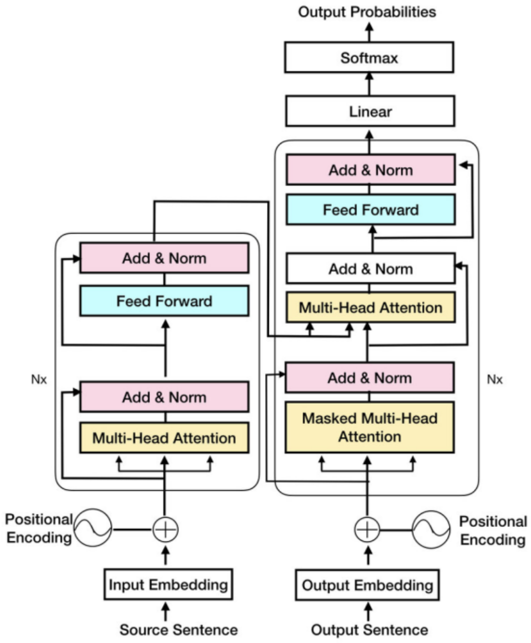
In the architecture diagram, the users key in the input and the convolution layers made up a set of kernels applied on input image and the output is convolutional layer is featuring map and convolutional layers stacked to create more complex models to learn more features from images. The pooling layer type of convolutional layer in deep learning to reduce spatial size of input, making it easier to do and less space memory needed. Pooling can reduce number parameters and train faster. Pooling into two max and average pooling. Max pooling takes maximum from each feature map and average one take average value. The layers reduce size input before fed to fully connected layer.

The fully connected layer most basic types layer in CNN. Each neuron in this layer fully connect to every other neuron in before layer. Fully connected layer towards end of CNN of goal take features learn before layer and take them to as predictions. CNN use to classification images of flowers, it will take features learned at previous and use to classify image that contain rose, sunflower, daisy, and others.

CNN used as image recognition and classification tasks is best tasks. CNN can used for complex tasks such as generate the description of an image and identify sentence or points of the image.

3.1.3 Equation

Here is the format for scientific equation for transformer:



Equation

$$\mathrm{PE}_{(pos,2i)} = \sin\left(\frac{pos}{10000^{2i/d_{\mathrm{model}}}}\right)$$
$$\mathrm{PE}_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{2i/d_{\mathrm{model}}}}\right)$$

Where

pos denotes the position.

i denotes the dimension.

$$L_{XE}(\theta) = -\sum_{t=1}^{T} \log\left(p_{\theta}\left(y_t^{\star} | y_{1:t-1}^{\star}\right)\right).$$

3.1.4 Use Case Diagram and Description

Image classification

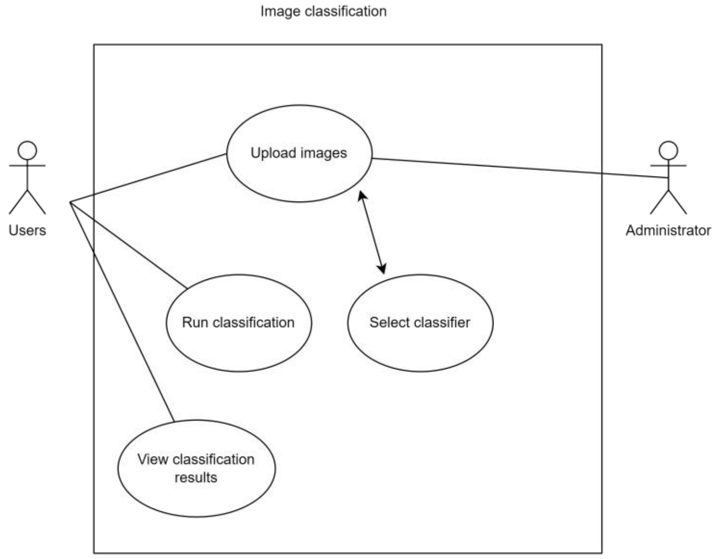


Figure 3.1.4.1 Use case diagram image classification

This is the use case diagram of Image classification which there are total two actors whose include users and administrator. The user is the person who interact with system and administrator is person responsible for creating the function and system maintenance and management process.

For the use cases, we can see that user and administrator both can upload images to the system but for administrator is input the datasets of image into system and users are upload images for image classification which to detect and classify images. User can select classifier that user can select specific image classification model or algorithm to use for classification task. User can run classification which initiates the classification results and process, and system will apply selected classifier to upload the images. User can view classification result after process classification with included predicted class or label of images.

Administrator can manage and update the image classification models that available on the system. Besides, administrator can perform routine maintenance tasks on system which include backups and update the dataset classification.

The include relationships which are upload image use case includes the Select classifier and Run Classification use cases because selecting a classifier and run the classification was the procedure include to upload an image. For the extend relationship which include Run Classification use case be extended to include additional function, such saving the results classification results or provide the details of statistics.

Image Captions

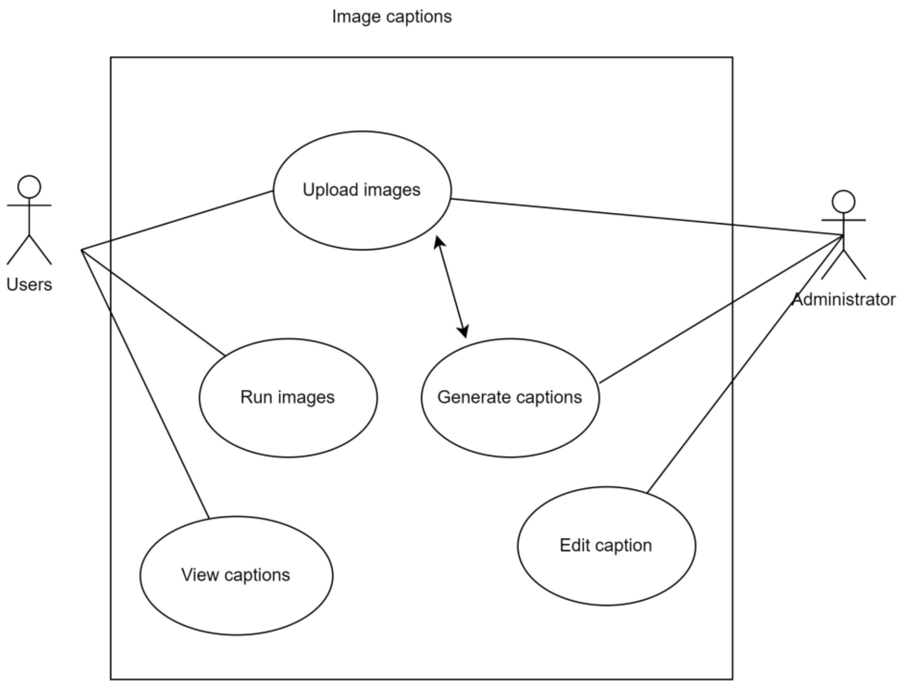


Figure 3.1.4.2 Use Case Diagram for Image Captions

There are two actors include in this use case diagram which includes user and administrator. User is the person who interact with the system to produce results or output. The administrator is an individual that for system management, maintenance, and system provider.

For the use cases, user, and administrator both can upload image to the system but for administrator is input the Kaggle datasets of image into system and users are upload images for image captions which to detect and provide back with images and captions together. The system will preprocess the uploaded images to prepare for caption generation, include with resizing and normalize the images. The extra image feature of Inception V3 by system utilizes the models to extract relevant features from preprocessed image, these features serve as input to caption generation RNN. RNN based model generate textual caption based extracted image features through natural language processing and neural networks. User can upload images for run and can view captions after caption generate and they can view and read generated caption for image.

Administrator can generate the captions and upload to system with datasets which the captions match with the images to ensure high accuracy on the output, but user can only view on the captions and images through output. Administrator can edit the captions for images which to avoid the wrongness content provide for user. They can manage and update the image caption models with RNN and Inception V3 to configure available at system. Lastly, perform routine maintenance tasks on system, includes updates and backups. The include relationship which have preprocess image includes upload images because preprocessing needed after image upload.

3.1.5 Activity diagram

Image classification

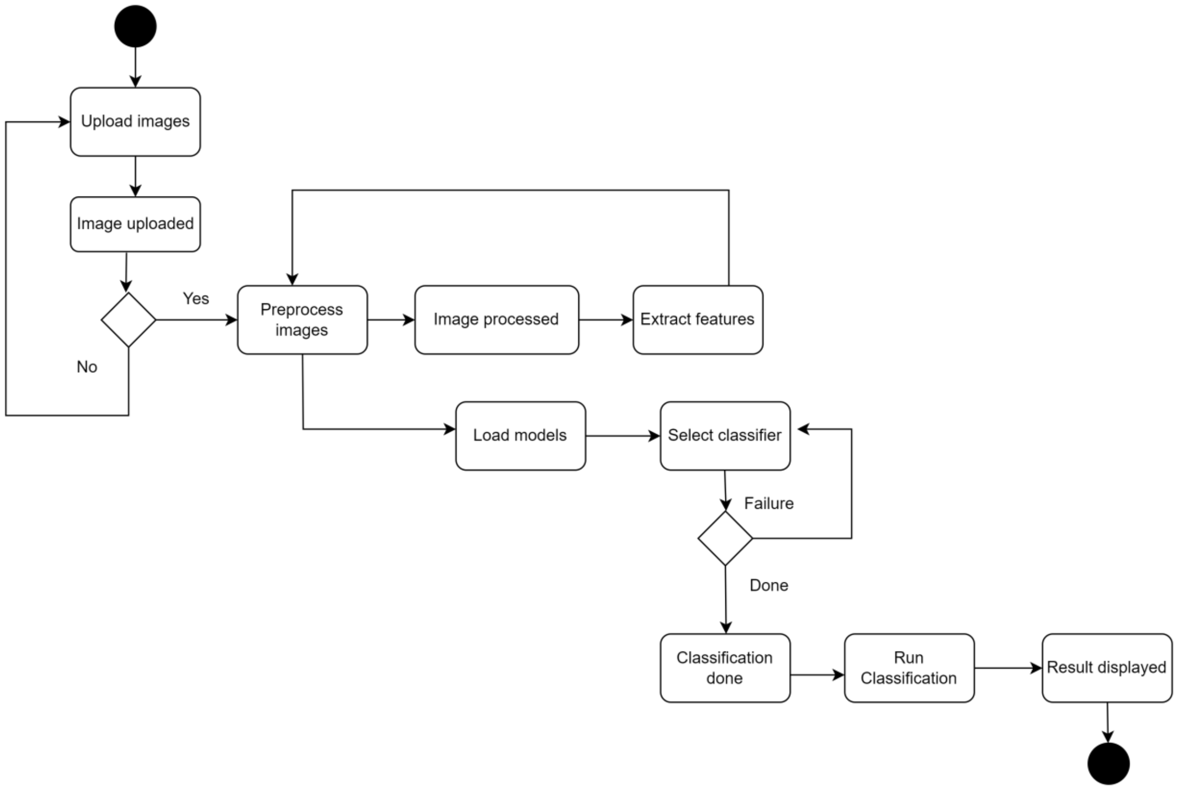


Figure 3.1.5.1 Activity diagram Image classification

The activity diagram shows the activities of upload images begins upload to the system with datasets. If uploaded success, process to preprocess images and if failure go back and upload the datasets again. When preprocess images start, the image will extract features and go back preprocess and load the models to select classifier auto. If classifier select failure repeat the step of select classifier and after done go to run classification and finally the result displayed and end.

Image captioning

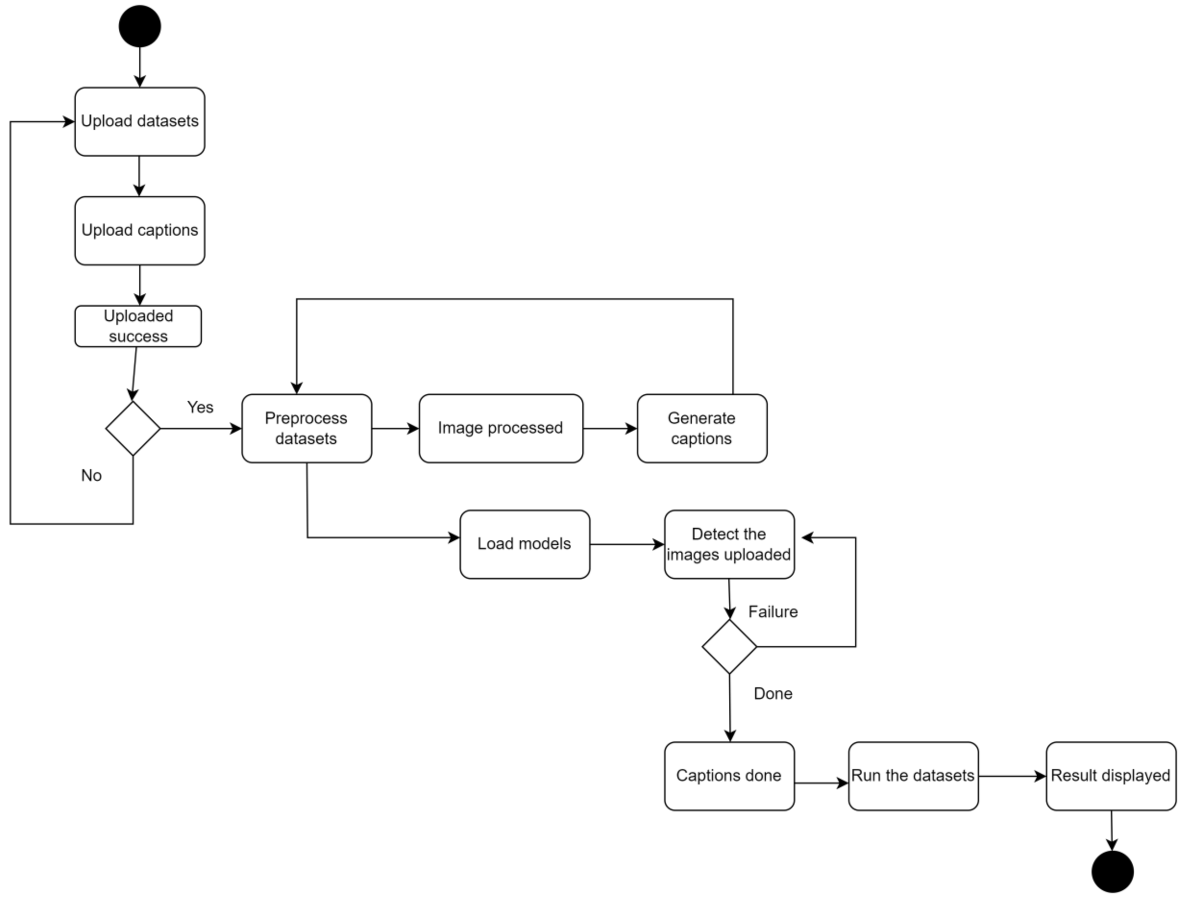


Figure 3.1.5.2 Activity Diagram for Image Captioning

The activity diagram show that image captioning activity diagram which system will upload datasets and captions together to system. If uploaded success, it will preprocess datasets and if failure will go back and upload again. After preprocess datasets, the image processed and go for generate captions and load the models to detect the images uploaded or not. If failure, try the step of detect images uploaded again and done go for caption processing and run the datasets to show the results for displaying.

Chapter 4 SYSTEM DESIGN

4.1 System Block Diagram

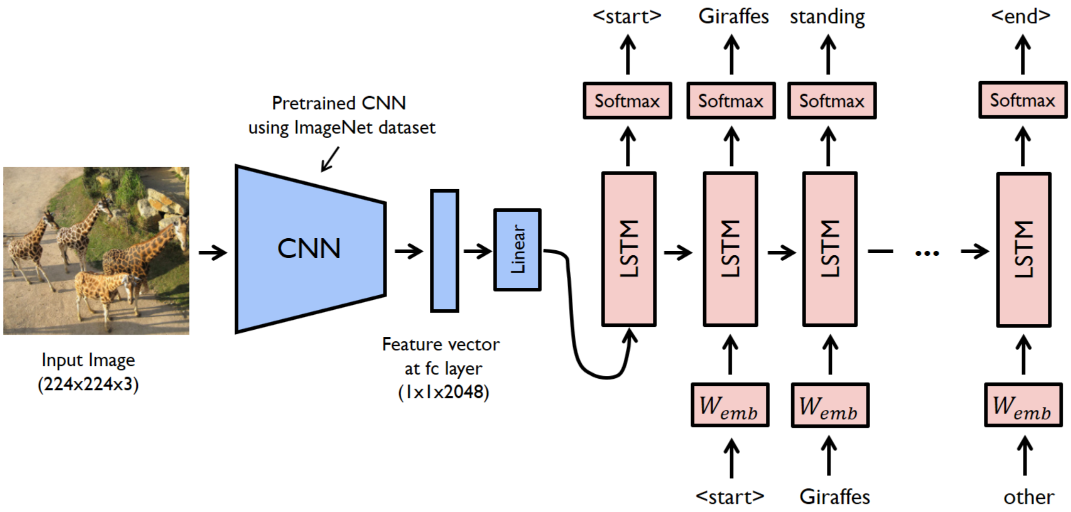


Figure 4.1 System Block Diagram

This is the system block diagram to train for deep learning model using Convolutional Neural Network with Recurrent Neural Network with the plus long shirt-term memory (LSTM) aim to provide caption and classification for images. The neural network has an active understand of images, so must train datasets for understand to generate captions and we use CNNs for images and use recurrent neural networks for text. Use RNN to generate caption word by word. During training, have corresponding captions for every image and training to give first word from caption and the feature vector to attention mechanism and output attention directly go to RNN, and task RNN predict the next word in caption. If prediction correct, the loss minimum and if wrong the loss maximum. The process generating repeat until end of sentence reached. Attention mechanism word as name suggests and automatically learn to focus various regions in image.

4.2 Systems Components Specifications

The system components specification for image captioning and image classification involves defining the necessary elements and their functionalities to perform these tasks effectively. For image classification, the system typically comprises an input layer for receiving image data, a convolutional neural network (CNN) architecture responsible for feature extraction and pattern recognition, fully connected layers for classification, and an output layer providing class probabilities or labels. Components such as GPUs or TPUs may be utilized for accelerated computations. Image captioning, on the other hand, requires a combination of CNN and recurrent neural network (RNN) components. The CNN extracts image features, which are then fed into the RNN, comprising LSTM or Transformer layers, for natural language generation. Both tasks necessitate pre-processing modules for data preparation, which may involve resizing, normalization, and feature extraction. Additionally, post-processing components may be incorporated for enhancing caption readability or refining classification results. The system should also consider hardware compatibility and scalability, making it adaptable for various applications and datasets. Overall, these components collectively enable the system to perform image classification and captioning tasks efficiently and accurately.

4.3 Circuits and Components Design

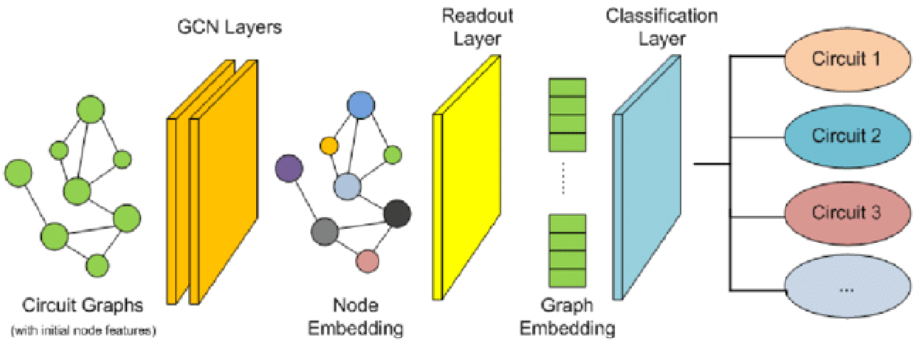


Figure 4.3 Components and circuits image classification

Designing the circuit and components for image classification using a Graph Convolutional Network (GCN) involves several key steps. In this design, transform the image dataset into a graph structure where nodes represent image regions and edges signify relationships between regions based on spatial proximity. Each node is associated with features extracted from these regions. The core of the design lies in the GCN architecture, which comprises an input layer, multiple graph convolutional layers for aggregating information from neighboring nodes, activation functions, optional pooling layers to reduce graph size, and an output layer for class probability prediction. The selection of components is crucial, considering the implementation of matrix multiplication units, activation function circuits, and pooling circuitry. The hardware platform choice, whether a CPU, GPU, FPGA, or custom ASIC, impacts performance and power efficiency. If applicable, PCB layout design should account for proper grounding and signal integrity. Building a physical prototype for testing, sourcing components, and manufacturing at scale follow. Rigorous testing, validation, and compliance checks ensure the circuit meets specifications. Iterative improvements based on testing results are implemented, and production is scaled accordingly. Collaborative efforts between hardware engineers and domain experts are fundamental for the successful realization of this custom hardware for GCN-based image classification.

4.4 System Component Interaction Operation

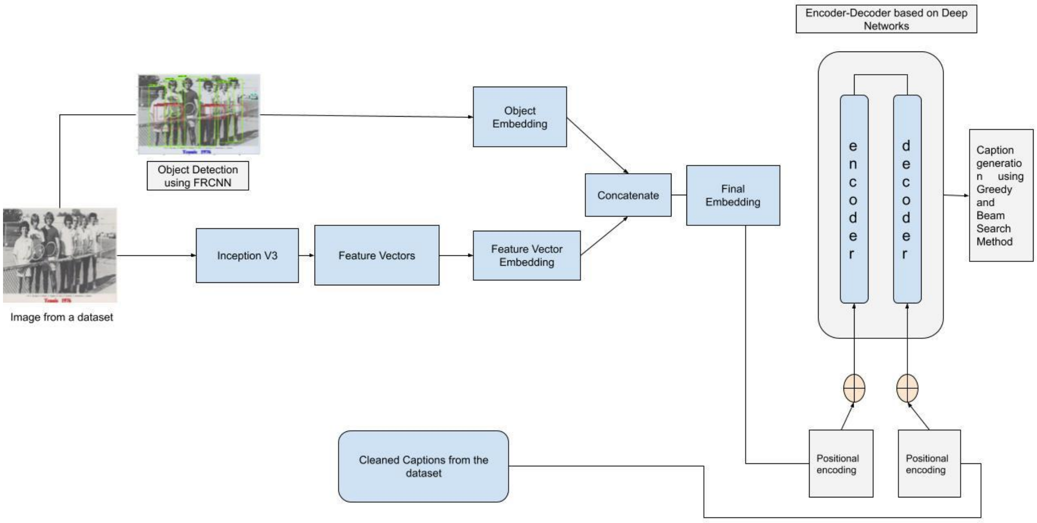


Figure 4.4 Systems Component

The system components for image captioning and image classification utilizing the Inception V3 architecture and Recurrent Neural Networks (RNNs) involve a well-defined sequence of operations. Initially, both tasks begin with input images. Inception V3, a powerful convolutional neural network (CNN), serves as the primary feature extractor for both tasks, extracting intricate visual features from the input images. For image classification, the extracted features undergo classification using fully connected layers, ultimately assigning class labels to the images. Conversely, for image captioning, the extracted features serve as the initial input to an RNN, such as an LSTM or Transformer, which generates sequential outputs in the form of natural language captions, capturing the essence of the images. This interaction between the Inception V3 feature extractor and the RNN language model forms the core of the system, with data flowing between them during both training and inference. Throughout training, the components are optimized to minimize classification or captioning loss using optimization techniques like gradient descent. The evaluation of image classification involves measuring classification accuracy, while image captioning performance is assessed using metrics such as BLEU, METEOR, or CIDEr, comparing the generated captions to reference captions. This system architecture effectively combines computer vision and natural language processing to perform image-related tasks, generating textual descriptions or classifying images with high accuracy.

Chapter 5 System Outcome and Discussion

5.1 Hardware setup

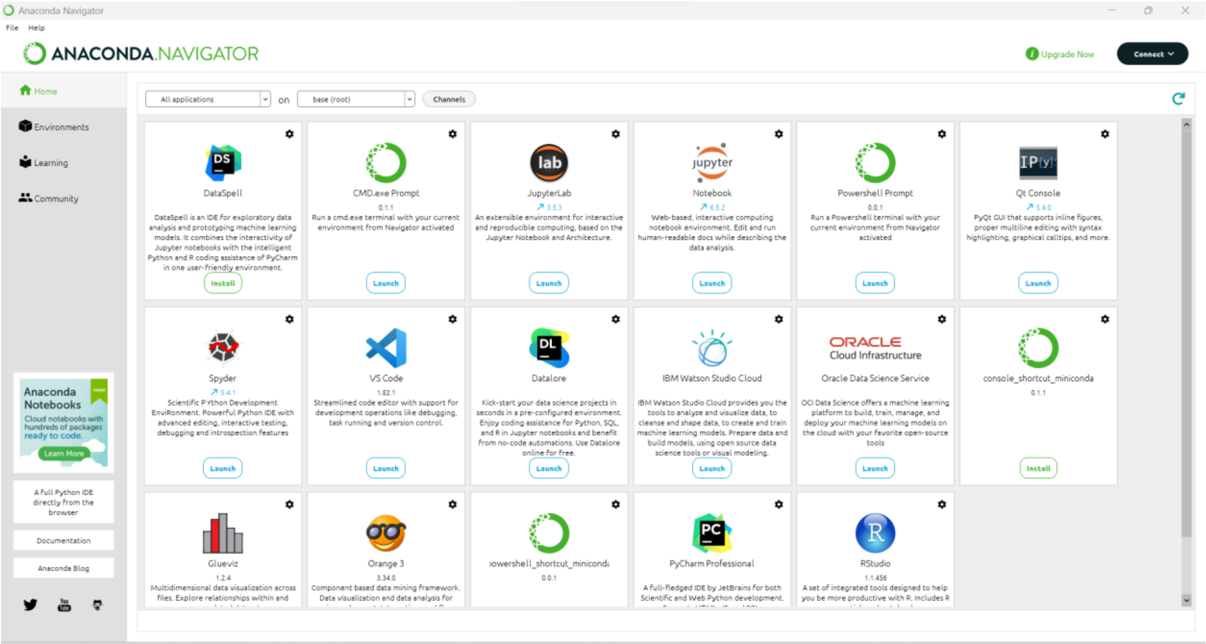
The hardware involved in this project is a laptop and mobile device. Using of both devices of hardware tools are for research-based and ensure that the data collection was correct. A computer issued for the process of 3D visualization and segmentation from MRI and CT datasets to obtain the 3D model objects, then it also used for applying AR technology on the 3D model objects. A mobile device used for testing the correctness of data and research.

Table 5.1 Specifications of laptop

Description	**Specifications**
Model	Asus A510U series
Processor	Intel Core i5-8250U
Operating System	Windows 11
Graphic	NVIDIA GeForce GT 930MX 2GB DDR3
Memory	12GB DDR4 RAM
Storage	128GB SATA HDD

5.2 Software setup

I am using google colab, cloud-based platform that provide by Google allows users to write and run Python code at a web-based interactive environment and it is good for deep learning. In this case, I use for my image captioning process. Because this platform is free and don’t need setup and run at cloud which build in Python. It provides GPU Acceleration which free GPU access means can speed up training of deep learning models. I download anaconda navigator for launching the Jupyter notebook and we can download features we want through the environment path section to add more features.



DataSpell
DataSpell is an IDE for exploratory data analysis and prototyping machine learning models. It combines the interactivity of Jupyter notebooks with the intelligent Python and R coding assistance of PyCharm in one user-friendly environment.

CMD.exe Prompt
0.1.1
Run a cmd.exe terminal with your current environment from Navigator activated

JupyterLab
↗ 3.5.3
An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

Notebook
↗ 6.5.2
Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

Powershell Prompt
0.0.1
Run a Powershell terminal with your current environment from Navigator activated

Qt Console
↗ 5.4.0
PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.

Spyder
↗ 5.4.1
Scientific PYthon Development EnviRonment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features

VS Code
1.82.1
Streamlined code editor with support for development operations like debugging, task running and version control.

Datalore
Kick-start your data science projects in seconds in a pre-configured environment. Enjoy coding assistance for Python, SQL, and R in Jupyter notebooks and benefit from no-code automations. Use Datalore online for free.

IBM Watson Studio Cloud
IBM Watson Studio Cloud provides you the tools to analyze and visualize data, to cleanse and shape data, to create and train machine learning models. Prepare data and build models, using open source data science tools or visual modeling.

Oracle Data Science Service
OCI Data Science offers a machine learning platform to build, train, manage, and deploy your machine learning models on the cloud with your favorite open-source tools

console_shortcut_miniconda
0.1.1

Glueviz
1.2.4
Multidimensional data visualization across files. Explore relationships within and

Orange 3
3.34.0
Component based data mining framework. Data visualization and data analysis for

owershell_shortcut_minicond
0.0.1

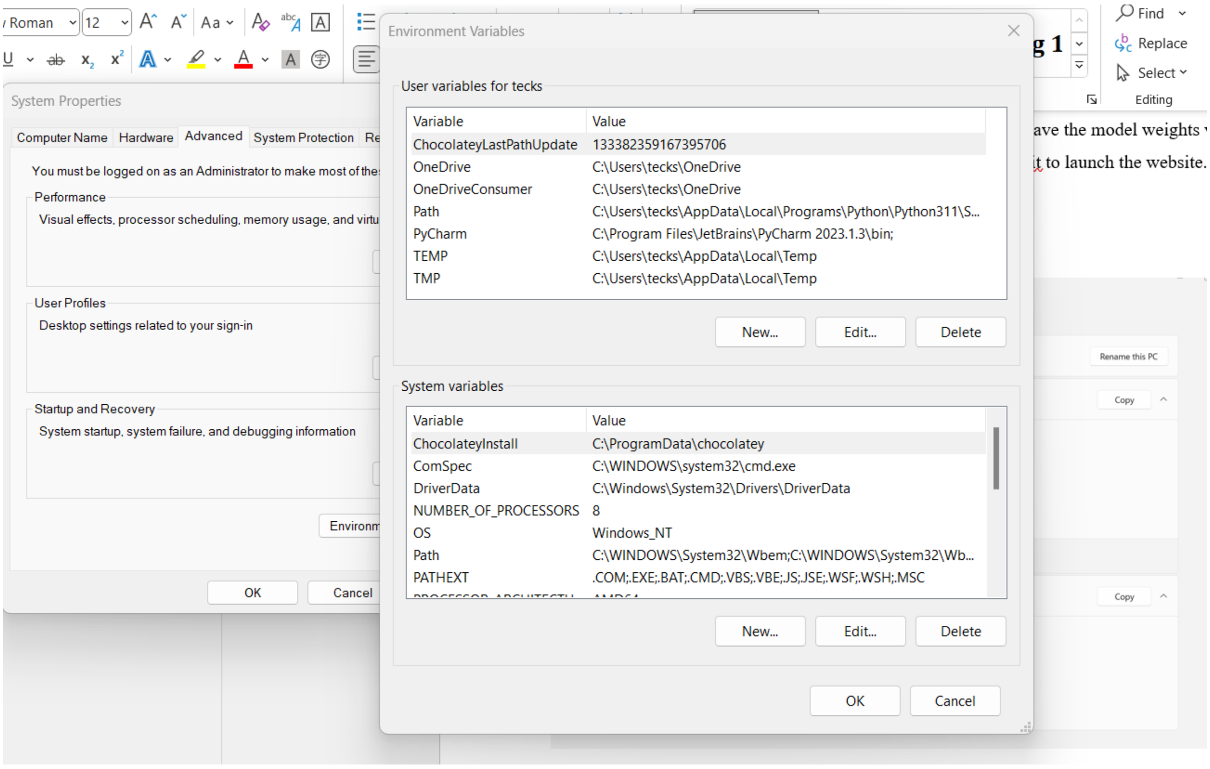
PyCharm Professional
A full-fledged IDE by JetBrains for both Scientific and Web Python development.

RStudio
1.1.456
A set of integrated tools designed to help you be more productive with R. Includes R

Figure 5.2 Anaconda Navigator

I am using visual studio code to connect from google colab and save the model weights with connect to visual studio code python code and I run with streamlit to launch the website.

5.3 Setting and Configuration



ave the model weights
t to launch the website.

System Properties

Computer Name | Hardware | Advanced | System Protection | Re

You must be logged on as an Administrator to make most of the

Performance
Visual effects, processor scheduling, memory usage, and virtu

User Profiles
Desktop settings related to your sign-in

Startup and Recovery
System startup, system failure, and debugging information

Environm

OK Cancel

Environment Variables

User variables for tecks

Variable	Value
ChocolateyLastPathUpdate	133382359167395706
OneDrive	C:\Users\tecks\OneDrive
OneDriveConsumer	C:\Users\tecks\OneDrive
Path	C:\Users\tecks\AppData\Local\Programs\Python\Python311\S...
PyCharm	C:\Program Files\JetBrains\PyCharm 2023.1.3\bin;
TEMP	C:\Users\tecks\AppData\Local\Temp
TMP	C:\Users\tecks\AppData\Local\Temp

New... Edit... Delete

System variables

Variable	Value
ChocolateyInstall	C:\ProgramData\chocolatey
ComSpec	C:\WINDOWS\system32\cmd.exe
DriverData	C:\Windows\System32\Drivers\DriverData
NUMBER_OF_PROCESSORS	8
OS	Windows_NT
Path	C:\WINDOWS\System32\Wbem;C:\WINDOWS\System32\Wb...
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC
PROCESSOR_ARCHITECTU...	AMD64

New... Edit... Delete

OK Cancel

Figure 5.3 Setting and Configuration

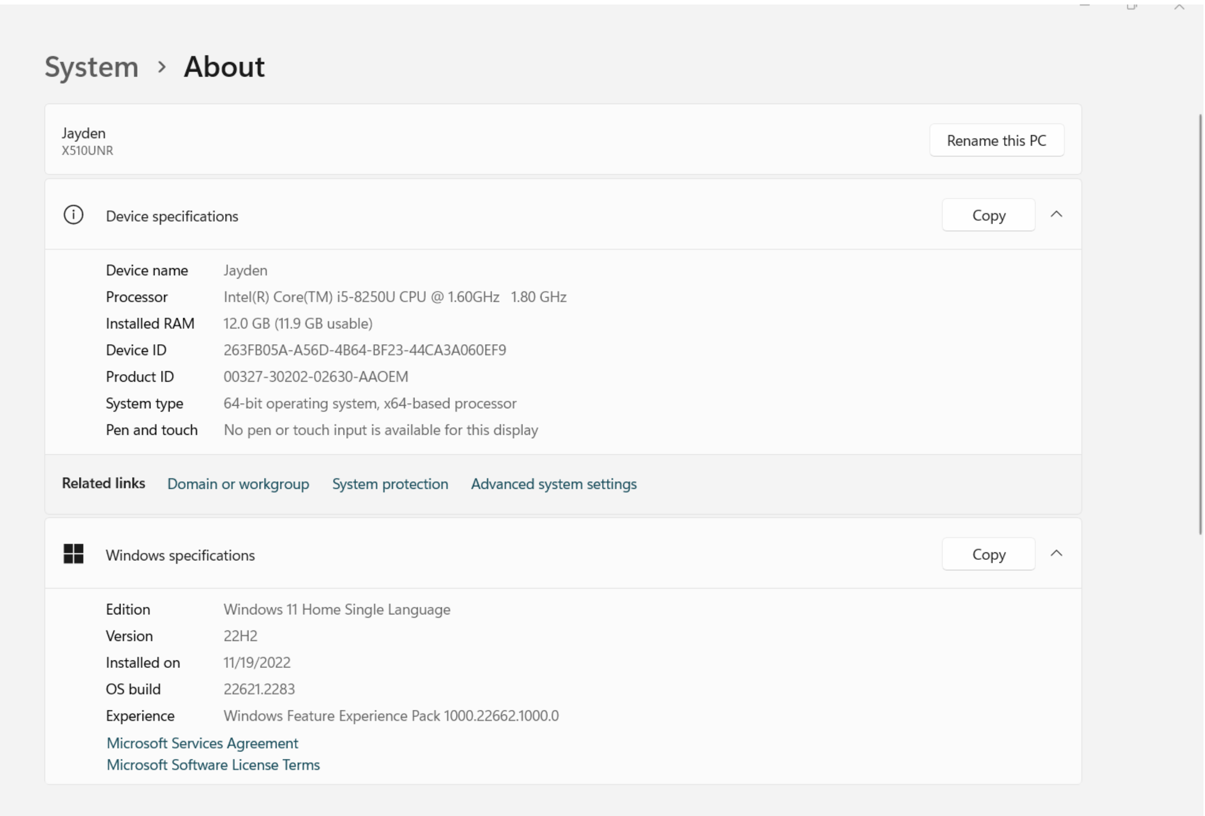
We need go to setting and find system environment after that need to change path at above click edit change to connect with python path directories.

The hardware involved in this project is laptop and mobile device. A computer issued for the process of 2D visualization and segmentation from visual studio and kaggle datasets to obtain the 2D model objects, then it also used for applying AR technology on the 2D model objects. A laptop is used for testing and deploying this website application in image classification.
The following are the steps of image classification process:

i. Import the libraries.
ii. Load the data.
iii. Look at the data types of variables.
iv. Get the shape of arrays.
v. Have a look at first image as array.
vi. Show the image as pictures.
vii. Get the image label.
viii. Get the image classification.
ix. Print the image class.
x. Convert labels into set of 10 numbers input to neural networks.
xi. Print new labels.
xii. Print new labels of image above.
xiii. Normalize pixels to value between 0 and 1.
xiv. Create models architecture.
xv. Add first layer.
xvi. Add a pooling layer.
xvii. Add another convolutional layer.
xviii. Add another pooling layer.
xix. Add a flattening layer.
xx. Add layers with 1000 neurons.
xxi. Add drop out layer.
xxii. Compile models.
xxiii. Train models.
xxiv. Evaluate models using test data set.
xxv. Visualize models accuracy.
xxvi. Visualize models loss.

xxvii. Test model with example.
xxviii. Show image.
xxix. Resize image.
xxx. Get models prediction.
xxxi. Show the prediction.
xxxii. Sort prediction.
xxxiii. Print first five predictions.

5.4 System Operation



System › About

Jayden
X510UNR

Device specifications

Device name	Jayden
Processor	Intel(R) Core(TM) i5-8250U CPU @ 1.60GHz 1.80 GHz
Installed RAM	12.0 GB (11.9 GB usable)
Device ID	263FB05A-A56D-4B64-BF23-44CA3A060EF9
Product ID	00327-30202-02630-AAOEM
System type	64-bit operating system, x64-based processor
Pen and touch	No pen or touch input is available for this display

Windows specifications

Edition	Windows 11 Home Single Language
Version	22H2
Installed on	11/19/2022
OS build	22621.2283
Experience	Windows Feature Experience Pack 1000.22662.1000.0

Microsoft Services Agreement
Microsoft Software License Terms

Figure 5.4 System Operation

5.5 Implementation Issues and Challenges

One of the primary challenges is generating captions that accurately describe the content of the image while considering the context. Achieving a balance between specificity and relevance is difficult. Images can often be interpreted in multiple ways, leading to ambiguity in captions. Resolving this ambiguity to provide clear and concise captions is a significant challenge. Besides, to decrease failure caption of image accuracy is quite challenges. The deploying image captioning models in real-time applications with low latency requirements can be challenging due to the computational demands of some models.

5.6 Concluding Remark

Finally, picture categorization and captioning are two exciting disciplines within computer vision and machine learning, with numerous applications and continuing research. Image classification is labeling or categorizing photos, and image captioning entails creating descriptive textual explanations for images. Both domains face data quality, model complexity, generalization, and real-world implementation issues.

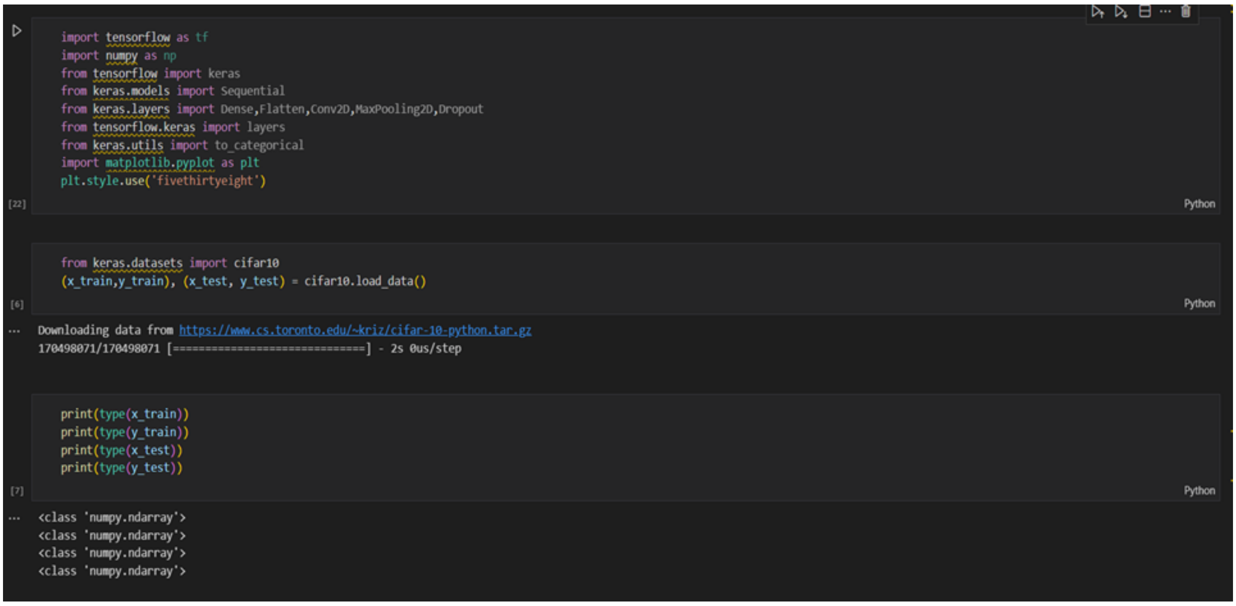
Despite these obstacles, tremendous progress has been made due to the advancement of deep learning algorithms, big datasets, and powerful computer resources. Transfer learning and pre-trained models have been critical in improving performance, making these technologies more accessible and relevant in a variety of fields. As these sectors progress, we should expect additional advances in picture interpretation, which will lead to better applications in areas such as healthcare, autonomous vehicles, content recommendation, and more. In the intriguing realms of image classification and image captioning, researchers and practitioners will continue to confront issues and push the frontiers of what is possible.

Overall, these technologies have the potential to transform how humans interact with and comprehend visual data, making them indispensable components of the ever-expanding realm of artificial intelligence and computer vision.

CHAPTER 6 SYSTEM EVALUATION AND DISCUSSION

6.1 System Testing and Performance Metrics

For image classification



```
import tensorflow as tf
import numpy as np
from tensorflow import keras
from keras.models import Sequential
from keras.layers import Dense,Flatten,Conv2D,MaxPooling2D,Dropout
from tensorflow.keras import layers
from keras.utils import to_categorical
import matplotlib.pyplot as plt
plt.style.use('fivethirtyeight')
```

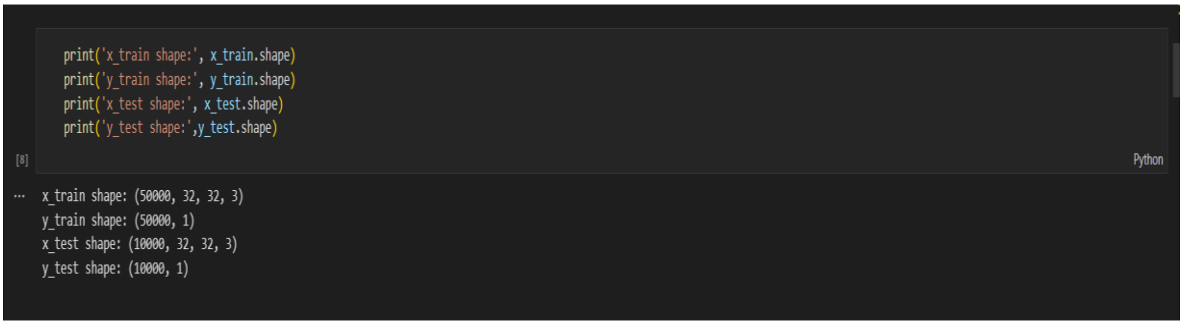
```
from keras.datasets import cifar10
(x_train,y_train), (x_test, y_test) = cifar10.load_data()
```

```
Downloading data from https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz
170498071/170498071 [==============================] - 2s 0us/step
```

```
print(type(x_train))
print(type(y_train))
print(type(x_test))
print(type(y_test))
```

```
<class 'numpy.ndarray'>
<class 'numpy.ndarray'>
<class 'numpy.ndarray'>
<class 'numpy.ndarray'>
```

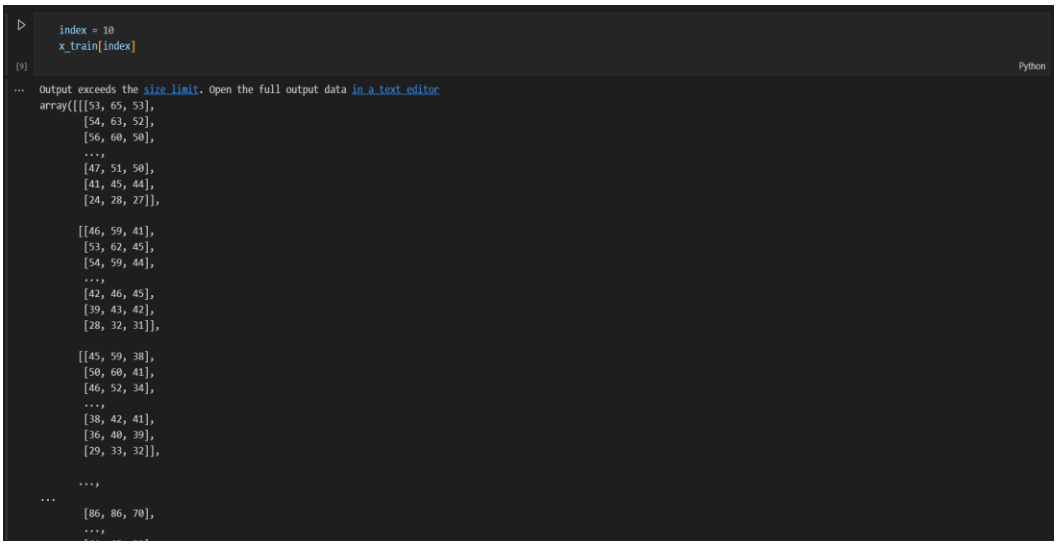
Import libraries necessary and get the datasets to load the data and print them out.



```
print('x_train shape:', x_train.shape)
print('y_train shape:', y_train.shape)
print('x_test shape:', x_test.shape)
print('y_test shape:',y_test.shape)
```

```
x_train shape: (50000, 32, 32, 3)
y_train shape: (50000, 1)
x_test shape: (10000, 32, 32, 3)
y_test shape: (10000, 1)
```

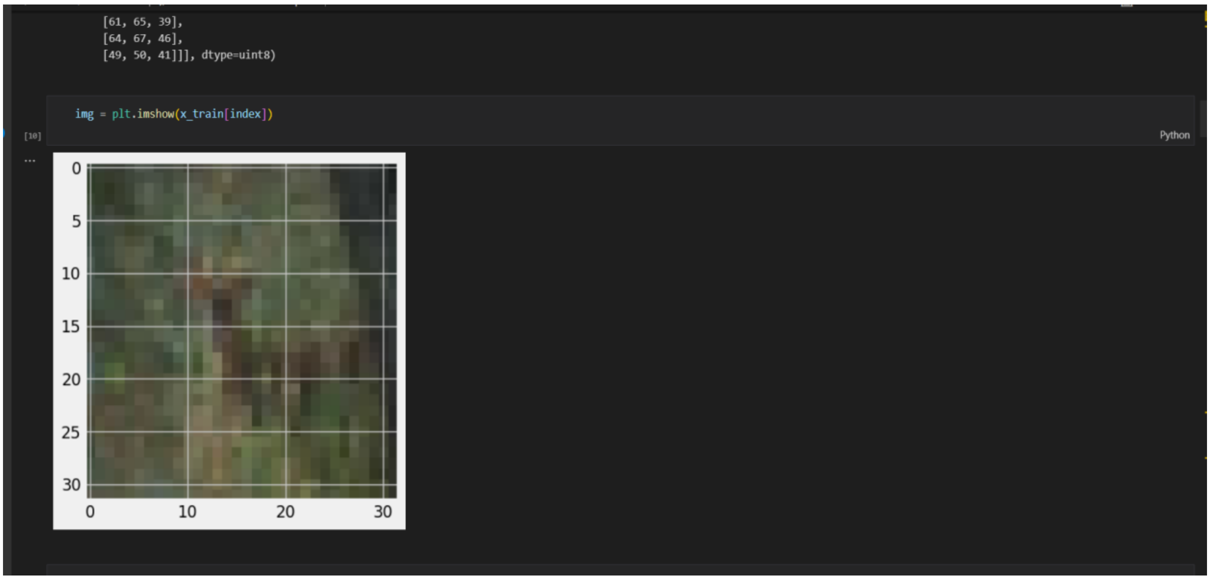
The system shows output of x_train shape and y_test of datasets.



```
index = 10
x_train[index]
```

```
Output exceeds the size limit. Open the full output data in a text editor
array([[[53, 65, 53],
        [54, 63, 52],
        [56, 60, 50],
        ...,
        [47, 51, 50],
        [41, 45, 44],
        [24, 28, 27]],

       [[46, 59, 41],
        [53, 62, 45],
        [54, 59, 44],
        ...,
        [42, 46, 45],
        [39, 43, 42],
        [28, 32, 31]],

       [[45, 59, 38],
        [50, 60, 41],
        [46, 52, 34],
        ...,
        [38, 42, 41],
        [36, 40, 39],
        [29, 33, 32]],

       ...,
...
        [86, 86, 70],
        ...,
```

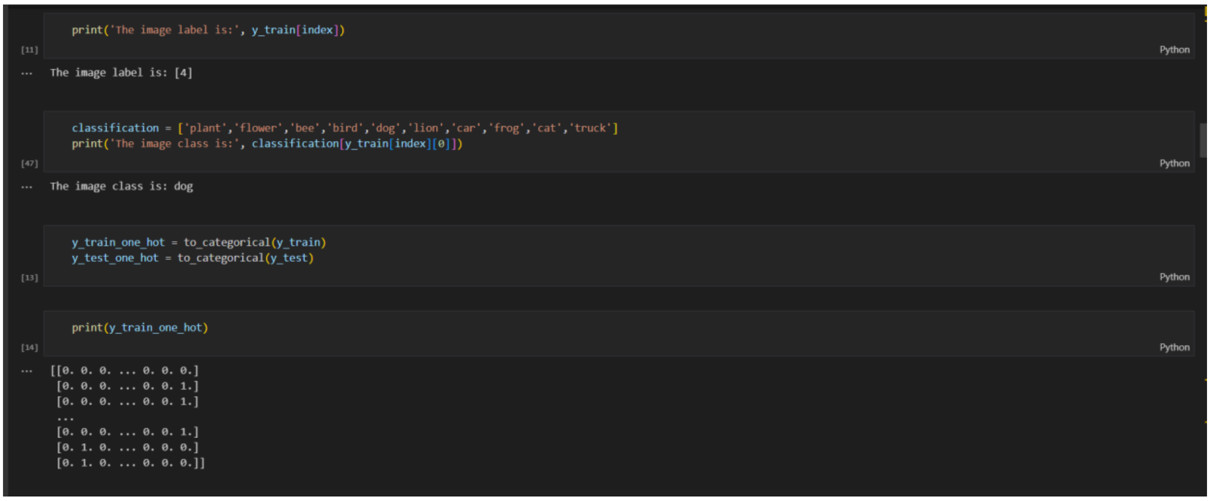
Declare the output size set at index of 10.



```
        [61, 65, 39],
        [64, 67, 46],
        [49, 50, 41]]], dtype=uint8)
```

```
img = plt.imshow(x_train[index])
```

Declare to show the output of picture.



```
print('The image label is:', y_train[index])
```

```
The image label is: [4]
```

```
classification = ['plant','flower','bee','bird','dog','lion','car','frog','cat','truck']
print('The image class is:', classification[y_train[index][0]])
```

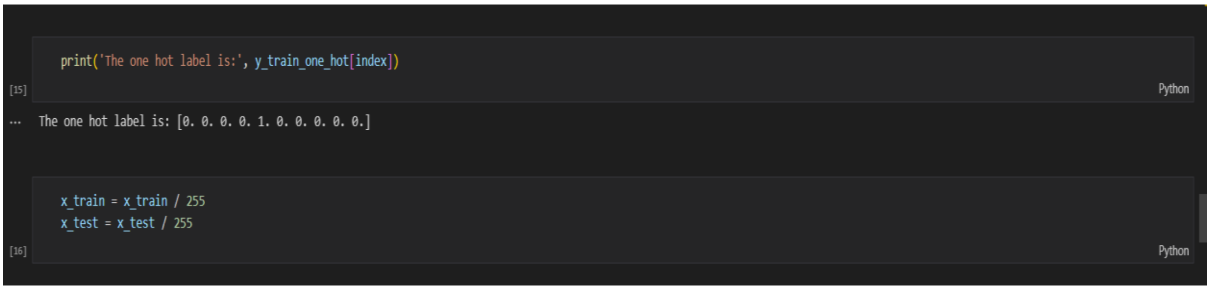
```
The image class is: dog
```

```
y_train_one_hot = to_categorical(y_train)
y_test_one_hot = to_categorical(y_test)
```

```
print(y_train_one_hot)
```

```
[[0. 0. 0. ... 0. 0. 0.]
 [0. 0. 0. ... 0. 0. 1.]
 [0. 0. 0. ... 0. 0. 1.]
 ...
 [0. 0. 0. ... 0. 0. 1.]
 [0. 1. 0. ... 0. 0. 0.]
 [0. 1. 0. ... 0. 0. 0.]]
```

Show how was the image_label and what is the image class which image label equal to 4 means that image class will show the fourth place of data. After train dataset, print out the datasets.

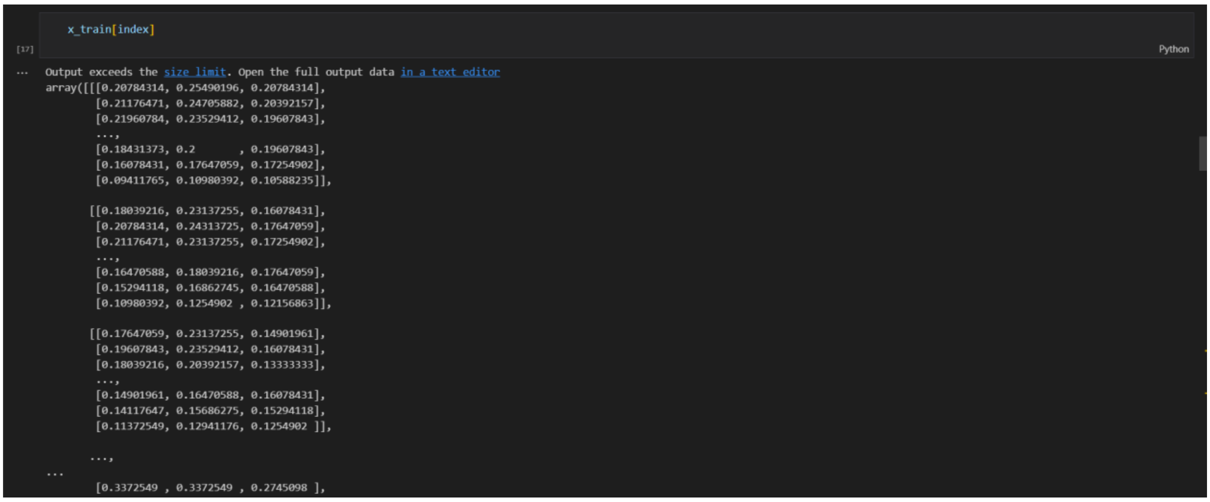


```
print('The one hot label is:', y_train_one_hot[index])
```

```
The one hot label is: [0. 0. 0. 0. 1. 0. 0. 0. 0. 0.]
```

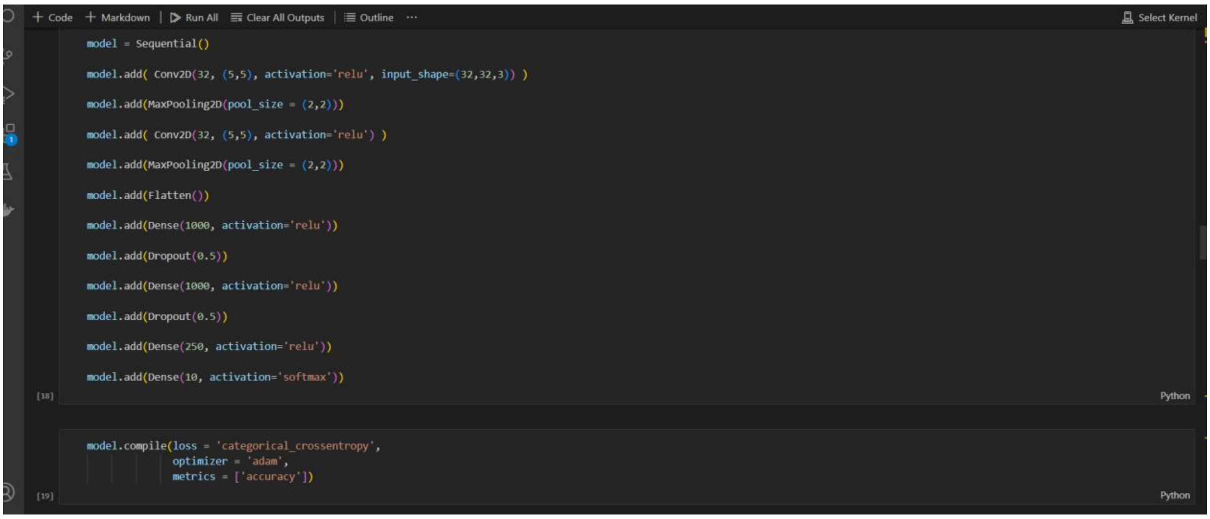
```
x_train = x_train / 255
x_test = x_test / 255
```

Print sentence description with y_train of image set.



```
x_train[index]
```

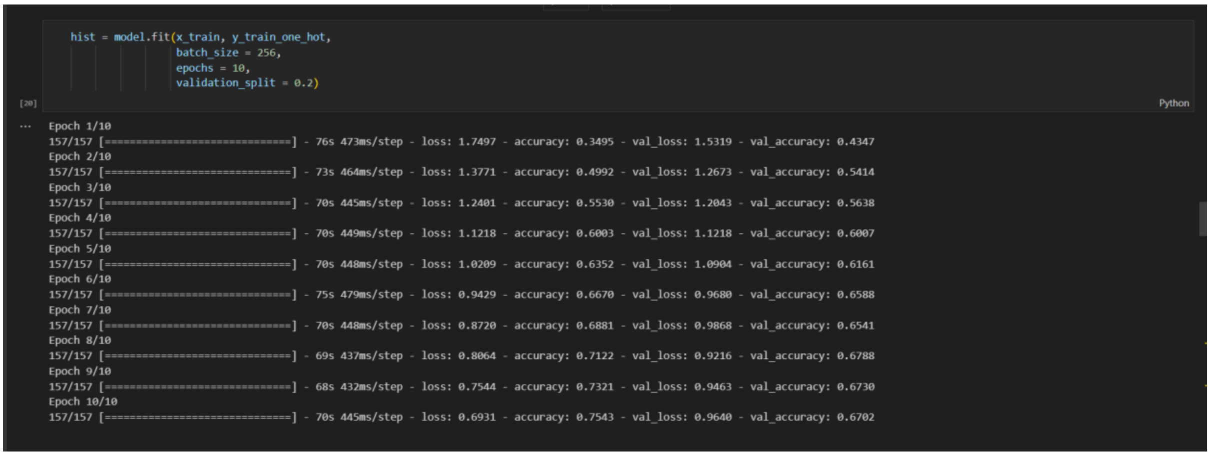
```
Output exceeds the size limit. Open the full output data in a text editor
array([[[[0.20784314, 0.25490196, 0.20784314],
         [0.21176471, 0.24705882, 0.20392157],
         [0.21960784, 0.23529412, 0.19607843],
         ...,
         [0.18431373, 0.2       , 0.19607843],
         [0.16078431, 0.17647059, 0.17254902],
         [0.09411765, 0.10980392, 0.10588235]],

        [[0.18039216, 0.23137255, 0.16078431],
         [0.20784314, 0.24313725, 0.17647059],
         [0.21176471, 0.23137255, 0.17254902],
         ...,
         [0.16470588, 0.18039216, 0.17647059],
         [0.15294118, 0.16862745, 0.16470588],
         [0.10980392, 0.1254902 , 0.12156863]],

        [[0.17647059, 0.23137255, 0.14901961],
         [0.19607843, 0.23529412, 0.16078431],
         [0.18039216, 0.20392157, 0.13333333],
         ...,
         [0.14901961, 0.16470588, 0.16078431],
         [0.14117647, 0.15686275, 0.15294118],
         [0.11372549, 0.12941176, 0.1254902 ]],

        ...,
...
         [0.3372549 , 0.3372549 , 0.2745098 ],
```



```
model = Sequential()

model.add( Conv2D(32, (5,5), activation='relu', input_shape=(32,32,3)) )

model.add(MaxPooling2D(pool_size = (2,2)))

model.add( Conv2D(32, (5,5), activation='relu') )

model.add(MaxPooling2D(pool_size = (2,2)))

model.add(Flatten())

model.add(Dense(1000, activation='relu'))

model.add(Dropout(0.5))

model.add(Dense(1000, activation='relu'))

model.add(Dropout(0.5))

model.add(Dense(250, activation='relu'))

model.add(Dense(10, activation='softmax'))
```

```
model.compile(loss = 'categorical_crossentropy',
              optimizer = 'adam',
              metrics = ['accuracy'])
```

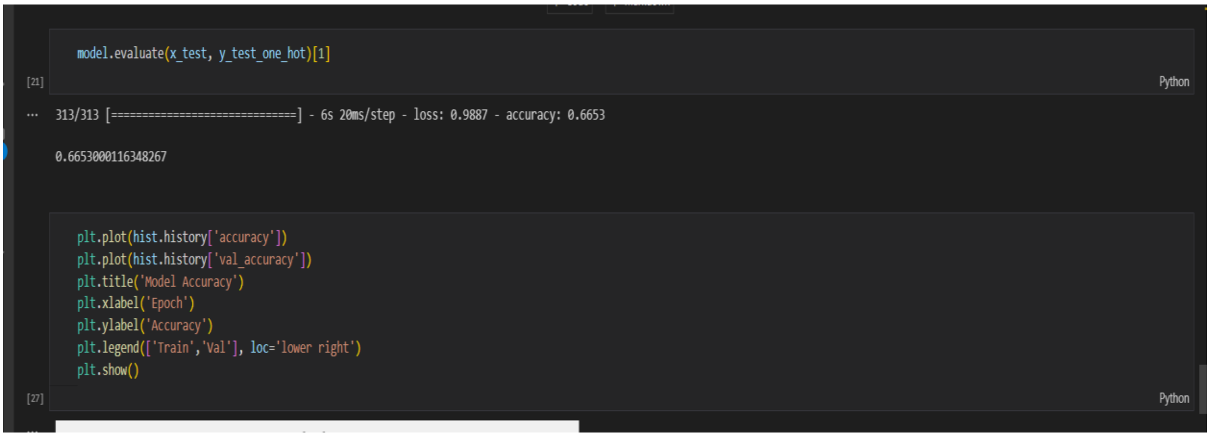
Set the model and add model to compile and optimizer it for accuracy.



```
hist = model.fit(x_train, y_train_one_hot,
                 batch_size = 256,
                 epochs = 10,
                 validation_split = 0.2)
```

```
Epoch 1/10
157/157 [==============================] - 76s 473ms/step - loss: 1.7497 - accuracy: 0.3495 - val_loss: 1.5319 - val_accuracy: 0.4347
Epoch 2/10
157/157 [==============================] - 73s 464ms/step - loss: 1.3771 - accuracy: 0.4992 - val_loss: 1.2673 - val_accuracy: 0.5414
Epoch 3/10
157/157 [==============================] - 70s 445ms/step - loss: 1.2401 - accuracy: 0.5530 - val_loss: 1.2043 - val_accuracy: 0.5638
Epoch 4/10
157/157 [==============================] - 70s 449ms/step - loss: 1.1218 - accuracy: 0.6003 - val_loss: 1.1218 - val_accuracy: 0.6007
Epoch 5/10
157/157 [==============================] - 70s 448ms/step - loss: 1.0209 - accuracy: 0.6352 - val_loss: 1.0904 - val_accuracy: 0.6161
Epoch 6/10
157/157 [==============================] - 75s 479ms/step - loss: 0.9429 - accuracy: 0.6670 - val_loss: 0.9680 - val_accuracy: 0.6588
Epoch 7/10
157/157 [==============================] - 70s 448ms/step - loss: 0.8720 - accuracy: 0.6881 - val_loss: 0.9868 - val_accuracy: 0.6541
Epoch 8/10
157/157 [==============================] - 69s 437ms/step - loss: 0.8064 - accuracy: 0.7122 - val_loss: 0.9216 - val_accuracy: 0.6788
Epoch 9/10
157/157 [==============================] - 68s 432ms/step - loss: 0.7544 - accuracy: 0.7321 - val_loss: 0.9463 - val_accuracy: 0.6730
Epoch 10/10
157/157 [==============================] - 70s 445ms/step - loss: 0.6931 - accuracy: 0.7543 - val_loss: 0.9640 - val_accuracy: 0.6702
```

Running the process and detect the loss and the accuracy of datasets of classification data is correct info or not.

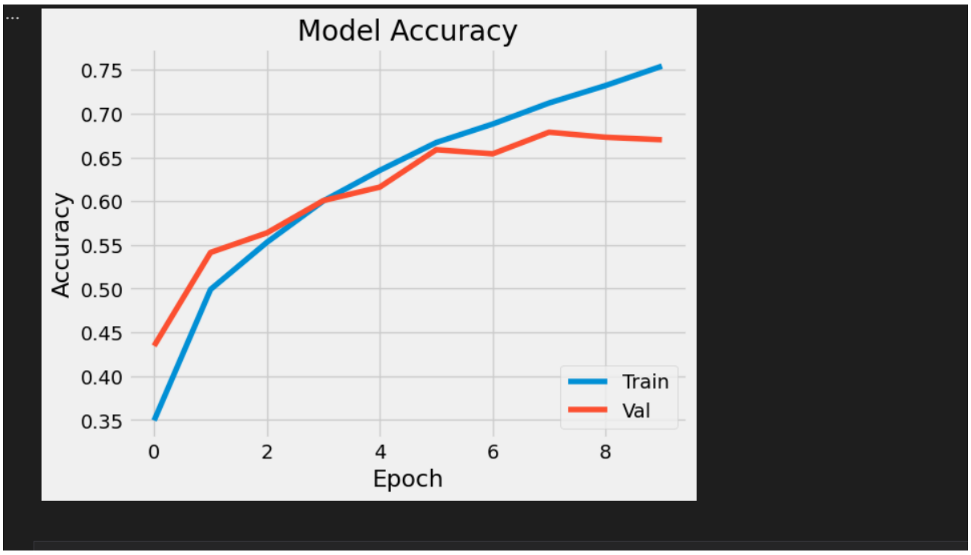


```
model.evaluate(x_test, y_test_one_hot)[1]
```

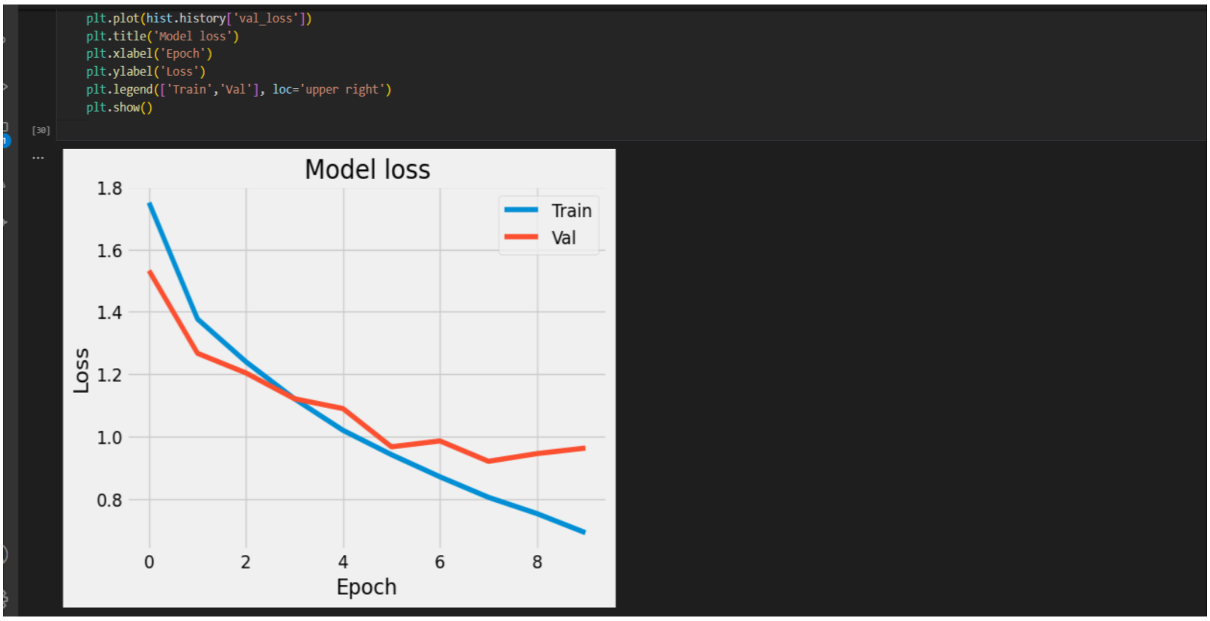
```
313/313 [==============================] - 6s 20ms/step - loss: 0.9887 - accuracy: 0.6653

0.6653000116348267
```

```
plt.plot(hist.history['accuracy'])
plt.plot(hist.history['val_accuracy'])
plt.title('Model Accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.legend(['Train','Val'], loc='lower right')
plt.show()
```

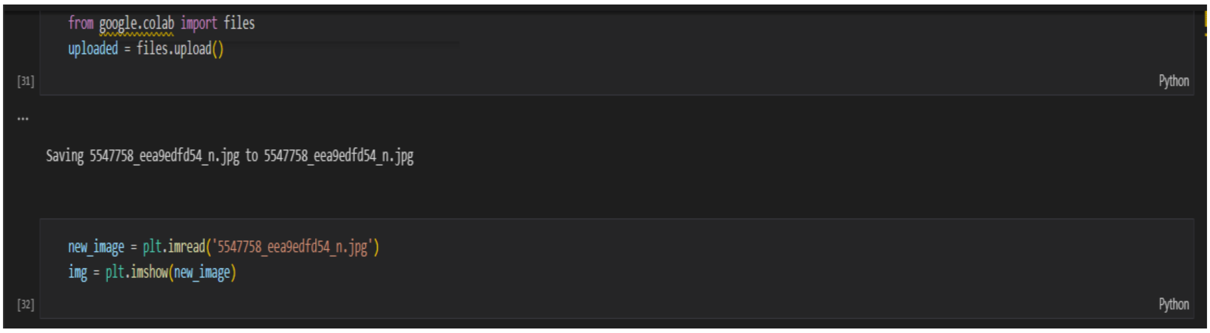
After evaluate accuracy of datasets, start to generate graph to plot the accuracy in graph to show data.



This is the model accuracy of train and val datasets whether image classification correct or not.



```
plt.plot(hist.history['val_loss'])
plt.title('Model loss')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.legend(['Train','Val'], loc='upper right')
plt.show()
```

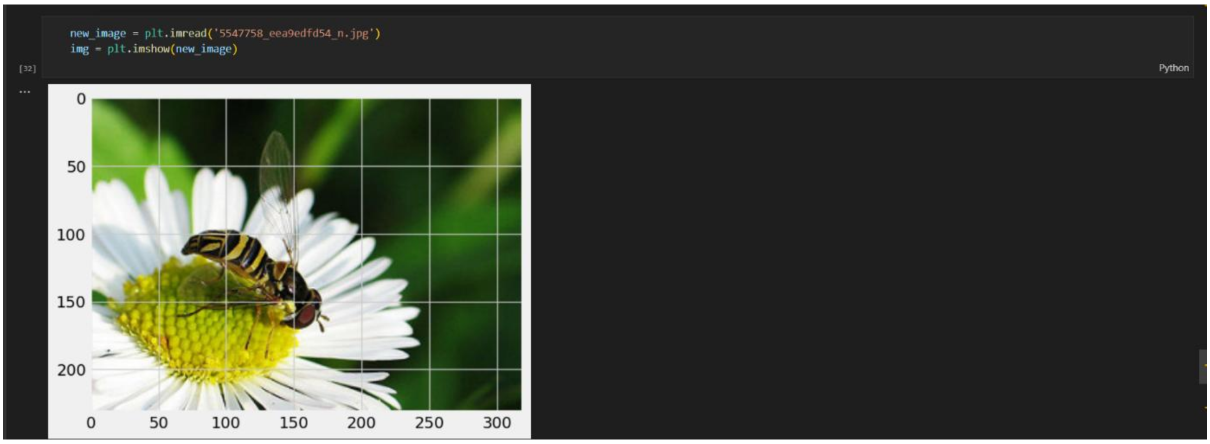


```
from google.colab import files
uploaded = files.upload()
```

```
Saving 5547758_eea9edfd54_n.jpg to 5547758_eea9edfd54_n.jpg
```

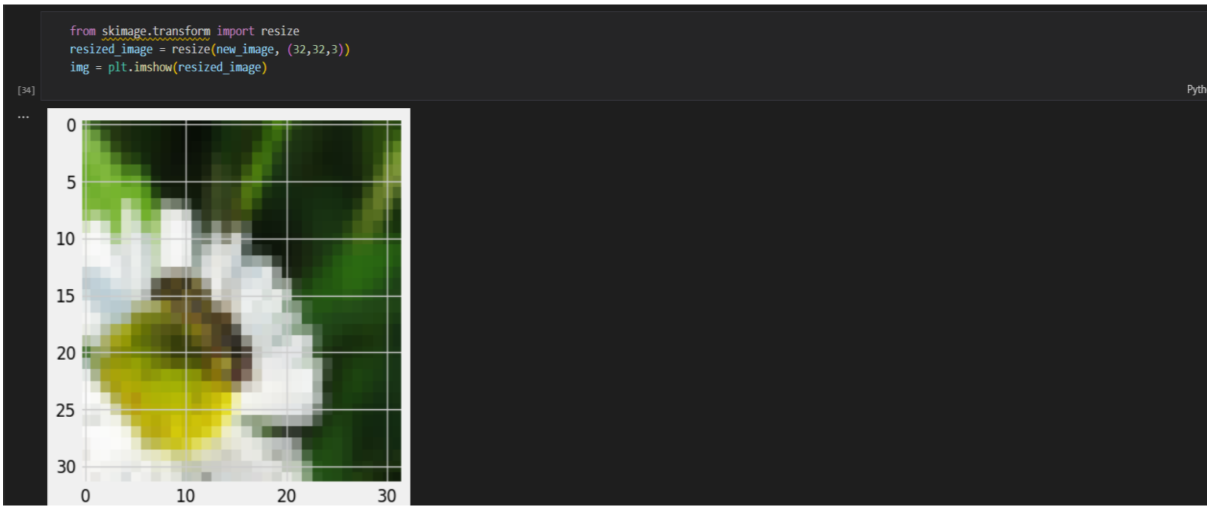
```
new_image = plt.imread('5547758_eea9edfd54_n.jpg')
img = plt.imshow(new_image)
```

From google colab to import files, upload the images we want to upload, and it will show saving the image with the id and show the new images.



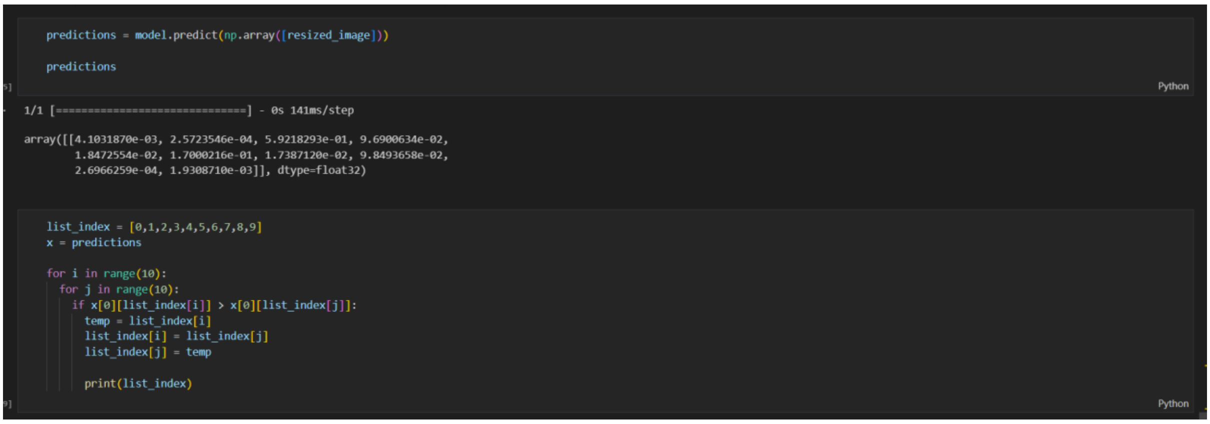
```
new_image = plt.imread('5547758_eea9edfd54_n.jpg')
img = plt.imshow(new_image)
```

The images show a bee on the flower.



```
from skimage.transform import resize
resized_image = resize(new_image, (32,32,3))
img = plt.imshow(resized_image)
```

Start to resize the image and show resize image output.

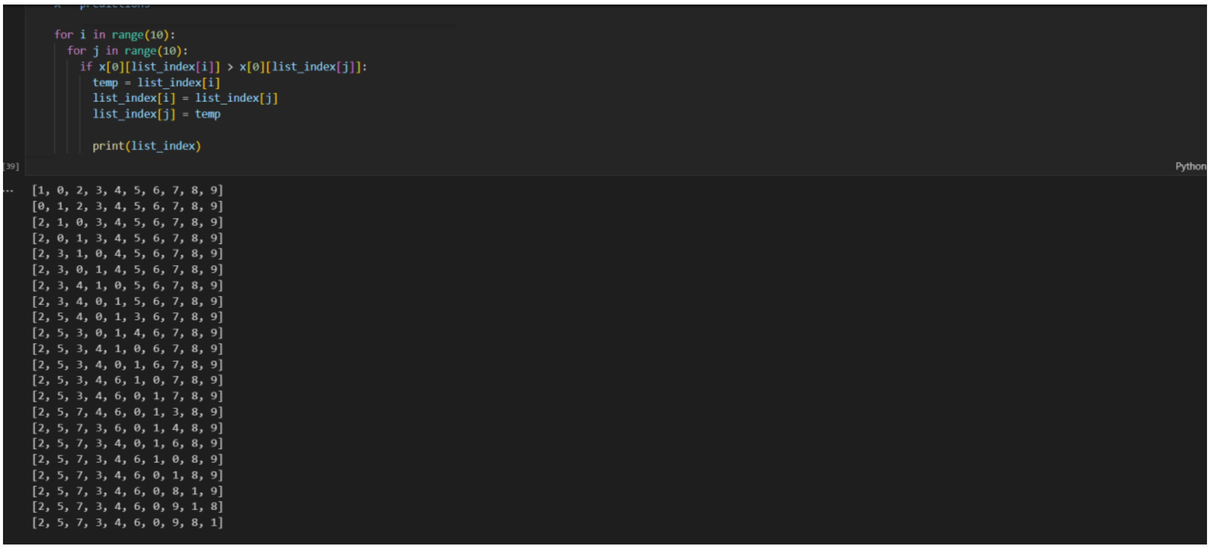


```
predictions = model.predict(np.array([resized_image]))

predictions
```

```
1/1 [==============================] - 0s 141ms/step

array([[4.1031870e-03, 2.5723546e-04, 5.9218293e-01, 9.6900634e-02,
        1.8472554e-02, 1.7000216e-01, 1.7387120e-02, 9.8493658e-02,
        2.6966259e-04, 1.9308710e-03]], dtype=float32)
```

```
list_index = [0,1,2,3,4,5,6,7,8,9]
x = predictions

for i in range(10):
  for j in range(10):
    if x[0][list_index[i]] > x[0][list_index[j]]:
      temp = list_index[i]
      list_index[i] = list_index[j]
      list_index[j] = temp

      print(list_index)
```

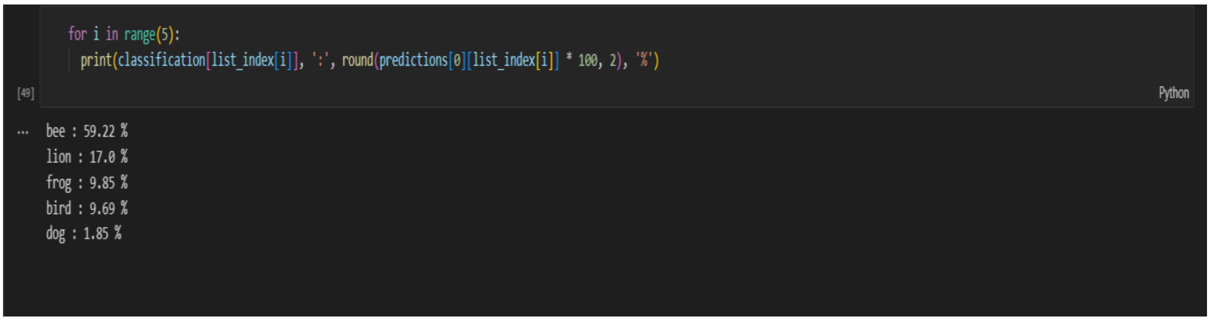
Predict the model of resized image.



```
for i in range(10):
  for j in range(10):
    if x[0][list_index[i]] > x[0][list_index[j]]:
      temp = list_index[i]
      list_index[i] = list_index[j]
      list_index[j] = temp

      print(list_index)
```

```
[1, 0, 2, 3, 4, 5, 6, 7, 8, 9]
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
[2, 1, 0, 3, 4, 5, 6, 7, 8, 9]
[2, 0, 1, 3, 4, 5, 6, 7, 8, 9]
[2, 3, 1, 0, 4, 5, 6, 7, 8, 9]
[2, 3, 0, 1, 4, 5, 6, 7, 8, 9]
[2, 3, 4, 1, 0, 5, 6, 7, 8, 9]
[2, 3, 4, 0, 1, 5, 6, 7, 8, 9]
[2, 5, 4, 0, 1, 3, 6, 7, 8, 9]
[2, 5, 3, 0, 1, 4, 6, 7, 8, 9]
[2, 5, 3, 4, 1, 0, 6, 7, 8, 9]
[2, 5, 3, 4, 0, 1, 6, 7, 8, 9]
[2, 5, 3, 4, 6, 1, 0, 7, 8, 9]
[2, 5, 3, 4, 6, 0, 1, 7, 8, 9]
[2, 5, 7, 4, 6, 0, 1, 3, 8, 9]
[2, 5, 7, 3, 6, 0, 1, 4, 8, 9]
[2, 5, 7, 3, 4, 0, 1, 6, 8, 9]
[2, 5, 7, 3, 4, 6, 1, 0, 8, 9]
[2, 5, 7, 3, 4, 6, 0, 1, 8, 9]
[2, 5, 7, 3, 4, 6, 0, 8, 1, 9]
[2, 5, 7, 3, 4, 6, 0, 9, 1, 8]
[2, 5, 7, 3, 4, 6, 0, 9, 8, 1]
```

Print the image with list_index format.

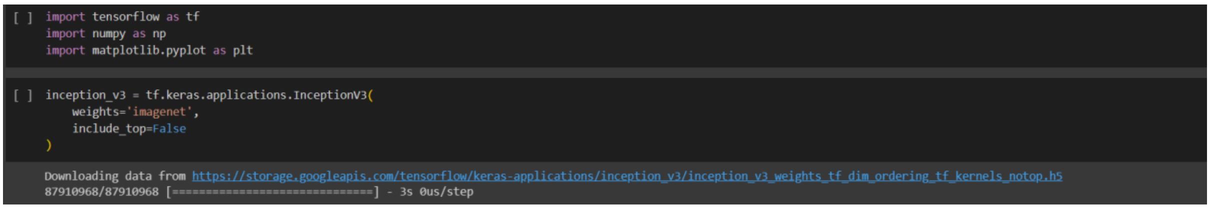


```
for i in range(5):
    print(classification[list_index[i]], ':', round(predictions[0][list_index[i]] * 100, 2), '%')
```

```
... bee : 59.22 %
    lion : 17.0 %
    frog : 9.85 %
    bird : 9.69 %
    dog : 1.85 %
```

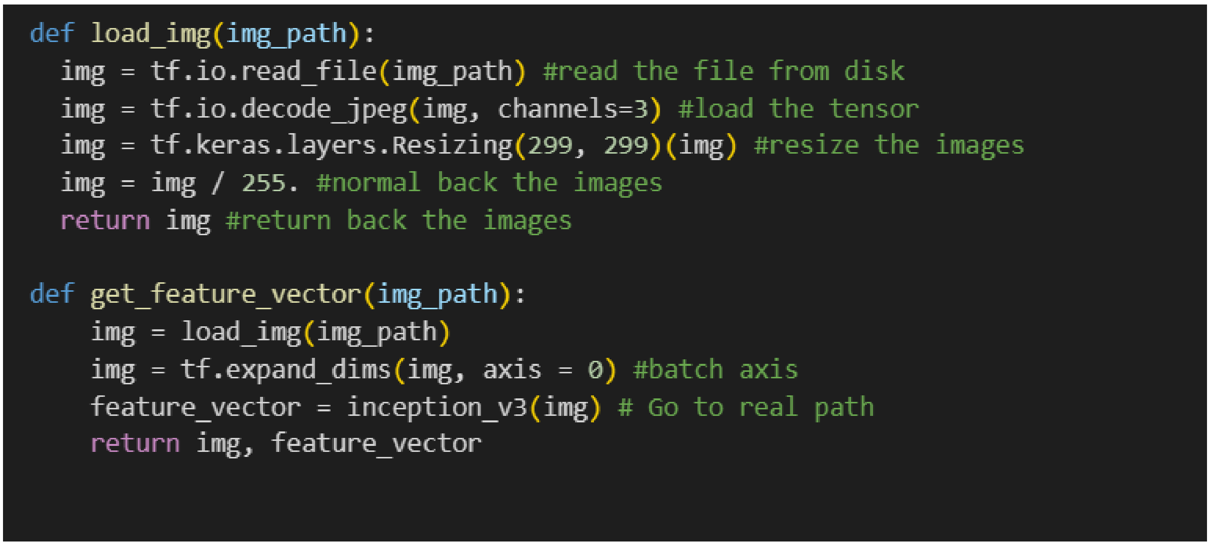
Show the picture inside the file to show every prediction of the image with percentage.

Image Caption



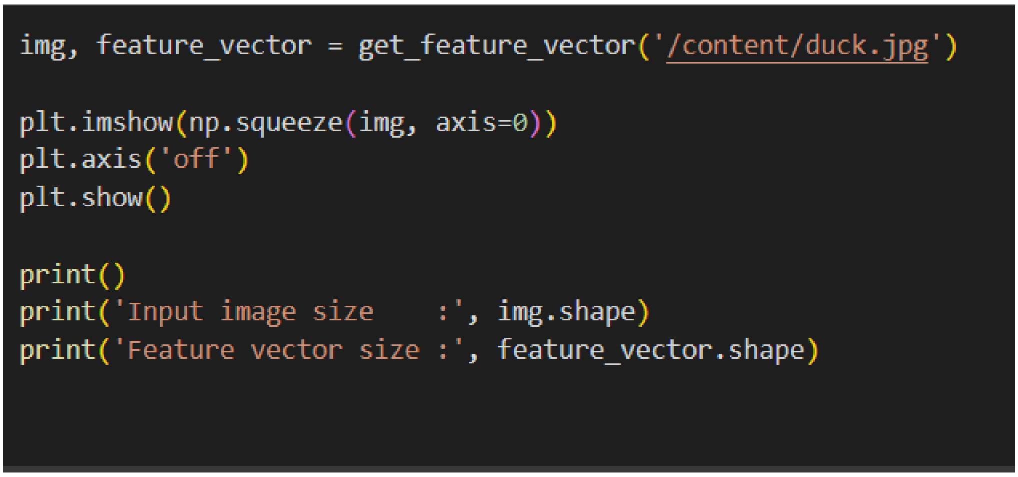
```
[ ] import tensorflow as tf
    import numpy as np
    import matplotlib.pyplot as plt

[ ] inception_v3 = tf.keras.applications.InceptionV3(
        weights='imagenet',
        include_top=False
    )

    Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/inception_v3/inception_v3_weights_tf_dim_ordering_tf_kernels_notop.h5
    87910968/87910968 [==============================] - 3s 0us/step
```

Import library necessaries and use inception_v3 with keras application of imagenet weights to download data from googleapis.



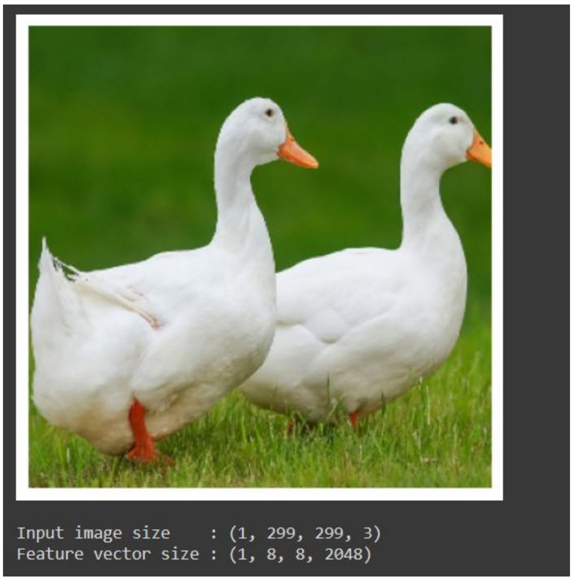
```
def load_img(img_path):
  img = tf.io.read_file(img_path) #read the file from disk
  img = tf.io.decode_jpeg(img, channels=3) #load the tensor
  img = tf.keras.layers.Resizing(299, 299)(img) #resize the images
  img = img / 255. #normal back the images
  return img #return back the images

def get_feature_vector(img_path):
    img = load_img(img_path)
    img = tf.expand_dims(img, axis = 0) #batch axis
    feature_vector = inception_v3(img) # Go to real path
    return img, feature_vector
```

The system will read the file from disk and load the tensor. After that, start to resize the images and normal back the images. If complete, return back images batch the axis goes to real path with feature vector.



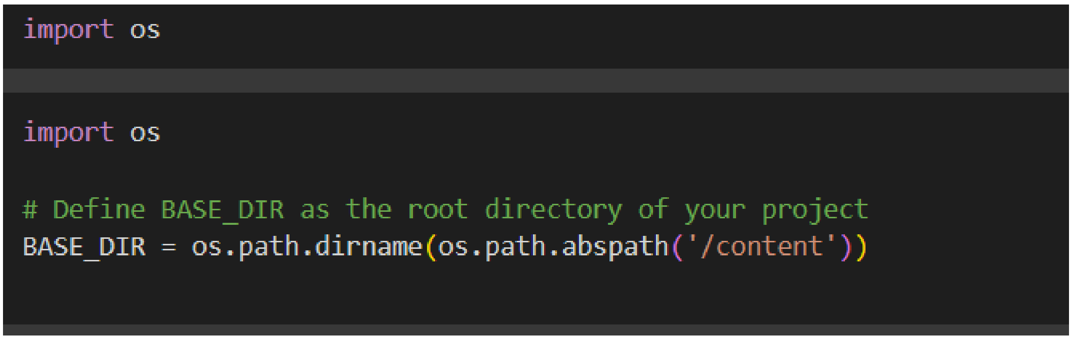
```
img, feature_vector = get_feature_vector('/content/duck.jpg')

plt.imshow(np.squeeze(img, axis=0))
plt.axis('off')
plt.show()

print()
print('Input image size    :', img.shape)
print('Feature vector size :', feature_vector.shape)
```

After that upload whatever images from laptop as testing at google colab which store at content part and show the image with details of input image size and feature vector size.



```
Input image size    : (1, 299, 299, 3)
Feature vector size : (1, 8, 8, 2048)
```

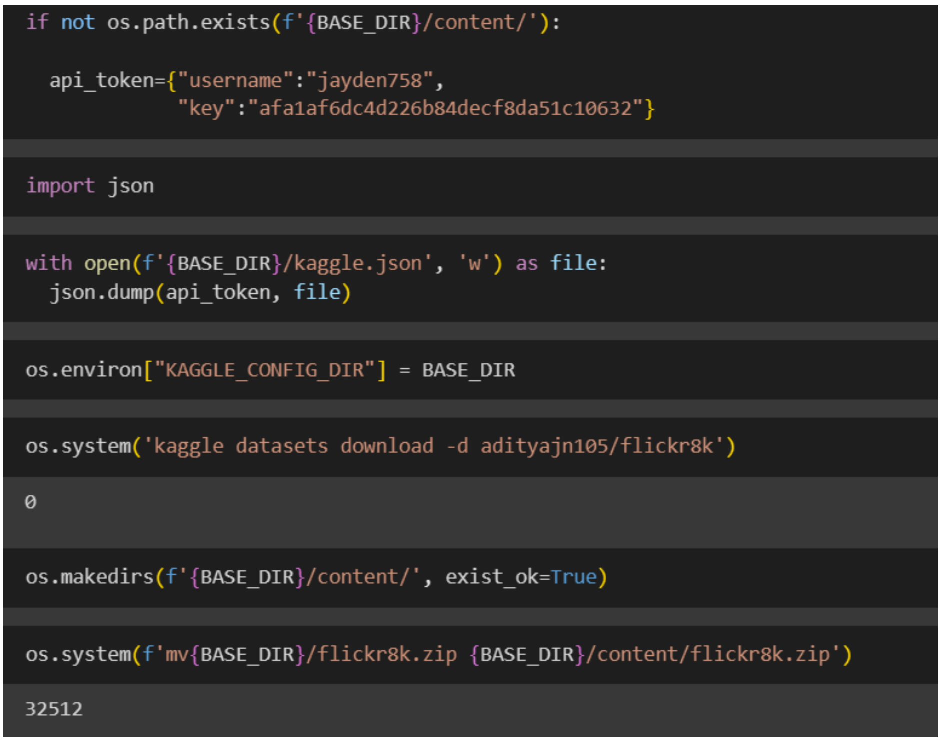
Details of input image size and feature vector size.



```
import os
```

```
import os

# Define BASE_DIR as the root directory of your project
BASE_DIR = os.path.dirname(os.path.abspath('/content'))
```

Import the os and define the BASE_DIR as root directory of project in case to save the Kaggle datasets of flickr8k.



```
if not os.path.exists(f'{BASE_DIR}/content/'):

  api_token={"username":"jayden758",
             "key":"afa1af6dc4d226b84decf8da51c10632"}
```

```
import json
```

```
with open(f'{BASE_DIR}/kaggle.json', 'w') as file:
  json.dump(api_token, file)
```

```
os.environ["KAGGLE_CONFIG_DIR"] = BASE_DIR
```

```
os.system('kaggle datasets download -d adityajn105/flickr8k')
```

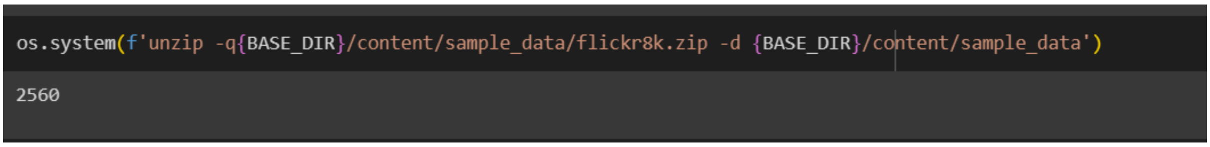
```
0
```

```
os.makedirs(f'{BASE_DIR}/content/', exist_ok=True)
```

```
os.system(f'mv{BASE_DIR}/flickr8k.zip {BASE_DIR}/content/flickr8k.zip')
```

```
32512
```

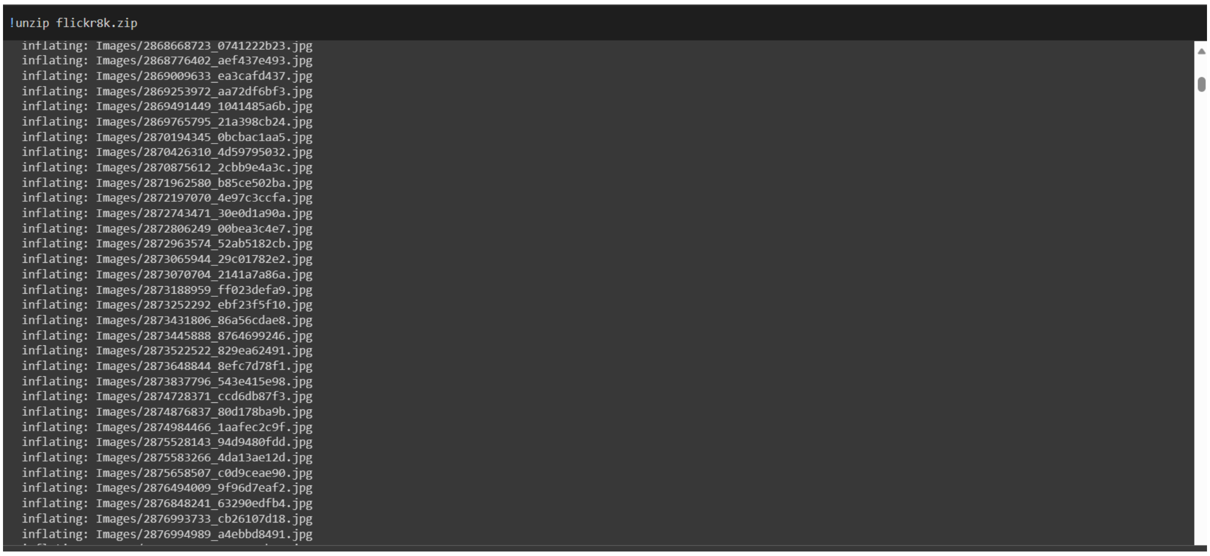
Start to download the Kaggle datasets. Before download dataset, need create an account at Kaggle to get the api_token which include username and key id. Configure the Kaggle, with import os and start download datasets from user of adityajn105, after exists will at directory path and import the datasets with zip file.



```
os.system(f'unzip -q{BASE_DIR}/content/sample_data/flickr8k.zip -d {BASE_DIR}/content/sample_data')
```

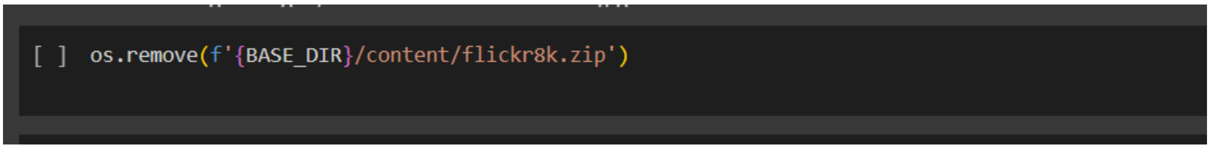
```
2560
```

Start to get system unzip the data file which store at directory file which at BASE_DIR of content/sample data locations.



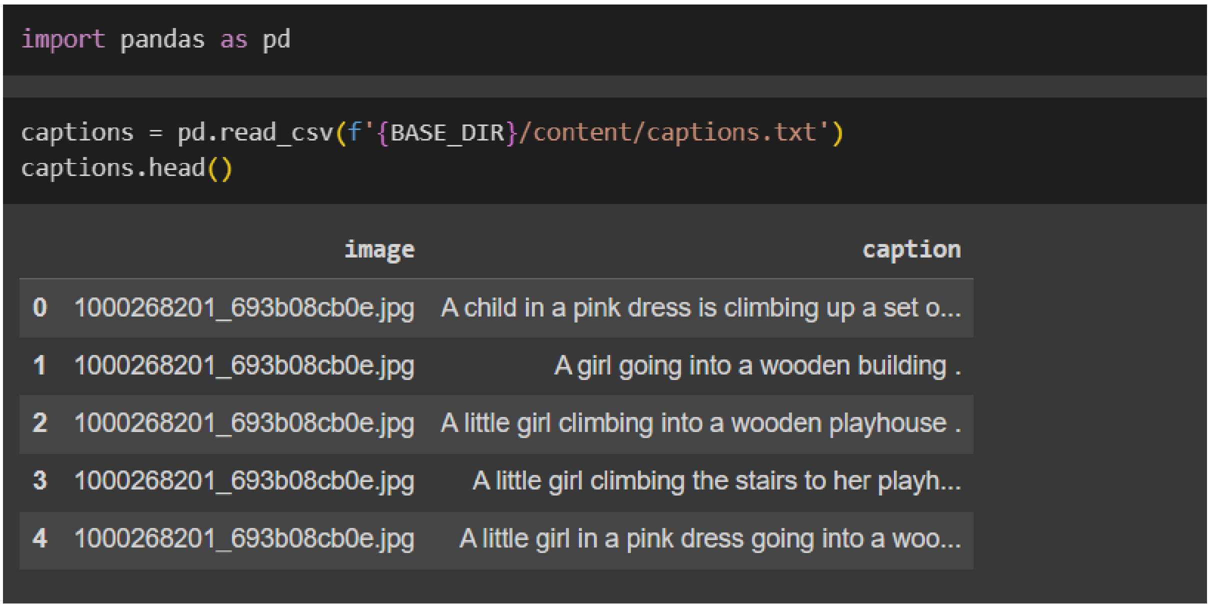
```
!unzip flickr8k.zip
  inflating: Images/2868668723_0741222b23.jpg
  inflating: Images/2868776402_aef437e493.jpg
  inflating: Images/2869009633_ea3cafd437.jpg
  inflating: Images/2869253972_aa72df6bf3.jpg
  inflating: Images/2869491449_1041485a6b.jpg
  inflating: Images/2869765795_21a398cb24.jpg
  inflating: Images/2870194345_0bcbac1aa5.jpg
  inflating: Images/2870426310_4d59795032.jpg
  inflating: Images/2870875612_2cbb9e4a3c.jpg
  inflating: Images/2871962580_b85ce502ba.jpg
  inflating: Images/2872197070_4e97c3ccfa.jpg
  inflating: Images/2872743471_30e0d1a90a.jpg
  inflating: Images/2872806249_00bea3c4e7.jpg
  inflating: Images/2872963574_52ab5182cb.jpg
  inflating: Images/2873065944_29c01782e2.jpg
  inflating: Images/2873070704_2141a7a86a.jpg
  inflating: Images/2873188959_ff023defa9.jpg
  inflating: Images/2873252292_ebf23f5f10.jpg
  inflating: Images/2873431806_86a56cdae8.jpg
  inflating: Images/2873445888_8764699246.jpg
  inflating: Images/2873522522_829ea62491.jpg
  inflating: Images/2873648844_8efc7d78f1.jpg
  inflating: Images/2873837796_543e415e98.jpg
  inflating: Images/2874728371_ccd6db87f3.jpg
  inflating: Images/2874876837_80d178ba9b.jpg
  inflating: Images/2874984466_1aafec2c9f.jpg
  inflating: Images/2875528143_94d9480fdd.jpg
  inflating: Images/2875583266_4da13ae12d.jpg
  inflating: Images/2875658507_c0d9ceae90.jpg
  inflating: Images/2876494009_9f96d7eaf2.jpg
  inflating: Images/2876848241_63290edfb4.jpg
  inflating: Images/2876993733_cb26107d18.jpg
  inflating: Images/2876994989_a4ebbd8491.jpg
```

With the unzip command input, system start to unzip the file and show progress the images and captions text file start unzip from file.



```
[ ]  os.remove(f'{BASE_DIR}/content/flickr8k.zip')
```

After unzipping successful, the data all located at path. Start to remove the zip file.

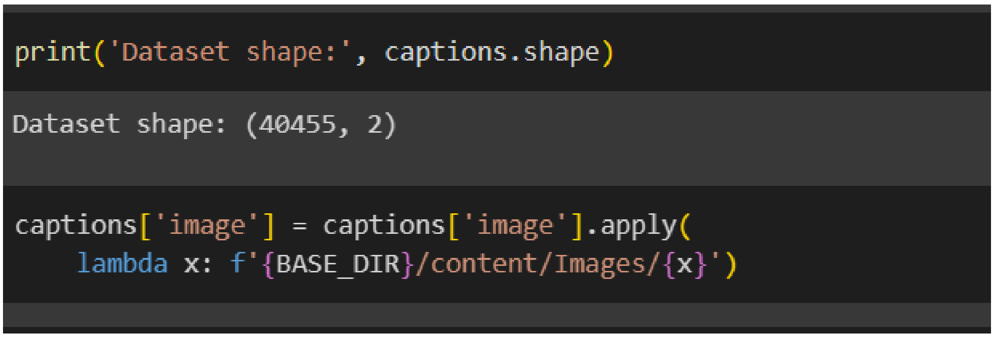


```
import pandas as pd
```

```
captions = pd.read_csv(f'{BASE_DIR}/content/captions.txt')
captions.head()
```

	image	caption
0	1000268201_693b08cb0e.jpg	A child in a pink dress is climbing up a set o...
1	1000268201_693b08cb0e.jpg	A girl going into a wooden building .
2	1000268201_693b08cb0e.jpg	A little girl climbing into a wooden playhouse .
3	1000268201_693b08cb0e.jpg	A little girl climbing the stairs to her playh...
4	1000268201_693b08cb0e.jpg	A little girl in a pink dress going into a woo...

Import pandas as pd to get system know the libraries. Type the command captions with read the file images from the path we store. It will show id, name of images and caption that suite with the images.

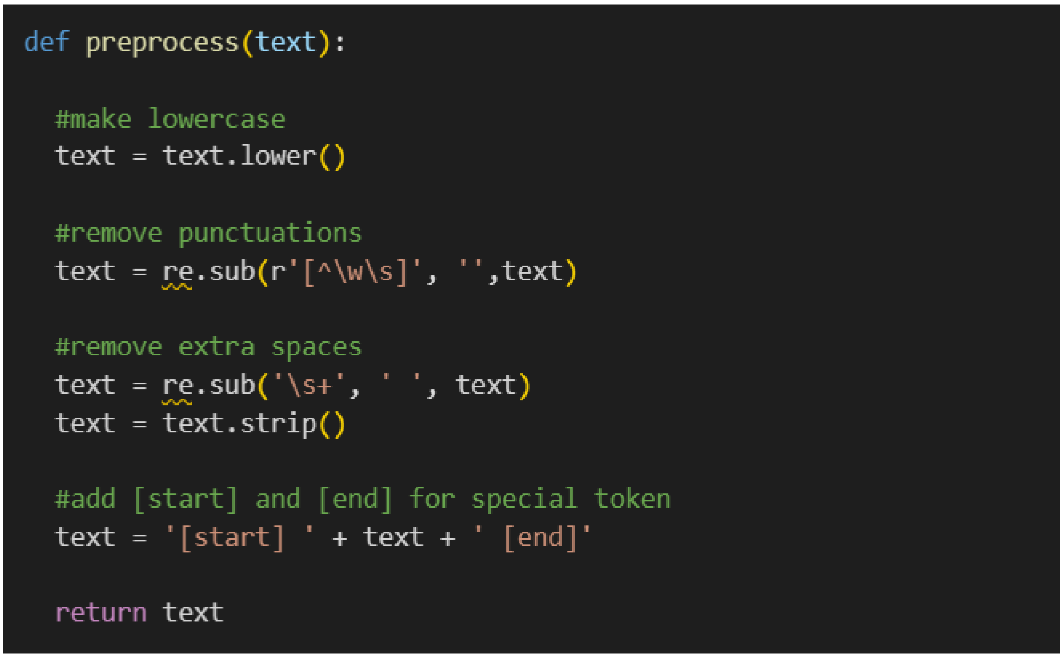


```
print('Dataset shape:', captions.shape)
```

```
Dataset shape: (40455, 2)
```

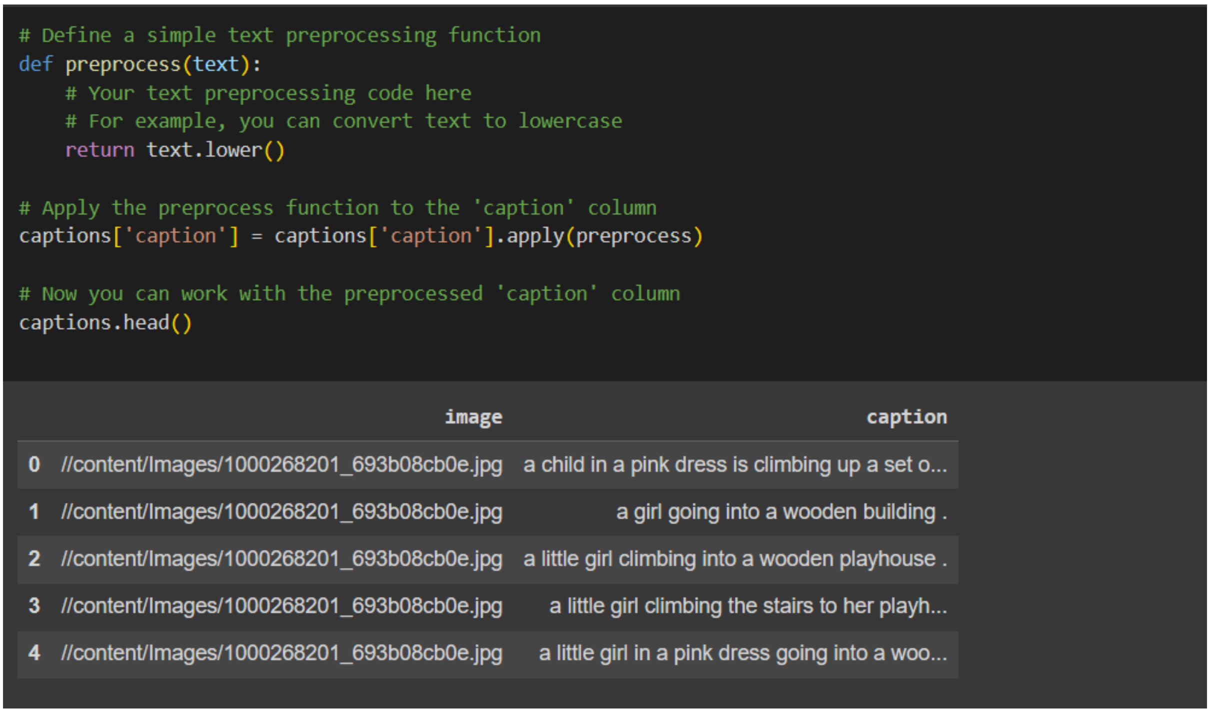
```
captions['image'] = captions['image'].apply(
    lambda x: f'{BASE_DIR}/content/Images/{x}')
```

Print Dataset shape with captions together and applied lambda x to the path of images.



```
def preprocess(text):

  #make lowercase
  text = text.lower()

  #remove punctuations
  text = re.sub(r'[^\w\s]', '',text)

  #remove extra spaces
  text = re.sub('\s+', ' ', text)
  text = text.strip()

  #add [start] and [end] for special token
  text = '[start] ' + text + ' [end]'

  return text
```

Set the captions text format which include make lowercase, remove punctuation, remove extra spaces, and add [start] and [end] at front and behind of sentence.

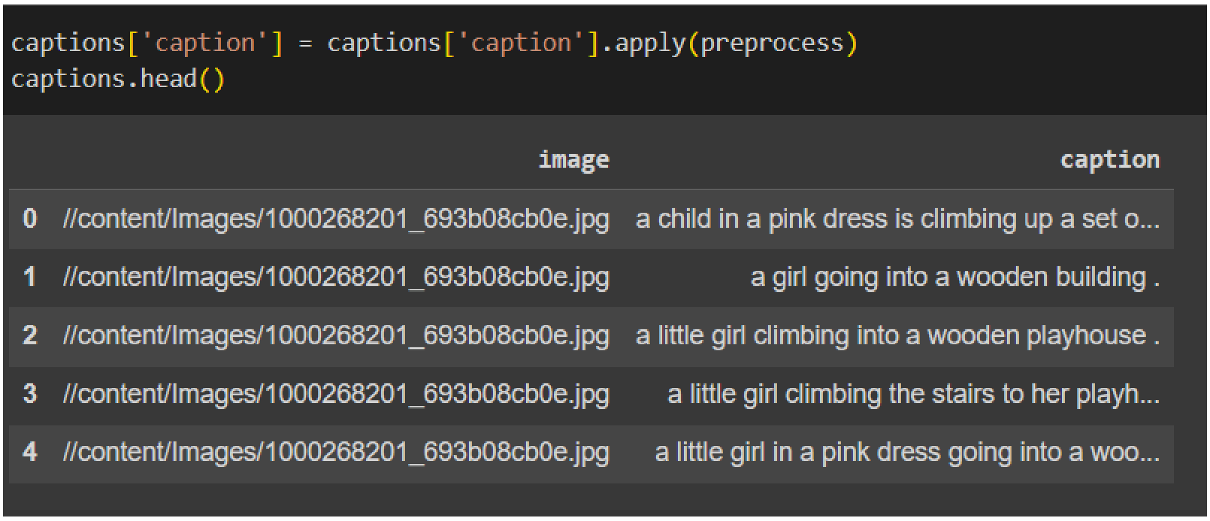


```python
# Define a simple text preprocessing function
def preprocess(text):
    # Your text preprocessing code here
    # For example, you can convert text to lowercase
    return text.lower()

# Apply the preprocess function to the 'caption' column
captions['caption'] = captions['caption'].apply(preprocess)

# Now you can work with the preprocessed 'caption' column
captions.head()
```

	image	caption
0	//content/Images/1000268201_693b08cb0e.jpg	a child in a pink dress is climbing up a set o...
1	//content/Images/1000268201_693b08cb0e.jpg	a girl going into a wooden building .
2	//content/Images/1000268201_693b08cb0e.jpg	a little girl climbing into a wooden playhouse .
3	//content/Images/1000268201_693b08cb0e.jpg	a little girl climbing the stairs to her playh...
4	//content/Images/1000268201_693b08cb0e.jpg	a little girl in a pink dress going into a woo...

Define a simple text preprocessing function. For example, lowercase of text, apply the preprocess function to ‘caption’ column, and work with preprocessed ‘caption’ column.

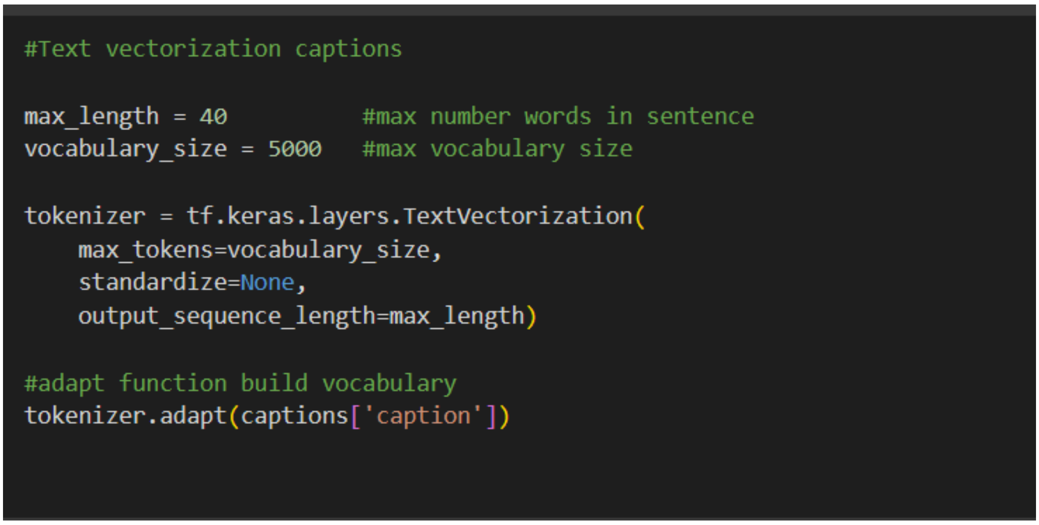
The above diagram is showing image and caption together follow the function we set.



```python
captions['caption'] = captions['caption'].apply(preprocess)
captions.head()
```

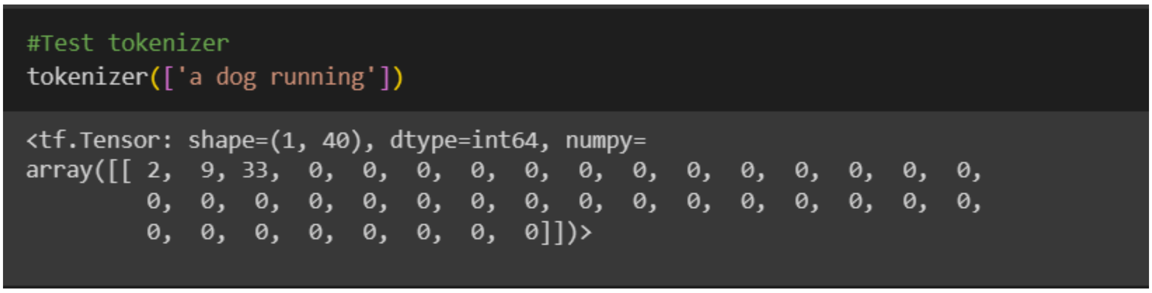
	image	caption
0	//content/Images/1000268201_693b08cb0e.jpg	a child in a pink dress is climbing up a set o...
1	//content/Images/1000268201_693b08cb0e.jpg	a girl going into a wooden building .
2	//content/Images/1000268201_693b08cb0e.jpg	a little girl climbing into a wooden playhouse .
3	//content/Images/1000268201_693b08cb0e.jpg	a little girl climbing the stairs to her playh...
4	//content/Images/1000268201_693b08cb0e.jpg	a little girl in a pink dress going into a woo...

Preprocess the images again.



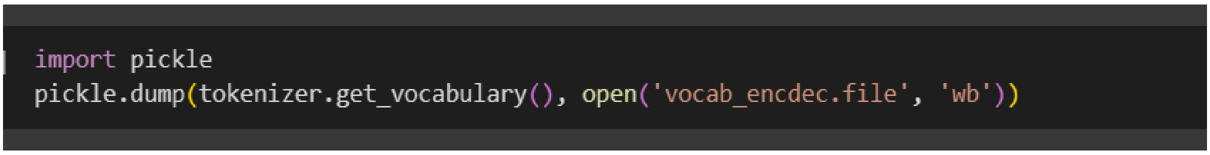
```
#Text vectorization captions

max_length = 40           #max number words in sentence
vocabulary_size = 5000    #max vocabulary size

tokenizer = tf.keras.layers.TextVectorization(
    max_tokens=vocabulary_size,
    standardize=None,
    output_sequence_length=max_length)

#adapt function build vocabulary
tokenizer.adapt(captions['caption'])
```

Text vectorization captions with set the max number words in a sentence can appear, I set as 40 max length and vocabulary size with 5000. Adapt function tokenizer to build vocabulary.



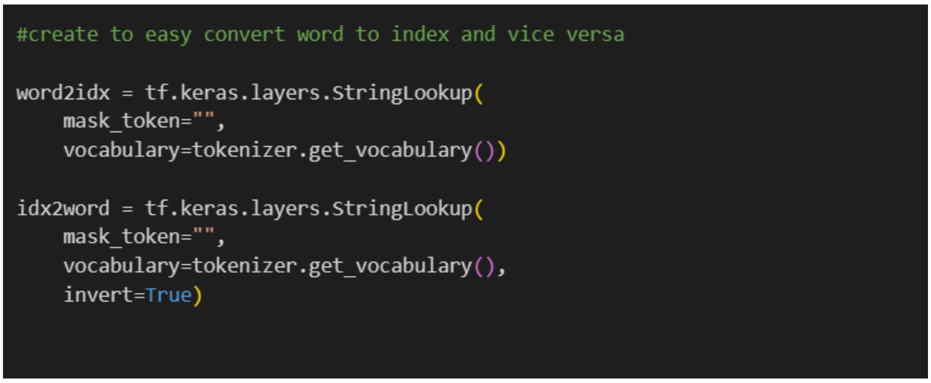
```
#Test tokenizer
tokenizer(['a dog running'])

<tf.Tensor: shape=(1, 40), dtype=int64, numpy=
array([[ 2,  9, 33,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
         0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
         0,  0,  0,  0,  0,  0,  0,  0]])>
```

Test tokenizer.



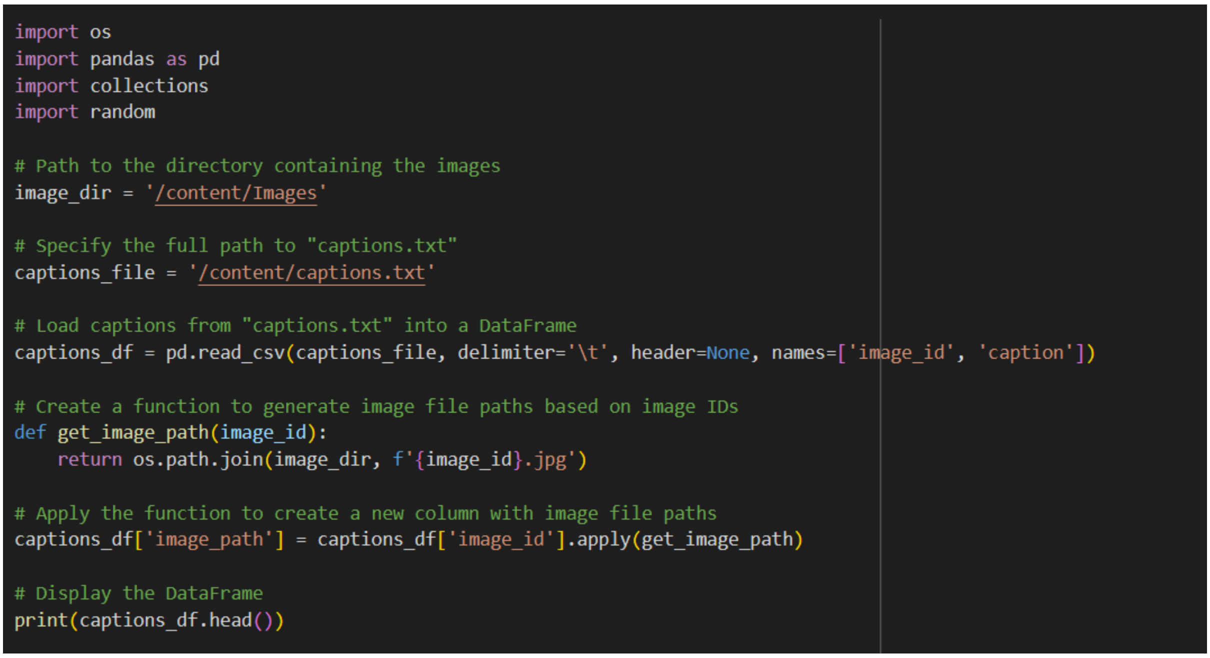
```
import pickle
pickle.dump(tokenizer.get_vocabulary(), open('vocab_encdec.file', 'wb'))
```

Import pickle file to tokenizer and get vocabulary with open encoder and decoder file.

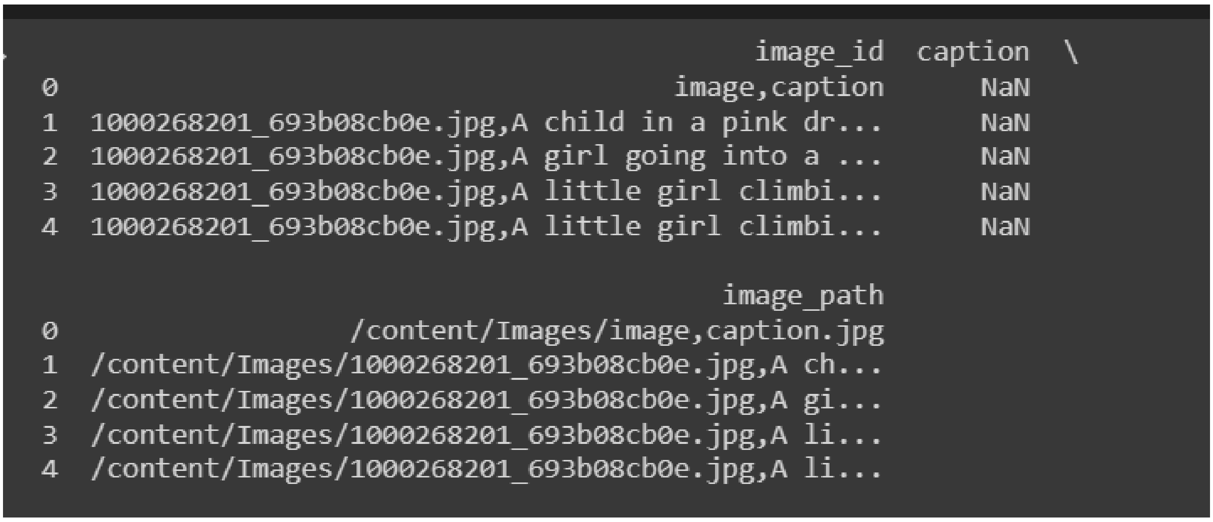


```
#create to easy convert word to index and vice versa

word2idx = tf.keras.layers.StringLookup(
    mask_token="",
    vocabulary=tokenizer.get_vocabulary())

idx2word = tf.keras.layers.StringLookup(
    mask_token="",
    vocabulary=tokenizer.get_vocabulary(),
    invert=True)
```

Start creating for easy convert the word to index and vice versa.

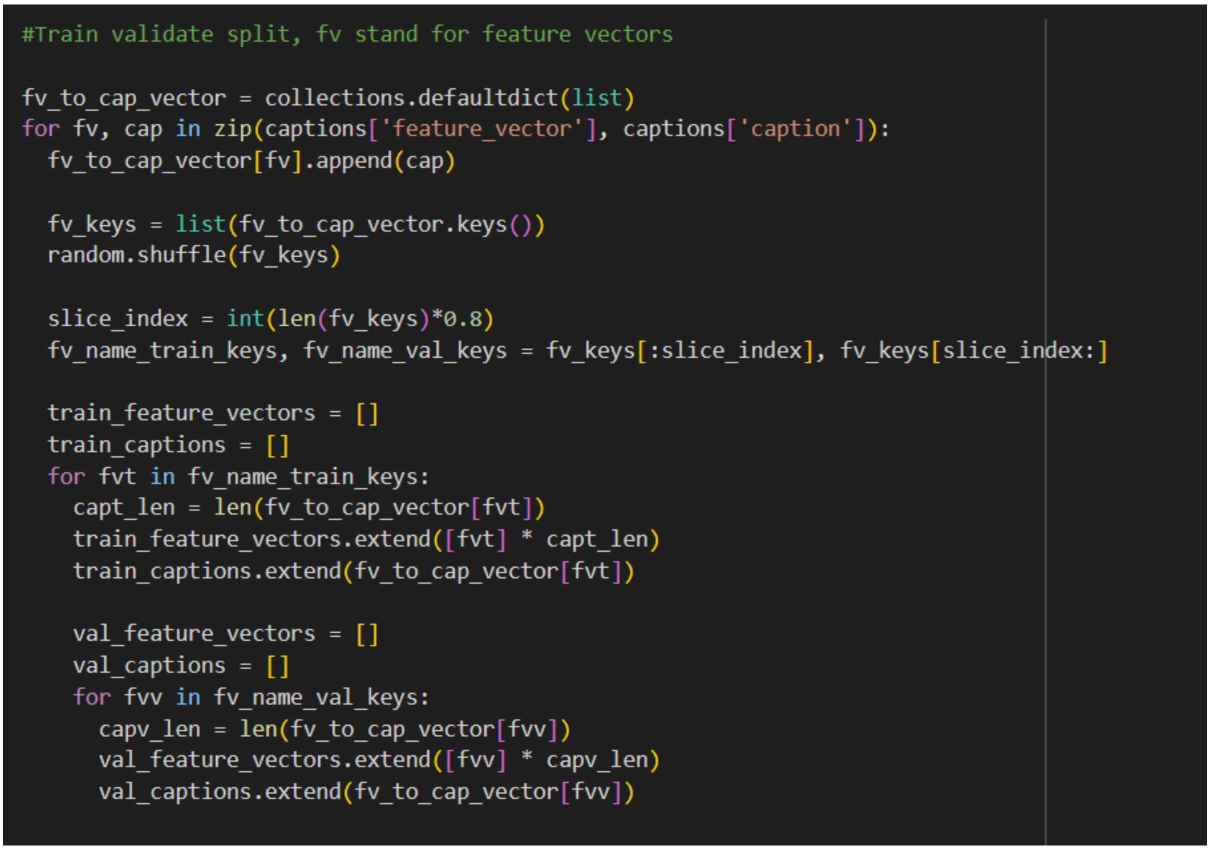


```
import os
import pandas as pd
import collections
import random

# Path to the directory containing the images
image_dir = '/content/Images'

# Specify the full path to "captions.txt"
captions_file = '/content/captions.txt'

# Load captions from "captions.txt" into a DataFrame
captions_df = pd.read_csv(captions_file, delimiter='\t', header=None, names=['image_id', 'caption'])

# Create a function to generate image file paths based on image IDs
def get_image_path(image_id):
    return os.path.join(image_dir, f'{image_id}.jpg')

# Apply the function to create a new column with image file paths
captions_df['image_path'] = captions_df['image_id'].apply(get_image_path)

# Display the DataFrame
print(captions_df.head())
```



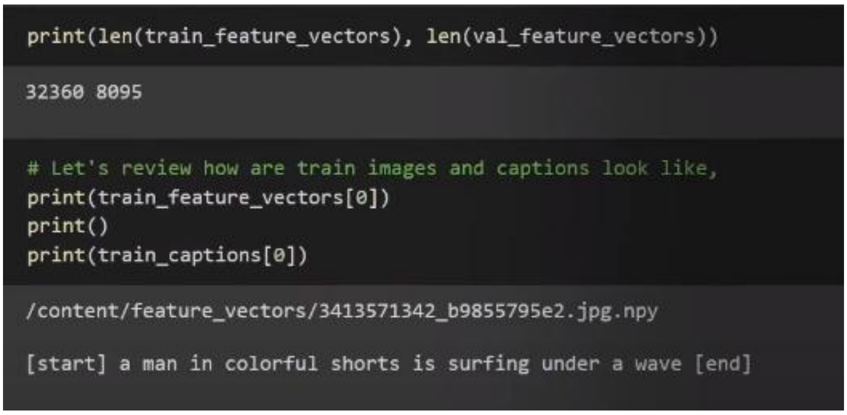
```
                                              image_id  caption  \
0                                        image,caption      NaN
1  1000268201_693b08cb0e.jpg,A child in a pink dr...      NaN
2  1000268201_693b08cb0e.jpg,A girl going into a ...      NaN
3  1000268201_693b08cb0e.jpg,A little girl climbi...      NaN
4  1000268201_693b08cb0e.jpg,A little girl climbi...      NaN

                                            image_path
0                /content/Images/image,caption.jpg
1  /content/Images/1000268201_693b08cb0e.jpg,A ch...
2  /content/Images/1000268201_693b08cb0e.jpg,A gi...
3  /content/Images/1000268201_693b08cb0e.jpg,A li...
4  /content/Images/1000268201_693b08cb0e.jpg,A li...
```

Import necessary libraries and path to the directory that contain the images with select the path that correctly place the images files, specify the full path to captions text file, load the captions from captions text file into a Data frame and to create a function generate the image file paths based on image id, apply the function captions to create a new column with image file paths and lastly display the data frame.



```
#Train validate split, fv stand for feature vectors

fv_to_cap_vector = collections.defaultdict(list)
for fv, cap in zip(captions['feature_vector'], captions['caption']):
  fv_to_cap_vector[fv].append(cap)

  fv_keys = list(fv_to_cap_vector.keys())
  random.shuffle(fv_keys)

  slice_index = int(len(fv_keys)*0.8)
  fv_name_train_keys, fv_name_val_keys = fv_keys[:slice_index], fv_keys[slice_index:]

  train_feature_vectors = []
  train_captions = []
  for fvt in fv_name_train_keys:
    capt_len = len(fv_to_cap_vector[fvt])
    train_feature_vectors.extend([fvt] * capt_len)
    train_captions.extend(fv_to_cap_vector[fvt])

    val_feature_vectors = []
    val_captions = []
    for fvv in fv_name_val_keys:
      capv_len = len(fv_to_cap_vector[fvv])
      val_feature_vectors.extend([fvv] * capv_len)
      val_captions.extend(fv_to_cap_vector[fvv])
```

Train the validate split, which fv stand for feature vectors.



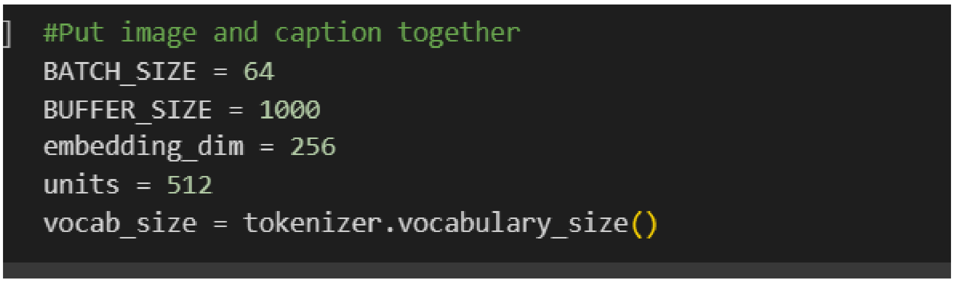
```
print(len(train_feature_vectors), len(val_feature_vectors))
```

```
32360 8095
```

```
# Let's review how are train images and captions look like,
print(train_feature_vectors[0])
print()
print(train_captions[0])
```

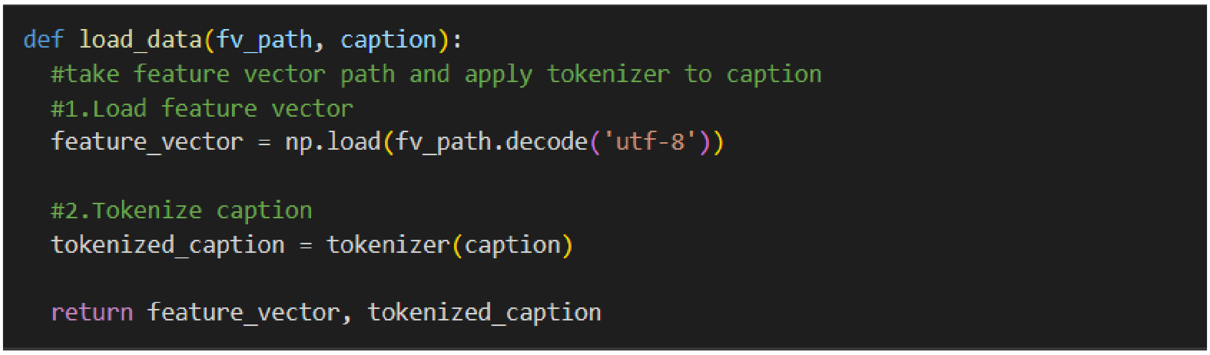
```
/content/feature_vectors/3413571342_b9855795e2.jpg.npy

[start] a man in colorful shorts is surfing under a wave [end]
```

Print the result with train and val feature vectors datasets. After that, train the image and caption like this.



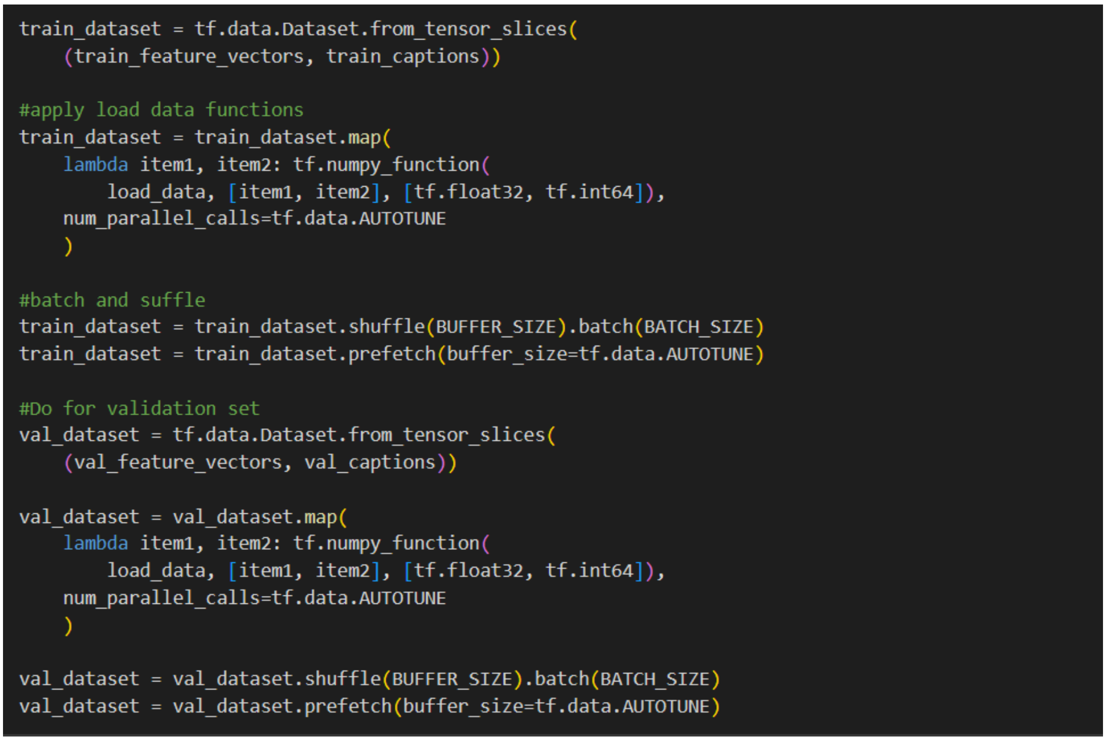
```
]  #Put image and caption together
   BATCH_SIZE = 64
   BUFFER_SIZE = 1000
   embedding_dim = 256
   units = 512
   vocab_size = tokenizer.vocabulary_size()
```

Put the image and caption together that shows for user with following details.



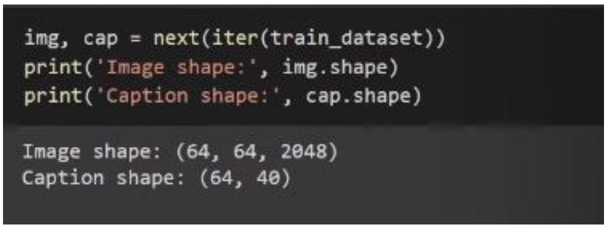
```
def load_data(fv_path, caption):
  #take feature vector path and apply tokenizer to caption
  #1.Load feature vector
  feature_vector = np.load(fv_path.decode('utf-8'))

  #2.Tokenize caption
  tokenized_caption = tokenizer(caption)

  return feature_vector, tokenized_caption
```

The image will take feature vector path and caption string and return the loaded feature vector array and apply the tokenizer to the caption which load and tokenize it.



```
train_dataset = tf.data.Dataset.from_tensor_slices(
    (train_feature_vectors, train_captions))

#apply load data functions
train_dataset = train_dataset.map(
    lambda item1, item2: tf.numpy_function(
        load_data, [item1, item2], [tf.float32, tf.int64]),
    num_parallel_calls=tf.data.AUTOTUNE
    )

#batch and suffle
train_dataset = train_dataset.shuffle(BUFFER_SIZE).batch(BATCH_SIZE)
train_dataset = train_dataset.prefetch(buffer_size=tf.data.AUTOTUNE)

#Do for validation set
val_dataset = tf.data.Dataset.from_tensor_slices(
    (val_feature_vectors, val_captions))

val_dataset = val_dataset.map(
    lambda item1, item2: tf.numpy_function(
        load_data, [item1, item2], [tf.float32, tf.int64]),
    num_parallel_calls=tf.data.AUTOTUNE
    )

val_dataset = val_dataset.shuffle(BUFFER_SIZE).batch(BATCH_SIZE)
val_dataset = val_dataset.prefetch(buffer_size=tf.data.AUTOTUNE)
```

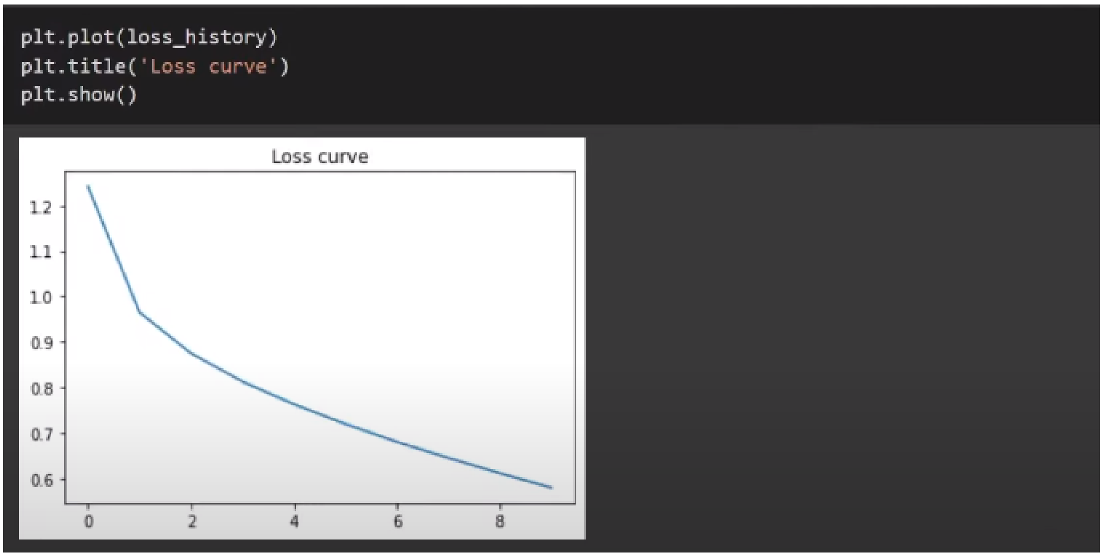
Train the datasets with feature vectors and train captions with apply load functions, batch, and shuffle, and do for the validations set datasets.



```
img, cap = next(iter(train_dataset))
print('Image shape:', img.shape)
print('Caption shape:', cap.shape)
```

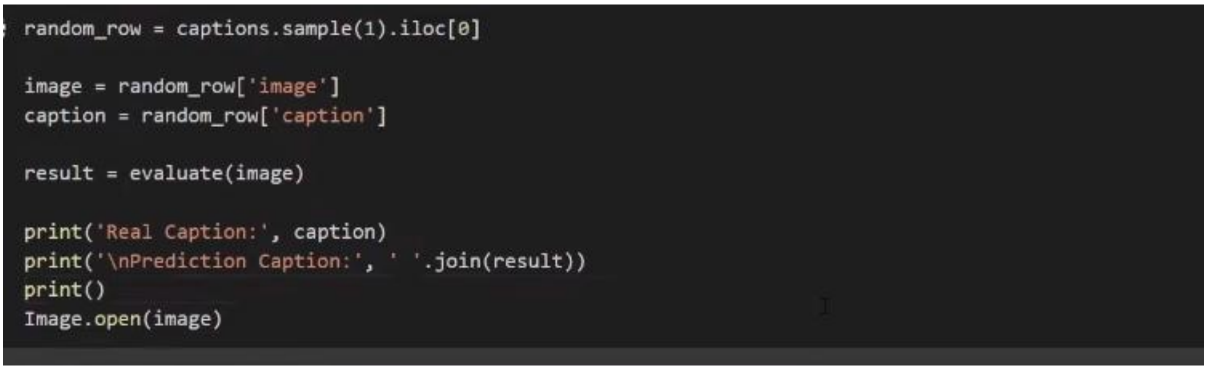
```
Image shape: (64, 64, 2048)
Caption shape: (64, 40)
```

Show image shape and caption shape.



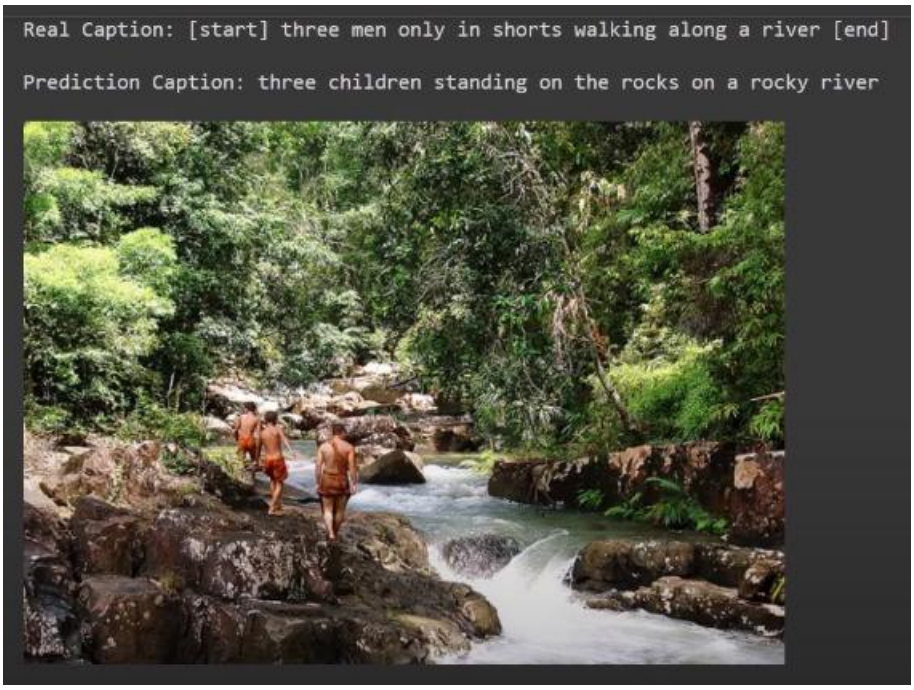
```
plt.plot(loss_history)
plt.title('Loss curve')
plt.show()
```

This is the graph of loss curve of accuracy datasets with images and caption process with loss history.



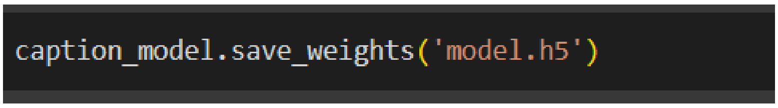
```
random_row = captions.sample(1).iloc[0]

image = random_row['image']
caption = random_row['caption']

result = evaluate(image)

print('Real Caption:', caption)
print('\nPrediction Caption:', ' '.join(result))
print()
Image.open(image)
```

Insert these coding to let system evaluate the image and random provide image with caption that suitable with the images.



```
Real Caption: [start] three men only in shorts walking along a river [end]

Prediction Caption: three children standing on the rocks on a rocky river
```

This is output for the random images and captions provided.

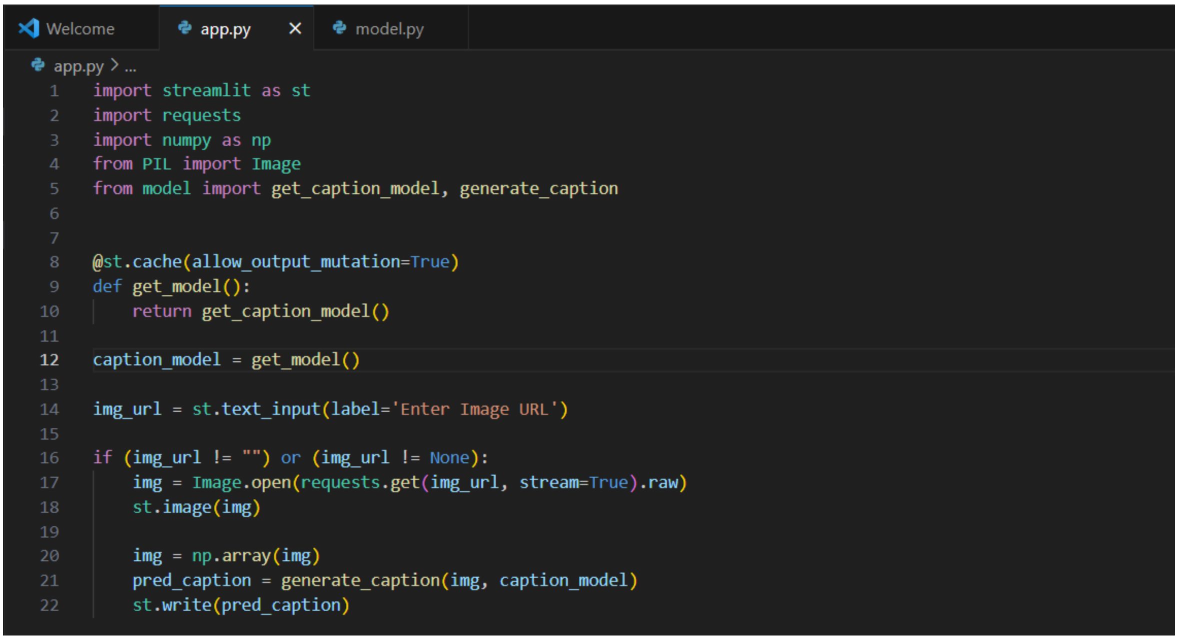


```python
caption_model.save_weights('model.h5')
```

Load the model and save the model weights in the streamlit environment and generate the captions.

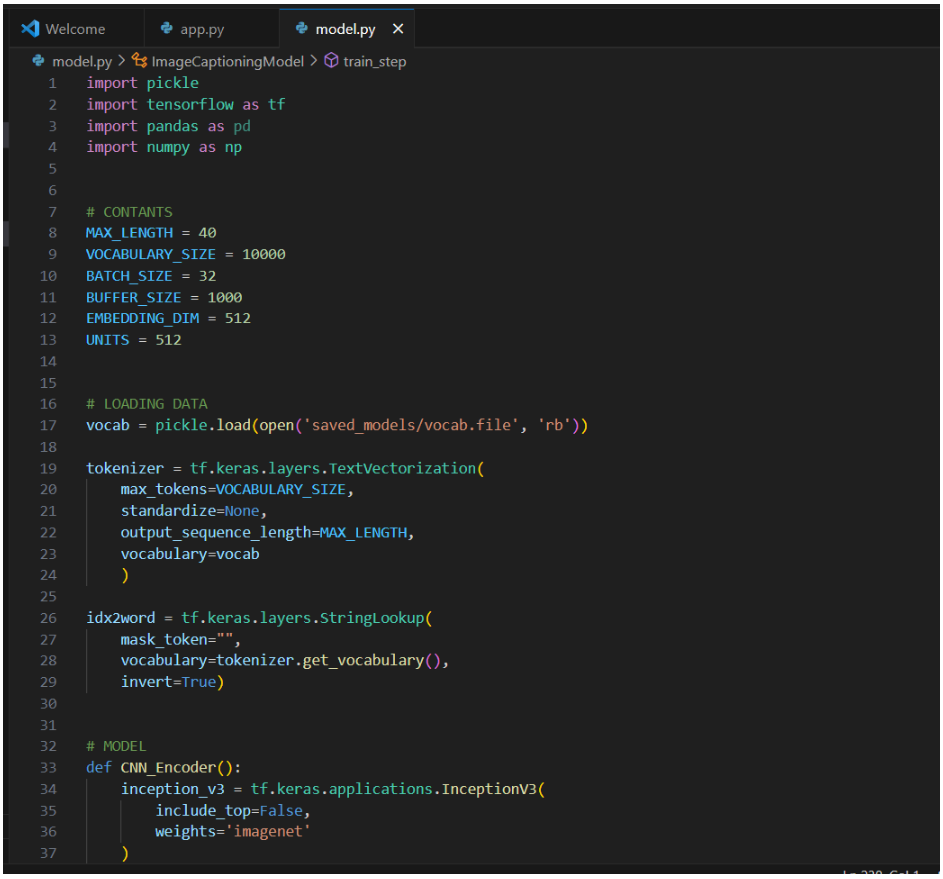
Visual studio code (Python file)

app.py

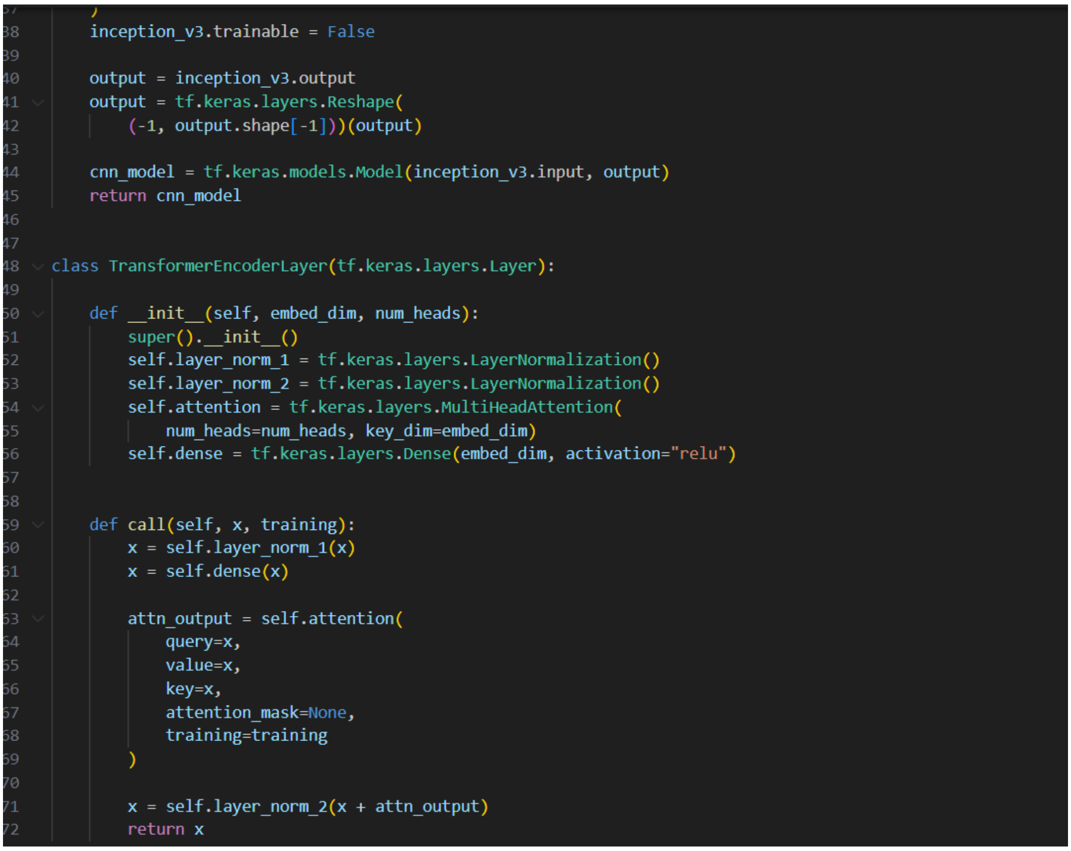


```python
import streamlit as st
import requests
import numpy as np
from PIL import Image
from model import get_caption_model, generate_caption


@st.cache(allow_output_mutation=True)
def get_model():
    return get_caption_model()

caption_model = get_model()

img_url = st.text_input(label='Enter Image URL')

if (img_url != "") or (img_url != None):
    img = Image.open(requests.get(img_url, stream=True).raw)
    st.image(img)

    img = np.array(img)
    pred_caption = generate_caption(img, caption_model)
    st.write(pred_caption)
```

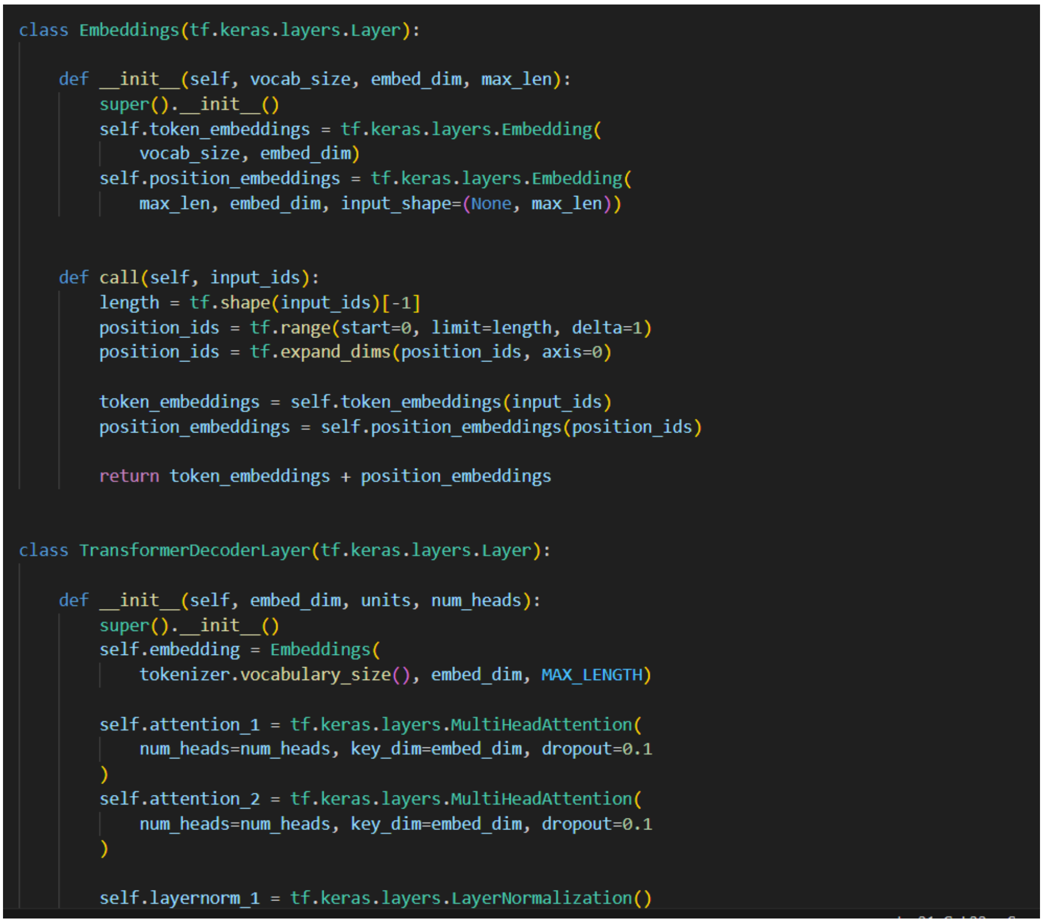
model.py



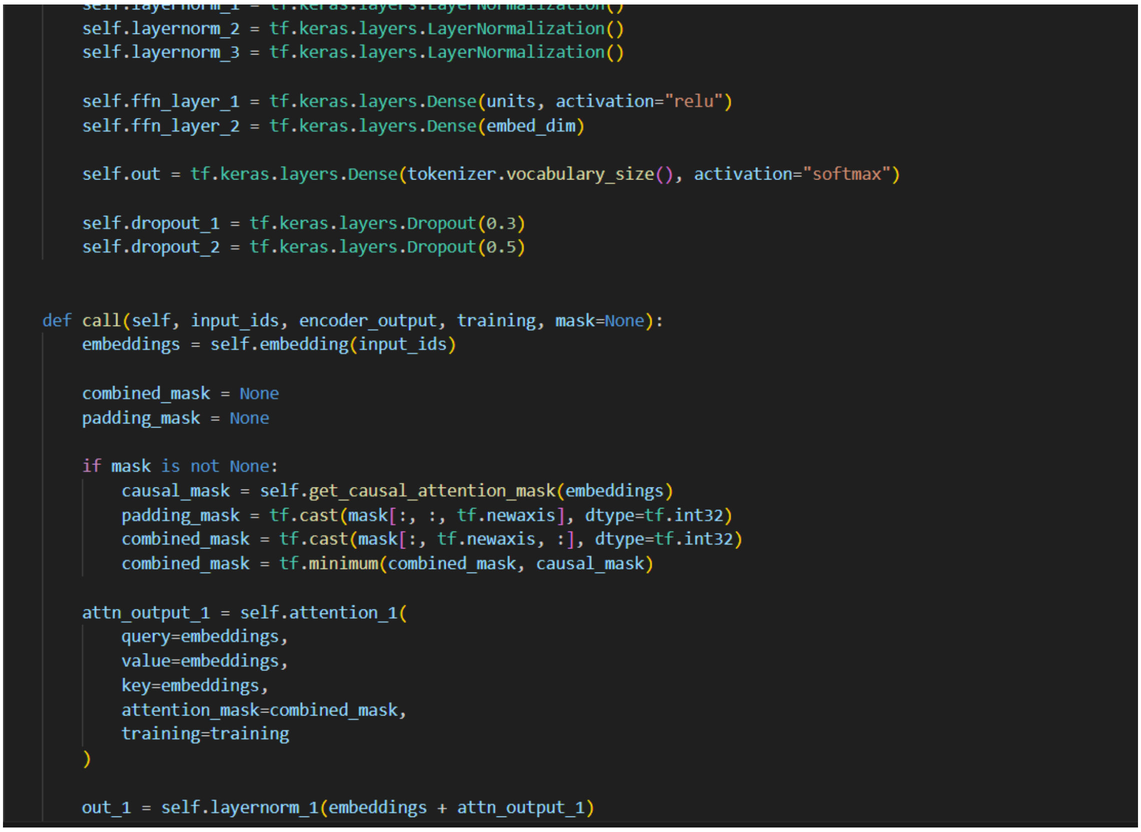
```python
import pickle
import tensorflow as tf
import pandas as pd
import numpy as np


# CONTANTS
MAX_LENGTH = 40
VOCABULARY_SIZE = 10000
BATCH_SIZE = 32
BUFFER_SIZE = 1000
EMBEDDING_DIM = 512
UNITS = 512


# LOADING DATA
vocab = pickle.load(open('saved_models/vocab.file', 'rb'))

tokenizer = tf.keras.layers.TextVectorization(
    max_tokens=VOCABULARY_SIZE,
    standardize=None,
    output_sequence_length=MAX_LENGTH,
    vocabulary=vocab
    )

idx2word = tf.keras.layers.StringLookup(
    mask_token="",
    vocabulary=tokenizer.get_vocabulary(),
    invert=True)


# MODEL
def CNN_Encoder():
    inception_v3 = tf.keras.applications.InceptionV3(
        include_top=False,
        weights='imagenet'
    )
```



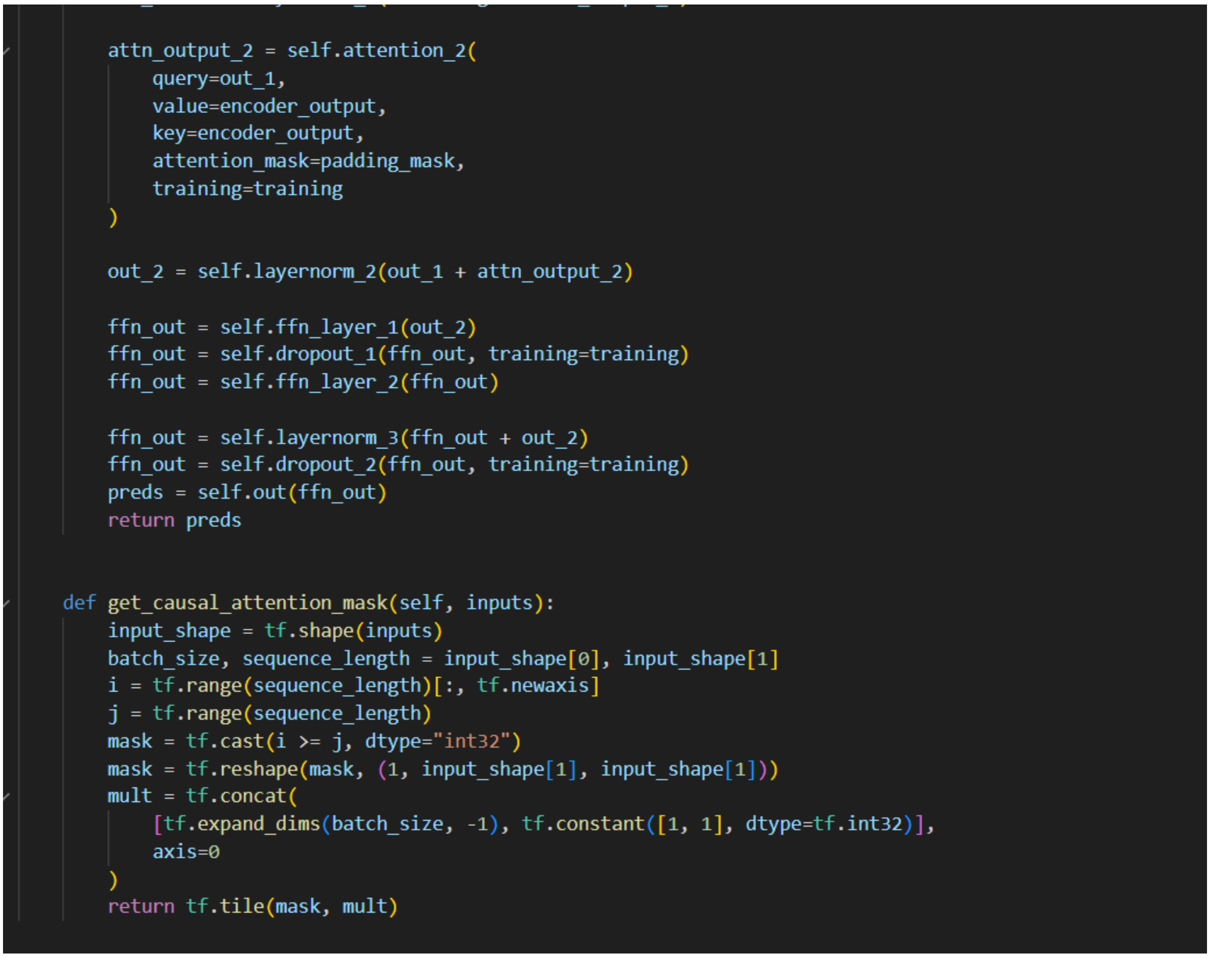
```python
    )
    inception_v3.trainable = False

    output = inception_v3.output
    output = tf.keras.layers.Reshape(
        (-1, output.shape[-1]))(output)

    cnn_model = tf.keras.models.Model(inception_v3.input, output)
    return cnn_model


class TransformerEncoderLayer(tf.keras.layers.Layer):

    def __init__(self, embed_dim, num_heads):
        super().__init__()
        self.layer_norm_1 = tf.keras.layers.LayerNormalization()
        self.layer_norm_2 = tf.keras.layers.LayerNormalization()
        self.attention = tf.keras.layers.MultiHeadAttention(
            num_heads=num_heads, key_dim=embed_dim)
        self.dense = tf.keras.layers.Dense(embed_dim, activation="relu")


    def call(self, x, training):
        x = self.layer_norm_1(x)
        x = self.dense(x)

        attn_output = self.attention(
            query=x,
            value=x,
            key=x,
            attention_mask=None,
            training=training
        )

        x = self.layer_norm_2(x + attn_output)
        return x
```



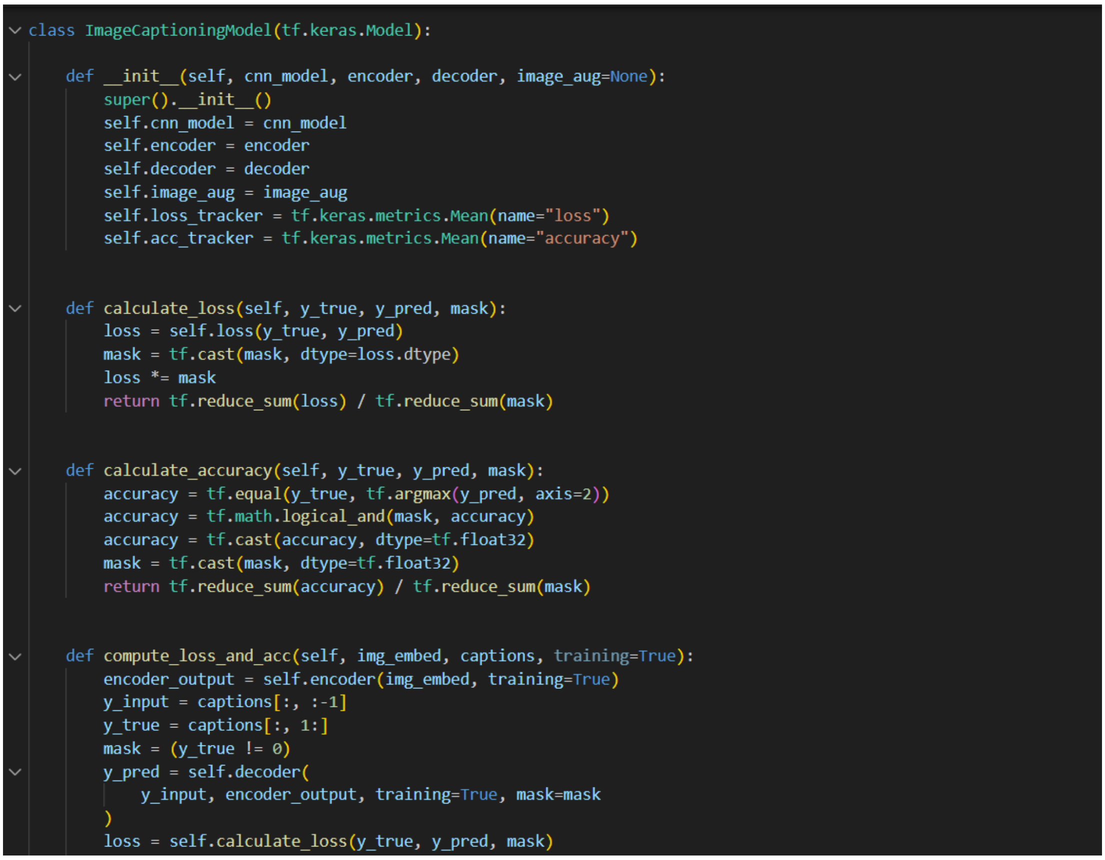
```python
class Embeddings(tf.keras.layers.Layer):

    def __init__(self, vocab_size, embed_dim, max_len):
        super().__init__()
        self.token_embeddings = tf.keras.layers.Embedding(
            vocab_size, embed_dim)
        self.position_embeddings = tf.keras.layers.Embedding(
            max_len, embed_dim, input_shape=(None, max_len))


    def call(self, input_ids):
        length = tf.shape(input_ids)[-1]
        position_ids = tf.range(start=0, limit=length, delta=1)
        position_ids = tf.expand_dims(position_ids, axis=0)

        token_embeddings = self.token_embeddings(input_ids)
        position_embeddings = self.position_embeddings(position_ids)

        return token_embeddings + position_embeddings


class TransformerDecoderLayer(tf.keras.layers.Layer):

    def __init__(self, embed_dim, units, num_heads):
        super().__init__()
        self.embedding = Embeddings(
            tokenizer.vocabulary_size(), embed_dim, MAX_LENGTH)

        self.attention_1 = tf.keras.layers.MultiHeadAttention(
            num_heads=num_heads, key_dim=embed_dim, dropout=0.1
        )
        self.attention_2 = tf.keras.layers.MultiHeadAttention(
            num_heads=num_heads, key_dim=embed_dim, dropout=0.1
        )

        self.layernorm_1 = tf.keras.layers.LayerNormalization()
```



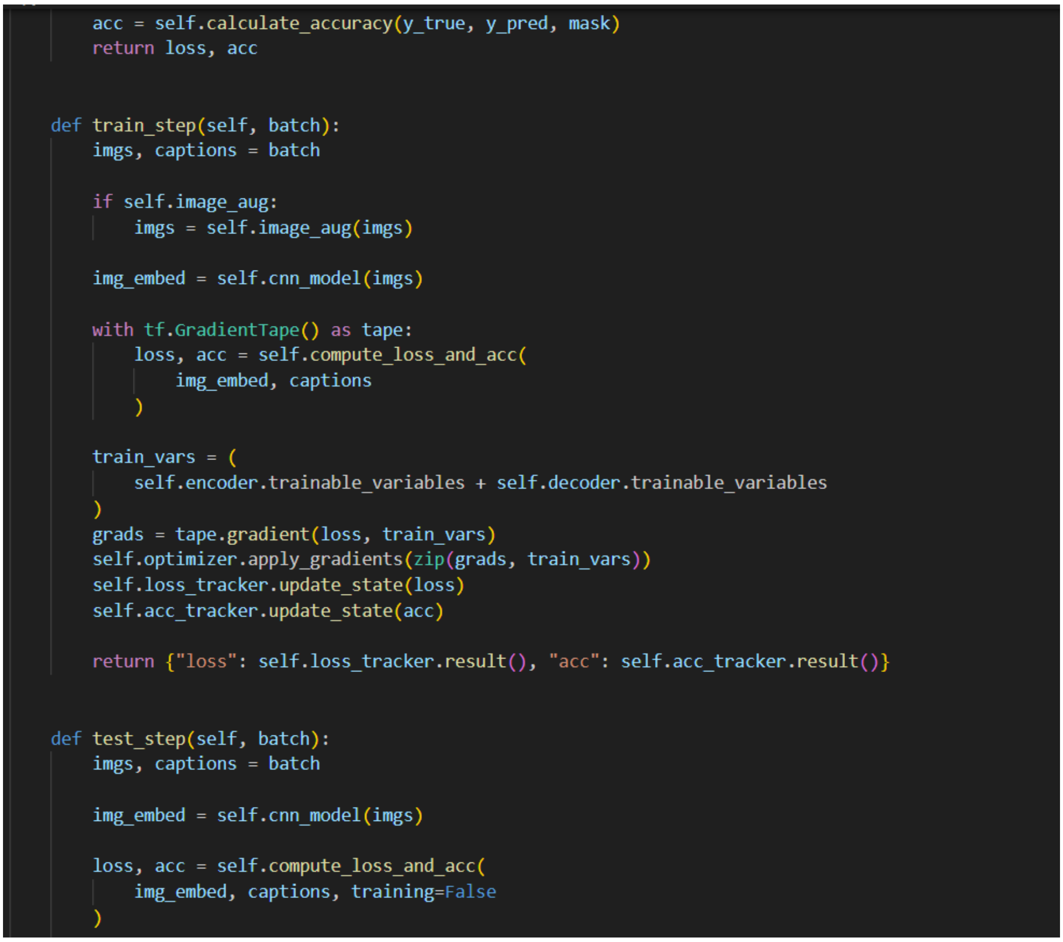
```python
        self.layernorm_2 = tf.keras.layers.LayerNormalization()
        self.layernorm_3 = tf.keras.layers.LayerNormalization()

        self.ffn_layer_1 = tf.keras.layers.Dense(units, activation="relu")
        self.ffn_layer_2 = tf.keras.layers.Dense(embed_dim)

        self.out = tf.keras.layers.Dense(tokenizer.vocabulary_size(), activation="softmax")

        self.dropout_1 = tf.keras.layers.Dropout(0.3)
        self.dropout_2 = tf.keras.layers.Dropout(0.5)


    def call(self, input_ids, encoder_output, training, mask=None):
        embeddings = self.embedding(input_ids)

        combined_mask = None
        padding_mask = None

        if mask is not None:
            causal_mask = self.get_causal_attention_mask(embeddings)
            padding_mask = tf.cast(mask[:, :, tf.newaxis], dtype=tf.int32)
            combined_mask = tf.cast(mask[:, tf.newaxis, :], dtype=tf.int32)
            combined_mask = tf.minimum(combined_mask, causal_mask)

        attn_output_1 = self.attention_1(
            query=embeddings,
            value=embeddings,
            key=embeddings,
            attention_mask=combined_mask,
            training=training
        )

        out_1 = self.layernorm_1(embeddings + attn_output_1)
```



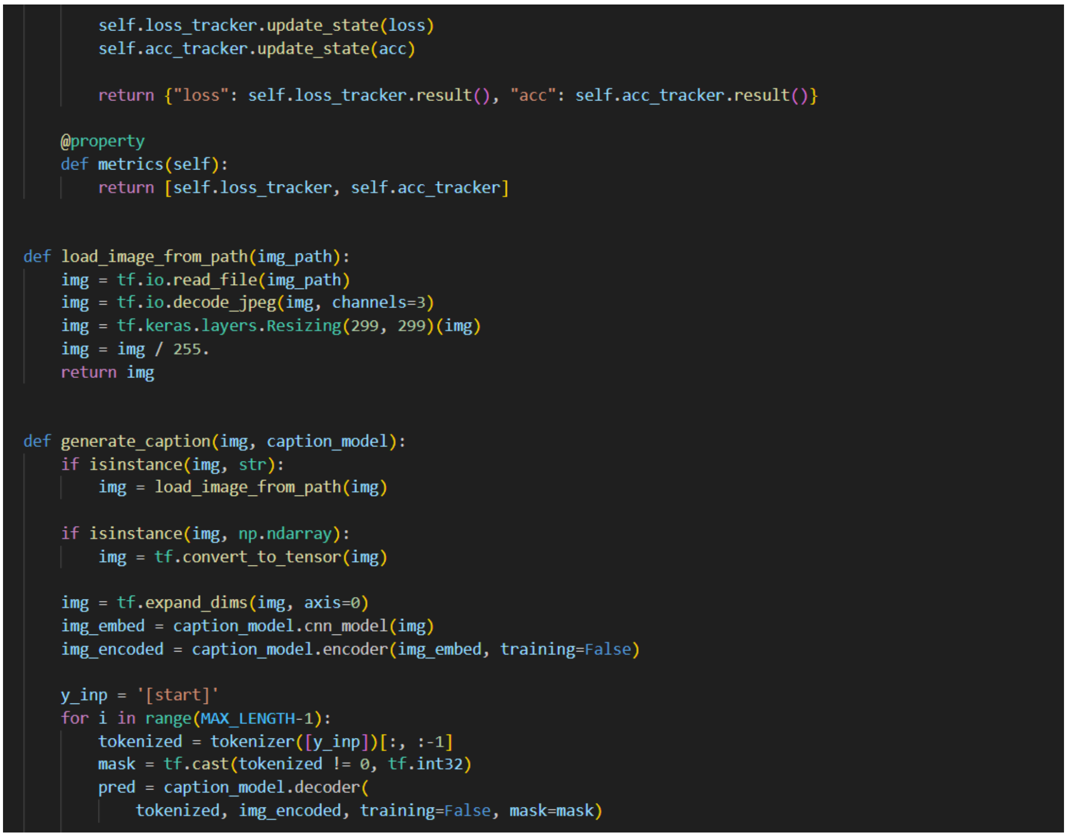
```python
        attn_output_2 = self.attention_2(
            query=out_1,
            value=encoder_output,
            key=encoder_output,
            attention_mask=padding_mask,
            training=training
        )

        out_2 = self.layernorm_2(out_1 + attn_output_2)

        ffn_out = self.ffn_layer_1(out_2)
        ffn_out = self.dropout_1(ffn_out, training=training)
        ffn_out = self.ffn_layer_2(ffn_out)

        ffn_out = self.layernorm_3(ffn_out + out_2)
        ffn_out = self.dropout_2(ffn_out, training=training)
        preds = self.out(ffn_out)
        return preds


    def get_causal_attention_mask(self, inputs):
        input_shape = tf.shape(inputs)
        batch_size, sequence_length = input_shape[0], input_shape[1]
        i = tf.range(sequence_length)[:, tf.newaxis]
        j = tf.range(sequence_length)
        mask = tf.cast(i >= j, dtype="int32")
        mask = tf.reshape(mask, (1, input_shape[1], input_shape[1]))
        mult = tf.concat(
            [tf.expand_dims(batch_size, -1), tf.constant([1, 1], dtype=tf.int32)],
            axis=0
        )
        return tf.tile(mask, mult)
```



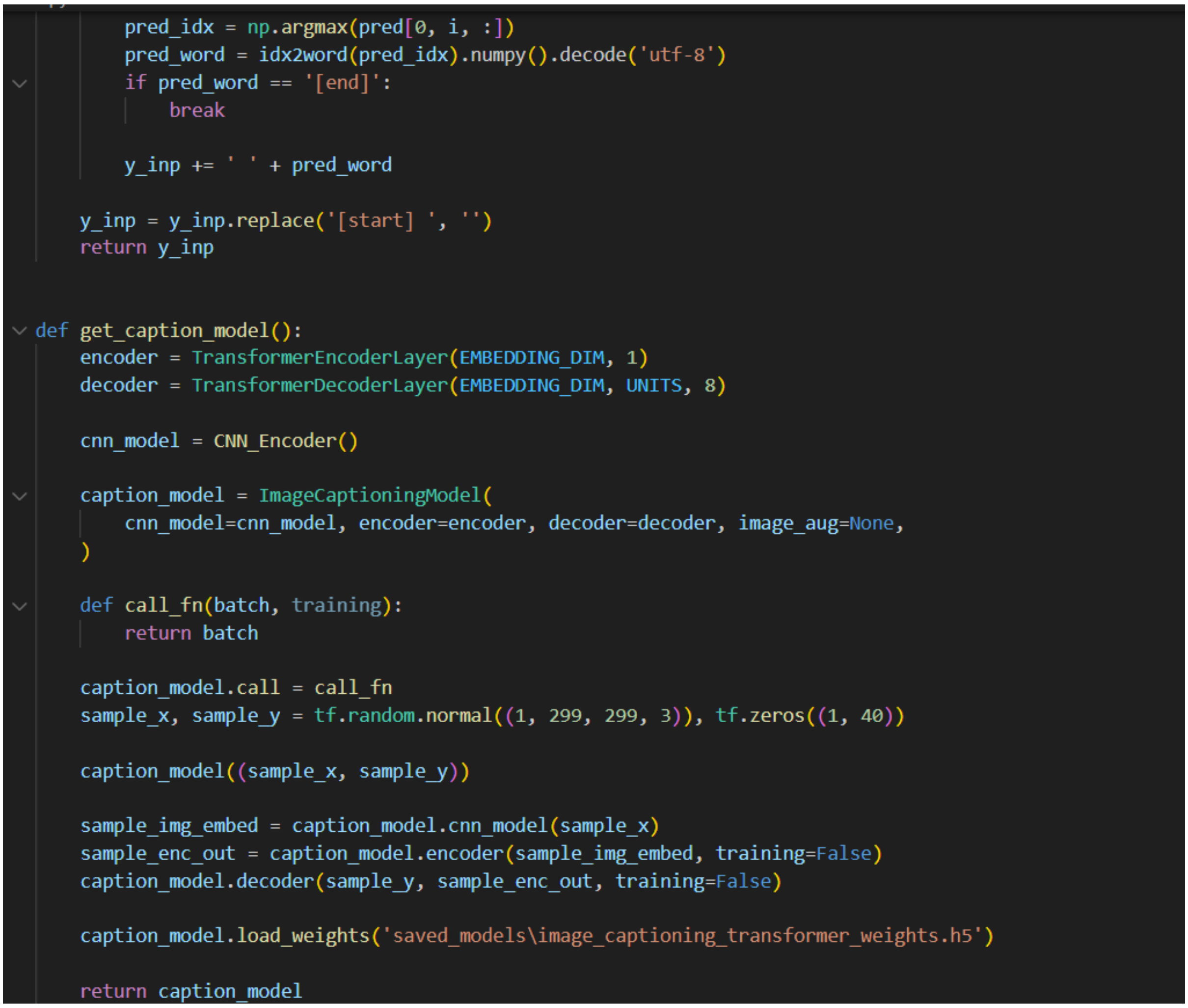
```python
class ImageCaptioningModel(tf.keras.Model):

    def __init__(self, cnn_model, encoder, decoder, image_aug=None):
        super().__init__()
        self.cnn_model = cnn_model
        self.encoder = encoder
        self.decoder = decoder
        self.image_aug = image_aug
        self.loss_tracker = tf.keras.metrics.Mean(name="loss")
        self.acc_tracker = tf.keras.metrics.Mean(name="accuracy")


    def calculate_loss(self, y_true, y_pred, mask):
        loss = self.loss(y_true, y_pred)
        mask = tf.cast(mask, dtype=loss.dtype)
        loss *= mask
        return tf.reduce_sum(loss) / tf.reduce_sum(mask)


    def calculate_accuracy(self, y_true, y_pred, mask):
        accuracy = tf.equal(y_true, tf.argmax(y_pred, axis=2))
        accuracy = tf.math.logical_and(mask, accuracy)
        accuracy = tf.cast(accuracy, dtype=tf.float32)
        mask = tf.cast(mask, dtype=tf.float32)
        return tf.reduce_sum(accuracy) / tf.reduce_sum(mask)


    def compute_loss_and_acc(self, img_embed, captions, training=True):
        encoder_output = self.encoder(img_embed, training=True)
        y_input = captions[:, :-1]
        y_true = captions[:, 1:]
        mask = (y_true != 0)
        y_pred = self.decoder(
            y_input, encoder_output, training=True, mask=mask
        )
        loss = self.calculate_loss(y_true, y_pred, mask)
```



```
        acc = self.calculate_accuracy(y_true, y_pred, mask)
        return loss, acc


    def train_step(self, batch):
        imgs, captions = batch

        if self.image_aug:
            imgs = self.image_aug(imgs)

        img_embed = self.cnn_model(imgs)

        with tf.GradientTape() as tape:
            loss, acc = self.compute_loss_and_acc(
                img_embed, captions
            )

        train_vars = (
            self.encoder.trainable_variables + self.decoder.trainable_variables
        )
        grads = tape.gradient(loss, train_vars)
        self.optimizer.apply_gradients(zip(grads, train_vars))
        self.loss_tracker.update_state(loss)
        self.acc_tracker.update_state(acc)

        return {"loss": self.loss_tracker.result(), "acc": self.acc_tracker.result()}


    def test_step(self, batch):
        imgs, captions = batch

        img_embed = self.cnn_model(imgs)

        loss, acc = self.compute_loss_and_acc(
            img_embed, captions, training=False
        )
```

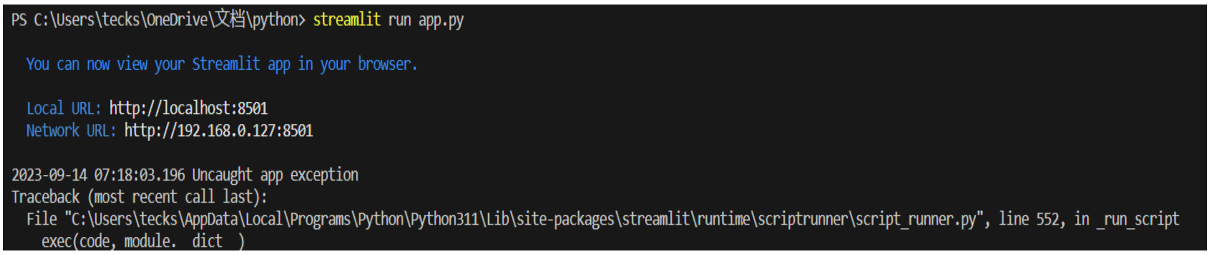


```
        self.loss_tracker.update_state(loss)
        self.acc_tracker.update_state(acc)

        return {"loss": self.loss_tracker.result(), "acc": self.acc_tracker.result()}

    @property
    def metrics(self):
        return [self.loss_tracker, self.acc_tracker]


def load_image_from_path(img_path):
    img = tf.io.read_file(img_path)
    img = tf.io.decode_jpeg(img, channels=3)
    img = tf.keras.layers.Resizing(299, 299)(img)
    img = img / 255.
    return img


def generate_caption(img, caption_model):
    if isinstance(img, str):
        img = load_image_from_path(img)

    if isinstance(img, np.ndarray):
        img = tf.convert_to_tensor(img)

    img = tf.expand_dims(img, axis=0)
    img_embed = caption_model.cnn_model(img)
    img_encoded = caption_model.encoder(img_embed, training=False)

    y_inp = '[start]'
    for i in range(MAX_LENGTH-1):
        tokenized = tokenizer([y_inp])[:, :-1]
        mask = tf.cast(tokenized != 0, tf.int32)
        pred = caption_model.decoder(
            tokenized, img_encoded, training=False, mask=mask)
```



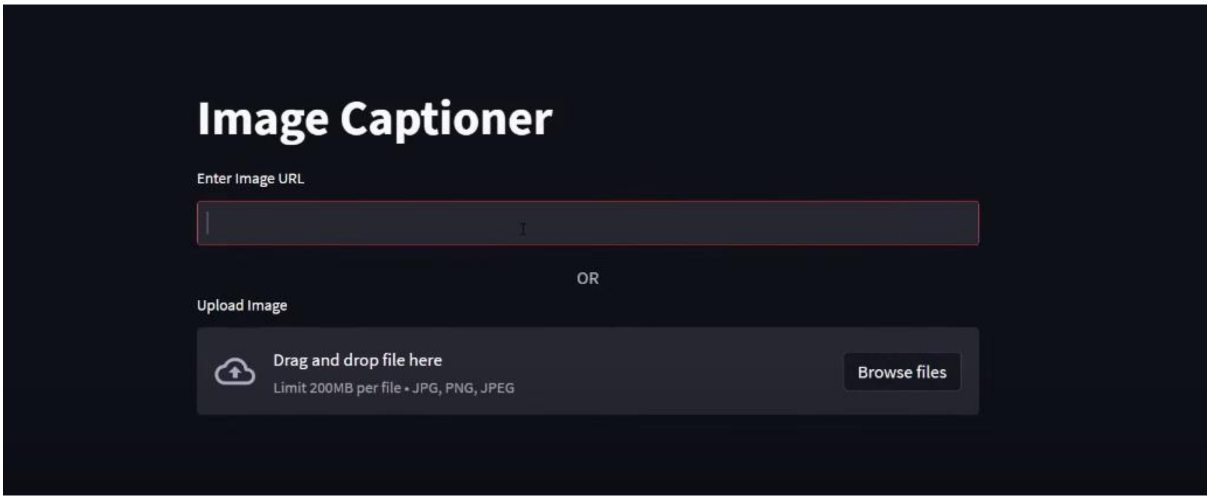
```
        pred_idx = np.argmax(pred[0, i, :])
        pred_word = idx2word(pred_idx).numpy().decode('utf-8')
        if pred_word == '[end]':
            break

        y_inp += ' ' + pred_word

    y_inp = y_inp.replace('[start] ', '')
    return y_inp


def get_caption_model():
    encoder = TransformerEncoderLayer(EMBEDDING_DIM, 1)
    decoder = TransformerDecoderLayer(EMBEDDING_DIM, UNITS, 8)

    cnn_model = CNN_Encoder()

    caption_model = ImageCaptioningModel(
        cnn_model=cnn_model, encoder=encoder, decoder=decoder, image_aug=None,
    )

    def call_fn(batch, training):
        return batch

    caption_model.call = call_fn
    sample_x, sample_y = tf.random.normal((1, 299, 299, 3)), tf.zeros((1, 40))

    caption_model((sample_x, sample_y))

    sample_img_embed = caption_model.cnn_model(sample_x)
    sample_enc_out = caption_model.encoder(sample_img_embed, training=False)
    caption_model.decoder(sample_y, sample_enc_out, training=False)

    caption_model.load_weights('saved_models\image_captioning_transformer_weights.h5')

    return caption_model
```

6.2 Testing Setup and Result

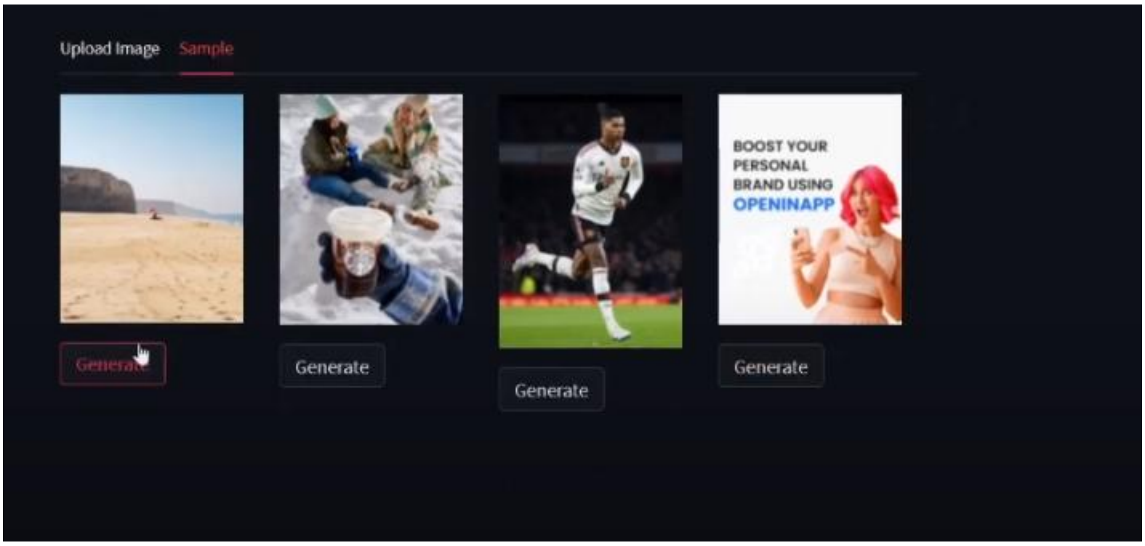


```
PS C:\Users\tecks\OneDrive\文档\python> streamlit run app.py

  You can now view your Streamlit app in your browser.

  Local URL: http://localhost:8501
  Network URL: http://192.168.0.127:8501

2023-09-14 07:18:03.196 Uncaught app exception
Traceback (most recent call last):
  File "C:\Users\tecks\AppData\Local\Programs\Python\Python311\Lib\site-packages\streamlit\runtime\scriptrunner\script_runner.py", line 552, in _run_script
    exec(code, module.__dict__)
```

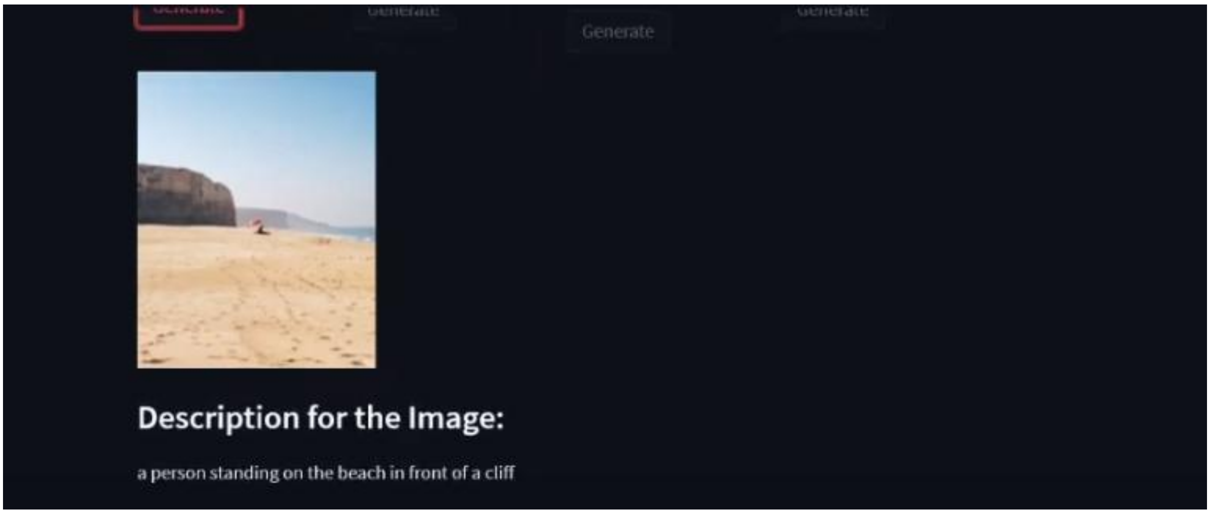
Connect visual studio python file by connect streamlit to run the file and here is the local URL and Network URL. After connecting, will direct bring you to browser see webpage.



This is main page with image captioner title, user can choose to enter image URL or choose to upload the image by click the browse files with the limit file size of 200MB per images which can be JPG, PNG, and JPEG format.



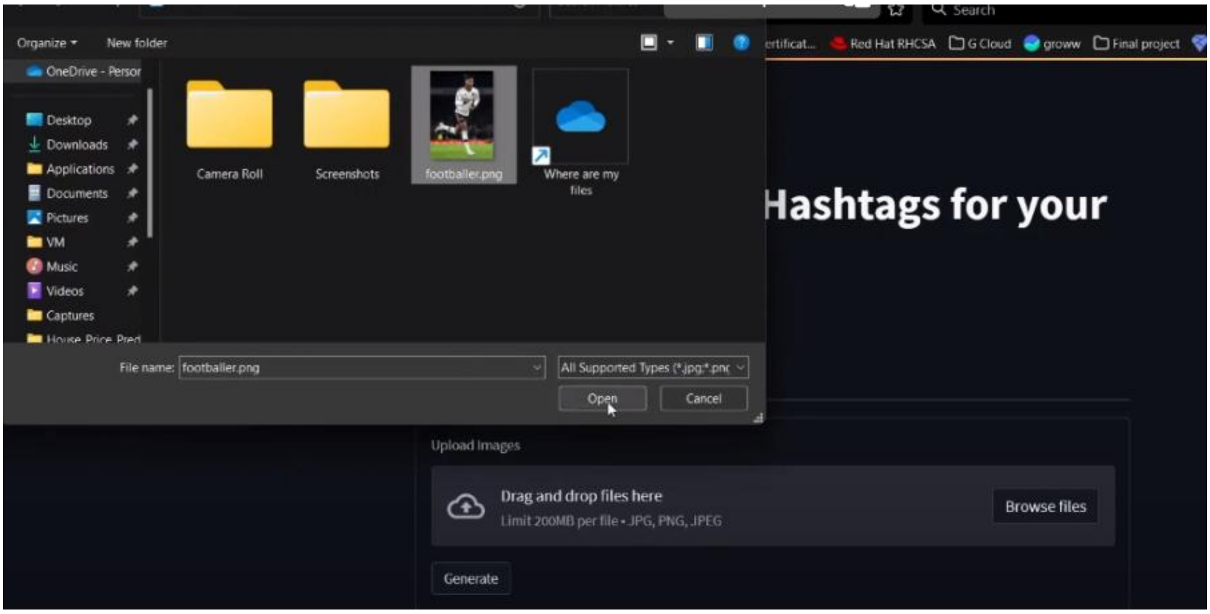
Here we can see the sample, we try to click generate button at first images see what it will send captions for us.



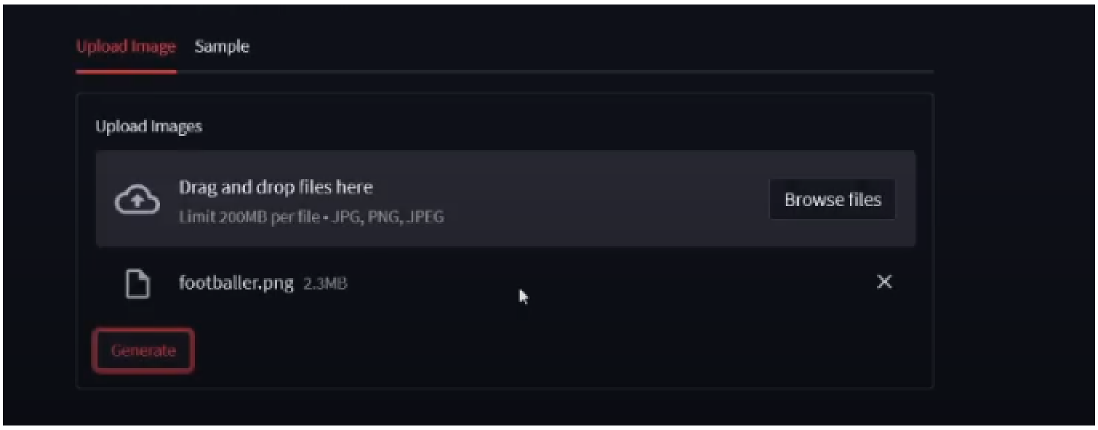
Description for the Image:

a person standing on the beach in front of a cliff

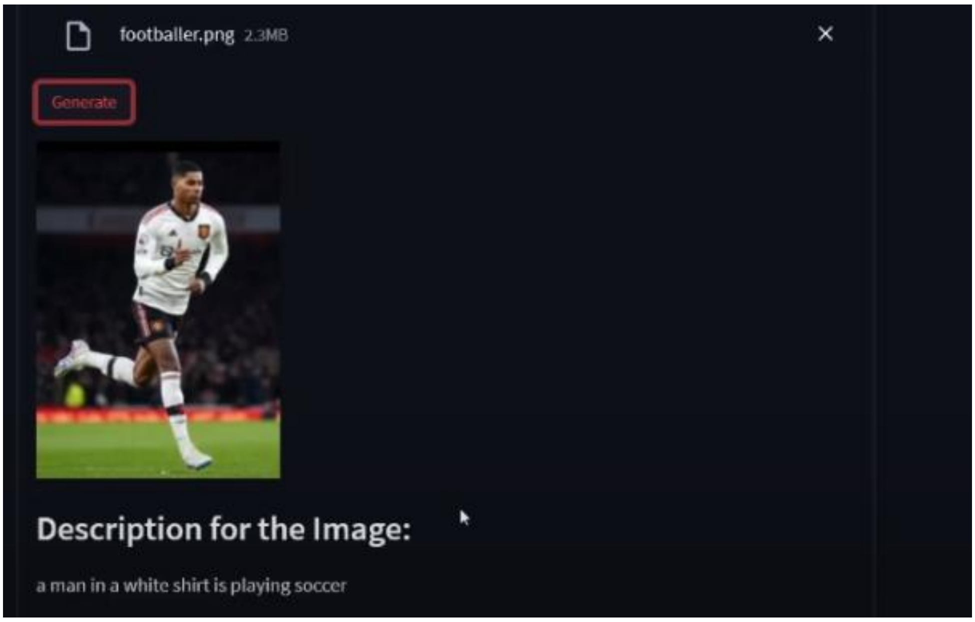
This is description of the image, which is captions detect by system, which shows as a person standing on the beach in front of a cliff.



This is the step when we click browse files button it will jump to our folder place which will let us to select what images files we would like to upload.



After successfully upload the image, it will show like this.



Description for the Image:

a man in a white shirt is playing soccer

This is images which suitable for the caption generate.

6.3 Project Challenges

The project challenges will be to maintain the large datasets to ensure the image classification or image captioning for one people is quite challenging. The motivation maintaining motivation and staying focused on a personal project, especially when it's self-initiated, can be difficult. Personal projects are often solitary endeavors. Some individuals may find it challenging to work alone for extended periods, leading to feelings of isolation. Self-learning on deep learning and data mining is difficult to understand as expert because it needs long time to develop and investigate around it.

6.4 Objectives Evaluation

1. The first research objective is to classify the images deep learning through python.
(Complete)

2.The second research objective is to develop a website to increase effectiveness in scene visualization and sentence-based algorithm identification.
(Complete)

3.The third objective is to build a recommendation system that matches the opinion of users with correct result rather than incorrect unrelated results.
(Complete)

6.5 Concluding Remark

Although the deep learning and data mining categories is difficult, I have tried my best to solve the challenges as I can. The system can help to build system help user to upload image when they don’t know what can get the meaning from images. The system will help user to solve the issues and problem.

CHAPTER 7 CONCLUSION AND RECOMMENDATION

7.1 Conclusion

Machine learning includes deep learning and data mining. This endeavor necessitates extensive research in the fields of deep learning and data mining. The study subject to be addressed is how to use language python and anaconda navigator to solve deep learning for scene visualization and sentence-based picture synthesis via image classification and image captioning. Image classification is a component of a project with numerous practical applications in a variety of domains, including object detection, medical imaging, content moderation, and quality control. Image captioning generators are simple programs that take an image and try to generate a caption that matches the gist of that image as nearly as possible, including the entire meaning of one image in just one sentence, saving time. The attention mechanism came to the rescue when the picture captioning between NLP and computer vision and work in coordination to make image captioning possible. The project's methodology and techniques are research-based, and it is currently in the research process. The language python and anaconda navigator would be used to launch the jupyter notebook and Google colab. In addition, the Flickr8k Dataset was obtained from Kaggle to launch the progress.

The project aims to describe how densely a user must capture a given scene for reliable rendering performance. The aim of the thesis is to propose new efficient algorithms for scene visualization and sentence image synthesis. In this thesis, find that it is useful to categorize IBR algorithms by the extent to which they use explicit scene geometry. Hence, these provides motivation for the implementation of this project. Besides, the visual content in nowadays which technology era become important make user can more easily to get the visual content that relevant to requirements can increase user experience by just simple insert the input. Most importantly can improve the effectiveness of the user with just provide simple input and system can generate output for user that was connected to the datasets. It can increase effectiveness in image classification and captioning process, which classification the image in group and provide output for user, user which upload the image will provide caption for user to know the many meaning of feature image in just one sentence. Python languages was using in this project which to classify the images deep learning through python and group them into respective categories to easy user on the image classify. Through the website, user can develop a website to increase effectiveness in scene visualization and sentence-based algorithm identification to run the program which upload images get meaning from images when user does not understand on it to matches the opinion of users with correct result rather than incorrect unrelated results.

The problem encountered were correctness to provide the accuracy for the output of sentences with images that can automatically detect the images with the objects and complexness information from the image will also affect the correctness through the output. The image feature will relate to the accuracy of the output and input provided. In case to implement the python code connect with streamlit take long time to configure on the system connect to website.

7.2 Recommendations

There are some enhancements can be made in this project. Firstly, fine-grained classification that investigate techniques for fine-grained image classification, where the goal is to distinguish between highly similar categories within a broader class. Include with multimodal fusion that combine image classification with other modalities such as text or audio to enable more comprehensive understanding and interaction in multimodal applications. Add privacy preserving classifications to investigate privacy-preserving methods for image classification, particularly in security applications to protect sensitive information.

Furthermore, the website can support notification feature in future, which website lacking. Notifications can let users to be notified any new updates, which include announcement posted. In this case, user won’t scared for worrying about miss important issues. I can make firebase Cloud messaging to send notification through email and together with web push notification of user allow or decline notification from website.

The system can be improved provide with more recommendations with sample provided which can be provided the classification and support different languages to provide the output in different languages, which let user have choices to choose language even though they don’t understand English. In this case, the target users will be enlarged from different countries.

REFERENCES

[1] P. Srinivasan, "Scene representations for view synthesis with deep learning," Berkeley.edu. [Online]. Available: https://digitalassets.lib.berkeley.edu/techreports/ucb/incoming/EECS-2020-214.pdf. [Accessed: 14-Sep-2023].

[2] "DeepAI," *DeepAI*. [Online]. Available: https://deepai.org/. [Accessed: 14-Sep-2023].

[3] Ucdavis.edu. [Online]. Available: https://mindlab.physics.ucdavis.edu/. [Accessed: 14-Sep-2023].

[4] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," in Lecture Notes in Computer Science, Cham: Springer International Publishing, 2015, pp. 234–241.

[5] S. Ghaffarian, J. Valente, M. van der Voort, and B. Tekinerdogan, "Effect of attention mechanism in deep learning-based remote sensing image processing: A systematic literature review," *Remote Sens. (Basel)*, vol. 13, no. 15, p. 2965, 2021.

[6] W. Li *et al.*, "Object-driven text-to-image synthesis via adversarial training," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 12174–12182.

[7] Z. Niu, G. Zhong, and H. Yu, "A review on the attention mechanism of deep learning," *Neurocomputing*, vol. 452, pp. 48–62, 2021.

[8] K. Xu et al., "Show, attend and tell: Neural image caption generation with visual attention," in Proceedings of the 32nd International Conference on Machine Learning, 07--09 Jul 2015, vol. 37, pp. 2048–2057.

[9] S. Degadwala, D. Vyas, H. Biswas, U. Chakraborty, and S. Saha, "Image captioning using inception V3 transfer learning model," in *2021 6th International Conference on Communication and Electronics Systems (ICCES)*, 2021, pp. 1103–1108.

[10] R. Kang, A. Sunil, and M. Chen, "Mobile app for text-to-image synthesis," in *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, Cham: Springer International Publishing, 2019, pp. 32–43.

[11] D. Zeng *et al.*, "Deep learning for scene classification: A survey," *arXiv [cs.CV]*, 2021.

[12] S. Chavda and M. Goyani, "Scene level image classification: A literature review," Neural Process. Lett., vol. 55, no. 3, pp. 2471–2520, 2023.

[13] S. Tyagi and D. Yadav, "A comprehensive review on image synthesis with adversarial networks: Theory, literature, and applications," Arch. Comput. Methods Eng., vol. 29, no. 5, pp. 2685–2705, 2022.

[14] Q. Long, M. Wang, and L. Li, "Generative Imagination Elevates Machine Translation," *arXiv [cs.CL]*, 2020.

[15] O. Boiman, E. Shechtman, and M. Irani, "In defense of Nearest-Neighbor based image classification," in *2008 IEEE Conference on Computer Vision and Pattern Recognition*, 2008, pp. 1–8.

[16] F. Wang *et al.*, "Residual Attention Network for Image Classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 3156–3164.

[17] *Neurips.cc.* [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2021/hash/07e87c2f4fc7f7c96116d8e2a92790f5-Abstract.html. [Accessed: 14-Sep-2023].

[18] K. Tang, M. Paluri, L. Fei-Fei, R. Fergus, and L. Bourdev, “Improving image classification with location context,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1008–1016.

[19] *Neurips.cc*. [Online]. Available: https://proceedings.neurips.cc/paper/2020/hash/d8330f857a17c53d217014ee776bfd50-Abstract.html. [Accessed: 14-Sep-2023].

[20] Y. Sun, B. Xue, M. Zhang, and G. G. Yen, “Evolving deep convolutional neural networks for image classification,” *IEEE Trans. Evol. Comput.*, vol. 24, no. 2, pp. 394–407, 2020.

[21] R. Castro, I. Pineda, W. Lim, and M. E. Morocho-Cayamcela, “Deep learning approaches based on transformer architectures for image captioning tasks,” *IEEE Access*, vol. 10, pp. 33679–33694, 2022.

[22] B. Makav and V. Kilic, “A new image captioning approach for visually impaired people,” in *2019 11th International Conference on Electrical and Electronics Engineering (ELECO)*, 2019, pp. 945–949.

[23] O. Sargar and S. Kinger, “Image captioning methods and metrics,” in 2021 International Conference on Emerging Smart Computing and Informatics (ESCI), 2021, pp. 522–526.

[24] Y. Chu, X. Yue, L. Yu, M. Sergei, and Z. Wang, “Automatic image captioning based on ResNet50 and LSTM with soft attention,” *Wirel. Commun. Mob. Comput.*, vol. 2020, pp. 1–7, 2020.

[25] Y. Li, T. Yao, Y. Pan, H. Chao, and T. Mei, “Pointing novel objects in image captioning,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 12497–12506.

[26] M. Z. Hossain, F. Sohel, M. F. Shiratuddin, H. Laga, and M. Bennamoun, “Text to image synthesis for improved image captioning,” *IEEE Access*, vol. 9, pp. 64918–64928, 2021.

[27] B. Wan, W. Jiang, Y.-M. Fang, M. Zhu, Q. Li, and Y. Liu, “Revisiting image captioning via maximum discrepancy competition,” *Pattern Recognit.*, vol. 122, no. 108358, p. 108358, 2022.

[28] M. Poongodi, M. Hamdi, and H. Wang, “Image and audio caps: automated captioning of background sounds and images using deep learning,” Multimed. Syst., 2022.

[29] A. K. Sharma, S. Chaurasia, and D. K. Srivastava, “Sentimental short sentences classification by using CNN deep learning model with fine tuned Word2Vec,” *Procedia Comput. Sci.*, vol. 167, pp. 1139–1147, 2020.

[30] J. Guo *et al.*, “GluonCV and GluonNLP: Deep learning in computer vision and natural language processing,” *Jmlr.org*. [Online]. Available: https://www.jmlr.org/papers/volume21/19-429/19-429.pdf?ref=https://githubhelp.com. [Accessed: 14-Sep-2023].

APPENDIX

Questionnaire

1. Do you know what is deep learning for scene visualization and sentence-based image synthesis?

A. Yes

B. No

C. Heard of it, but not really understand it.

2. Do you know what is U-Net convolution neural network?

A. Yes

B. No

C. Heard of it, but not really understand it.

3. Do you experiences in using any deep learning application (e.g visual recognition)?

A. Yes

B. No

C. Maybe

4. Do you think you will use and benefit from visualizing deep learning?

A. Yes

B. No

5. Do you feel difficulties in understanding and visualizing the 2D and 3D photo or images of knee and ankle foot structure on the paper materials.

A. Yes

B. No

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	**Study week no.: 2**
Student Name & ID: Beh Teck Sian 19ACB01560	
Supervisor: Dr Ramesh Kumar Ayyasamy	
Project Title: DEEP LEARNING FOR SCENE VISUALIZATION AND SENTENCE-BASED IMAGE SYNTHESIS	

1. WORK DONE

Done for chapter and literature review, add three more literature reviews.

2. WORK TO BE DONE

Done for chapter and literature review, add three more literature reviews.

3. PROBLEMS ENCOUNTERED

Difficulties on understand different term on techniques many choices.

4. SELF EVALUATION OF THE PROGRESS

Finish the work to be done on time and need to progress more faster to complete in time.

ramesh

Supervisor's signature

BEH

Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	**Study week no.: 4**
Student Name & ID: Beh Teck Sian 19ACB01560	
Supervisor: Dr Ramesh Kumar Ayyasamy	
Project Title: DEEP LEARNING FOR SCENE VISUALIZATION AND SENTENCE-BASED IMAGE SYNTHESIS	

1. WORK DONE

Start for system models, class diagram, system diagram.

2. WORK TO BE DONE

Done for system models, class diagram, and system diagram.

3. PROBLEMS ENCOUNTERED

Need produce the system models based on expectations.

4. SELF EVALUATION OF THE PROGRESS

Finish the work to be done on time.

rameh

Supervisor's signature

BEH

Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	**Study week no.: 6**
Student Name & ID: Beh Teck Sian 19ACB01560	
Supervisor: Dr Ramesh Kumar Ayyasamy	
Project Title: DEEP LEARNING FOR SCENE VISUALIZATION AND SENTENCE-BASED IMAGE SYNTHESIS	

1. WORK DONE Using python for image classification and image captioning deep learning.
2. WORK TO BE DONE Done for the progress.
3. PROBLEMS ENCOUNTERED Difficulties on apply python to run the large datasets
4. SELF EVALUATION OF THE PROGRESS Almost finish the progress.

rameesh

Supervisor's signature

BEH

Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	**Study week no.: 8**
Student Name & ID: Beh Teck Sian 19ACB01560	
Supervisor: Dr Ramesh Kumar Ayyasamy	
Project Title: DEEP LEARNING FOR SCENE VISUALIZATION AND SENTENCE-BASED IMAGE SYNTHESIS	

1. WORK DONE

Done for the coding part and apply connect to the visual studio code and deploy through streamlit.

2. WORK TO BE DONE

Done for the coding part and apply connect to the visual studio code and deploy through streamlit.

3. PROBLEMS ENCOUNTERED

Difficulties on connected datasets and from different platform connect with visual studio code to deploy the website.

4. SELF EVALUATION OF THE PROGRESS

Almost finish left connect to website.

ramesh

Supervisor's signature

BEH

Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	**Study week no.: 10**
Student Name & ID: Beh Teck Sian 19ACB01560	
Supervisor: Dr Ramesh Kumar Ayyasamy	
Project Title: DEEP LEARNING FOR SCENE VISUALIZATION AND SENTENCE-BASED IMAGE SYNTHESIS	

1. WORK DONE

Try the website and all the functionalities work properly or not.

2. WORK TO BE DONE

Done for the testing part and continue maintenance.

3. PROBLEMS ENCOUNTERED

Difficulties on the website maintenance because of datasets connected unsuccessful.

4. SELF EVALUATION OF THE PROGRESS

Solve the issue connect to browser.

ramesh
Supervisor's signature

BEH
Student's signature

FINAL YEAR PROJECT WEEKLY REPORT
(Project II)

Trimester, Year: Y3S3	**Study week no.: 12**
Student Name & ID: Beh Teck Sian 19ACB01560	
Supervisor: Dr Ramesh Kumar Ayyasamy	
Project Title: DEEP LEARNING FOR SCENE VISUALIZATION AND SENTENCE-BASED IMAGE SYNTHESIS	

1. WORK DONE

Start to complete the report by involve all the screenshot needed and system output.

2. WORK TO BE DONE

Complete the report.

3. PROBLEMS ENCOUNTERED

4. SELF EVALUATION OF THE PROGRESS

Complete the report.

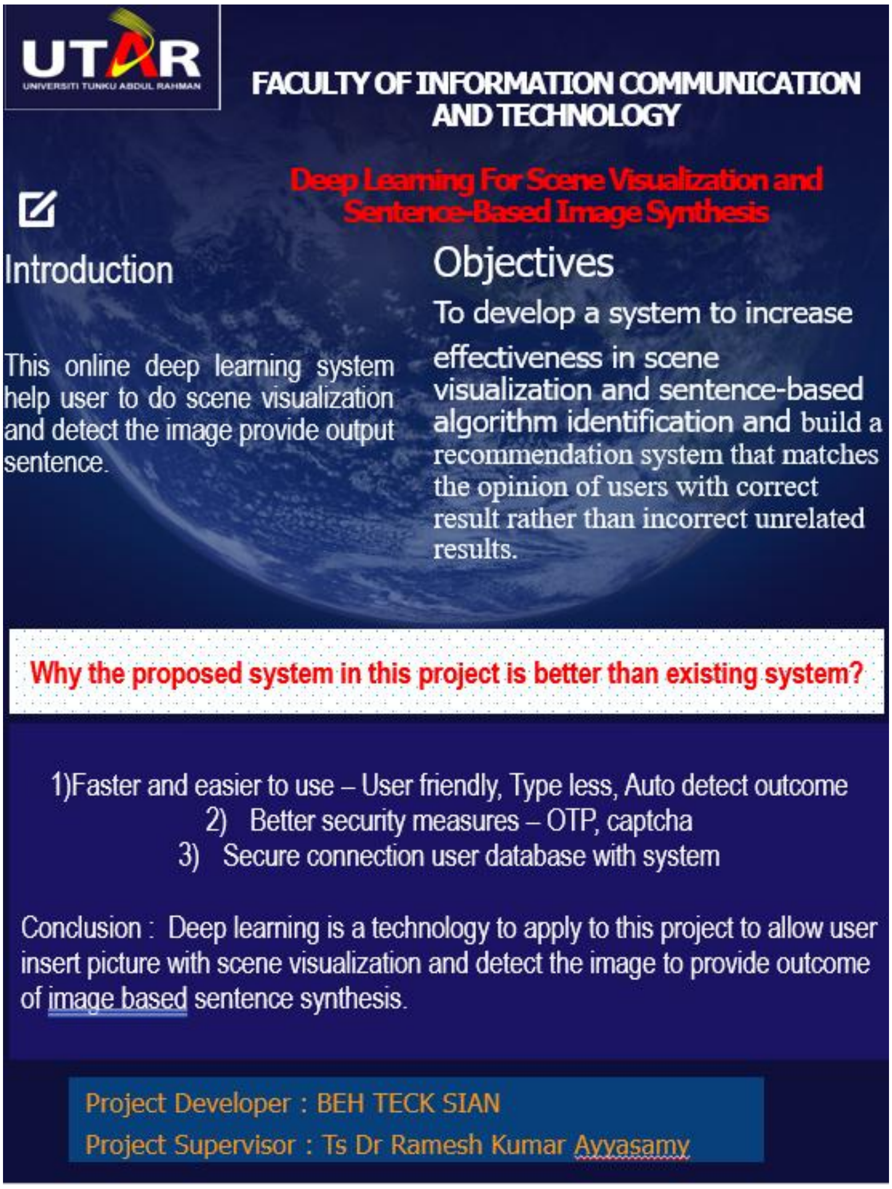
ramesh

Supervisor's signature

BEH

Student's signature

POSTER



UTAR
UNIVERSITI TUNKU ABDUL RAHMAN

FACULTY OF INFORMATION COMMUNICATION AND TECHNOLOGY

Deep Learning For Scene Visualization and Sentence-Based Image Synthesis

Introduction

This online deep learning system help user to do scene visualization and detect the image provide output sentence.

Objectives

To develop a system to increase effectiveness in scene visualization and sentence-based algorithm identification and build a recommendation system that matches the opinion of users with correct result rather than incorrect unrelated results.

Why the proposed system in this project is better than existing system?

1)Faster and easier to use – User friendly, Type less, Auto detect outcome

2) Better security measures – OTP, captcha

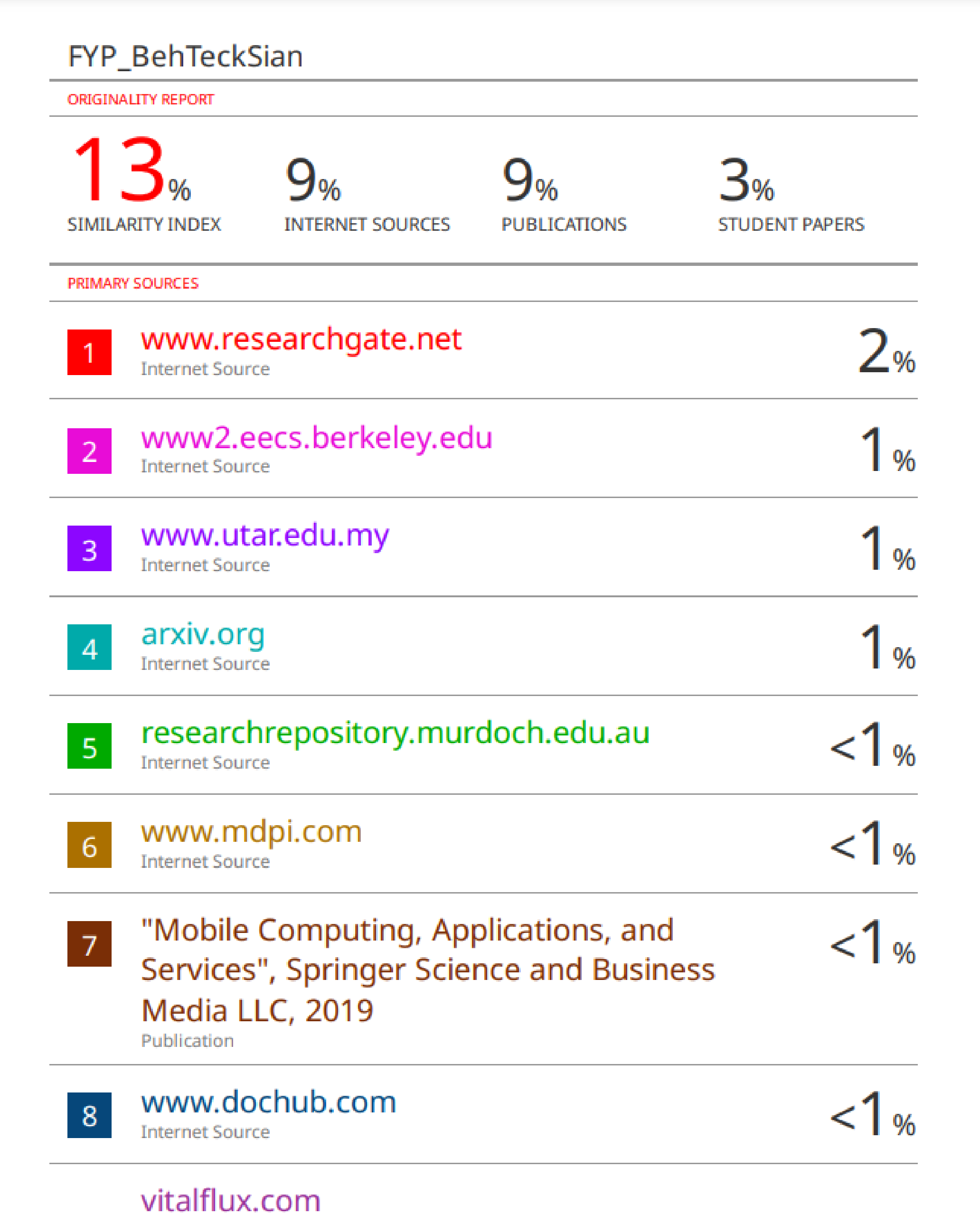
3) Secure connection user database with system

Conclusion : Deep learning is a technology to apply to this project to allow user insert picture with scene visualization and detect the image to provide outcome of image based sentence synthesis.

Project Developer : BEH TECK SIAN

Project Supervisor : Ts Dr Ramesh Kumar Ayyasamy

PLAGIARISM CHECK RESULT



FYP_BehTeckSian

ORIGINALITY REPORT

13%	9%	9%	3%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	www.researchgate.net Internet Source	2%
2	www2.eecs.berkeley.edu Internet Source	1%
3	www.utar.edu.my Internet Source	1%
4	arxiv.org Internet Source	1%
5	researchrepository.murdoch.edu.au Internet Source	<1%
6	www.mdpi.com Internet Source	<1%
7	"Mobile Computing, Applications, and Services", Springer Science and Business Media LLC, 2019 Publication	<1%
8	www.dochub.com Internet Source	<1%
	vitalflux.com	

FYP2 CHECKLIST

Universiti Tunku Abdul Rahman			
Form Title: Supervisor's Comments on Originality Report Generated by Turnitin for Submission of Final Year Project Report (for Undergraduate Programmes)			
Form Number: FM-IAD-005	Rev No.: 0	Effective Date: 01/10/2013	Page No.: 1of 1

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

Full Name(s) of Candidate(s)	BEH TECK SIAN
ID Number(s)	19ACB01560
Programme / Course	BACHELOR OF INFORMATION SYSTEMS (HONOURS) INFORMATION SYSTEM ENGINEERING
Title of Final Year Project	DEEP LEARNING FOR SCENE VISUALIZATION AND SENTENCE-BASED IMAGE SYNTHESIS

Similarity	**Supervisor's Comments (Compulsory if parameters of originality exceeds the limits approved by UTAR)**
Overall similarity index: 13 **%** **Similarity by source** Internet Sources: 9 % Publications: 9 % Student Papers: 3 %	
Number of individual sources listed of more than 3% similarity: 0	
Parameters of originality required and limits approved by UTAR are as Follows: **(i) Overall similarity index is 20% and below, and** **(ii) Matching of individual sources listed must be less than 3% each, and** **(iii) Matching texts in continuous block must not exceed 8 words** ***Note: Parameters (i) – (ii) shall exclude quotes, bibliography and text matches which are less than 8 words.***	

Note Supervisor/Candidate(s) is/are required to provide softcopy of full set of the originality report to Faculty/Institute

Based on the above results, I hereby declare that I am satisfied with the originality of the Final Year Project Report submitted by my student(s) as named above.

ramesh
Signature of Supervisor

Name: Dr.Ramesh Kumar Ayyasamy

Date: 15/09/2023

Signature of Co-Supervisor

Name: ______________________

Date: ______________________

UNIVERSITI TUNKU ABDUL RAHMAN

FACULTY OF INFORMATION & COMMUNICATION TECHNOLOGY (KAMPAR CAMPUS)

CHECKLIST FOR FYP2 THESIS SUBMISSION

Student Id	19ACB01560
Student Name	BEH TECK SIAN
Supervisor Name	Dr. Ramesh Kumar Ayyasamy

TICK (√)	**DOCUMENT ITEMS** Your report must include all the items below. Put a tick on the left column after you have checked your report with respect to the corresponding item.
√	Title Page
√	Signed Report Status Declaration Form
√	Signed FYP Thesis Submission Form
√	Signed form of the Declaration of Originality
√	Acknowledgement
√	Abstract
√	Table of Contents
√	List of Figures (if applicable)
√	List of Tables (if applicable)
√	List of Symbols (if applicable)
√	List of Abbreviations (if applicable)
√	Chapters / Content
√	Bibliography (or References)
√	All references in bibliography are cited in the thesis, especially in the chapter of literature review
√	Appendices (if applicable)
√	Weekly Log
√	Poster
√	Signed Turnitin Report (Plagiarism Check Result - Form Number: FM-IAD-005)
√	I agree 5 marks will be deducted due to incorrect format, declare wrongly the ticked of these items, and/or any dispute happening for these items in this report.

*Include this form (checklist) in the thesis (Bind together as the last page)

I, the author, have checked and confirmed all the items listed in the table are included in my report.

BEH
____
(Signature of Student)
Date: 12/9/2023