PROJECT NAME -IMAGE CAPTION GENERATION

TEAM ID -591732
TEAM MEMBERS
KELAVATH BALAJI NAIK
CHEEKATIPALLI NANDHINI
IRUGU SINDHU

Contents

1.Introduction

The ability to understand and describe visual content has long been a fundamental challenge in the field of computer vision. Image captioning, the task of generating a textual description that accurately captures the content of an image, has gained significant attention in recent years due to its potential applications in areas such as image understanding, content retrieval, and accessibility for visually impaired individuals.

The purpose of this project is to develop an image caption generator using Long Short-Term Memory (LSTM), a type of recurrent neural network (RNN) known for its ability to model sequential data. The model takes an input image and generates a descriptive caption that conveys the visual information present in the image. By combining the power of deep learning with natural language processing, we aim to bridge the gap between images and textual descriptions, enabling machines to comprehend and communicate the content of visual data.

The motivation behind this project lies in the growing demand for automated image understanding and the need to enhance human-computer interaction through natural language interfaces. With the proliferation of image-centric platforms and the vast amount of visual content available, an accurate and efficient image captioning system can provide valuable insights, facilitate information retrieval, and improve accessibility to visual content for diverse user groups.

Throughout this project, we will leverage the capabilities of the VGG16 model for image feature extraction and employ LSTM networks for sequence modeling and caption generation. The model will be trained on a dataset of images paired with corresponding textual descriptions. We will evaluate the performance of the model using metrics such as BLEU (Bilingual Evaluation Understudy) scores, which assess the similarity between generated captions and human-written references.

By developing an effective image caption generator, we aim to contribute to the advancement of computer vision and natural language processing, opening up new possibilities for understanding and interacting with visual content. This project report presents a comprehensive overview of the methodology, dataset, model architecture, training, evaluation, and deployment process, along with the results and discussions.

2. Problem Statement

The problem we aim to address in this project is the generation of accurate and meaningful captions for images. While humans can effortlessly describe the content of an image, teaching machines to do the same is a complex task. The challenges associated with image captioning include understanding the visual context, capturing relevant details, maintaining coherence and relevance in the generated text, and dealing with the inherent ambiguity and subjectivity of language.

The goal is to develop a model that can accurately perceive the visual information in an image and generate captions that not only describe the objects and scenes but also convey a deeper understanding of the content. The model should be able to capture relationships between objects, recognize actions or events, and incorporate contextual knowledge to produce coherent and contextually relevant descriptions.

3. Project Overview

This project follows a comprehensive workflow to build an image caption generator. It involves several key components:

Dataset: We utilize the Flickr8K dataset, which consists of a large collection of images along with descriptive captions. The dataset is preprocessed to clean the descriptions, remove punctuation, convert text to lowercase, and filter out irrelevant words.

Feature Extraction: We employ the VGG16 model, a deep convolutional neural network (CNN), to extract meaningful features from the input images. The pre-trained VGG16 model is used as a feature extractor by removing the last classification layer, enabling us to obtain a fixed-length vector representation for each image.

Caption Preprocessing: The textual captions are processed to create a vocabulary and tokenize the words. We employ techniques such as word-to-index mapping and one-hot encoding to represent the captions in a format suitable for training the LSTM-based caption generator.

Model Architecture: The core of the image caption generator is a combination of LSTM and dense layers. The LSTM network serves as a sequence model, generating captions word by word, while the dense layers help in mapping the extracted features from images to the generated captions.

Training and Evaluation: The model is trained using the prepared dataset, and the training process involves optimizing the model parameters using the Adam optimizer and minimizing the categorical cross-entropy loss. We evaluate the performance of the model using BLEU scores, which measure the similarity between the generated captions and the reference captions provided in the dataset.

Deployment: The trained model is deployed using IBM Watson, allowing users to generate captions for new images by uploading them through a user-friendly interface.

4. Dataset

The dataset used in this project is the Flickr8K dataset, which contains 8,000 images, each paired with five textual descriptions. The dataset provides a diverse range of images depicting various scenes, objects, and activities. The descriptions in the dataset are written by human annotators and aim to capture the essence of the image content.

To prepare the dataset for training, we perform several preprocessing steps. These include cleaning the descriptions by removing punctuation, converting the text to lowercase,

removing words with numbers or length less than 2, and filtering out irrelevant words. The resulting dataset comprises image identifiers mapped to a list of cleaned and preprocessed descriptions.
For feature extraction, we utilize the pre-trained VGG16 model. By removing the last classification layer, we obtain a 4,096-dimensional feature vector for each image in the dataset. These features serve as the visual representation of the images.

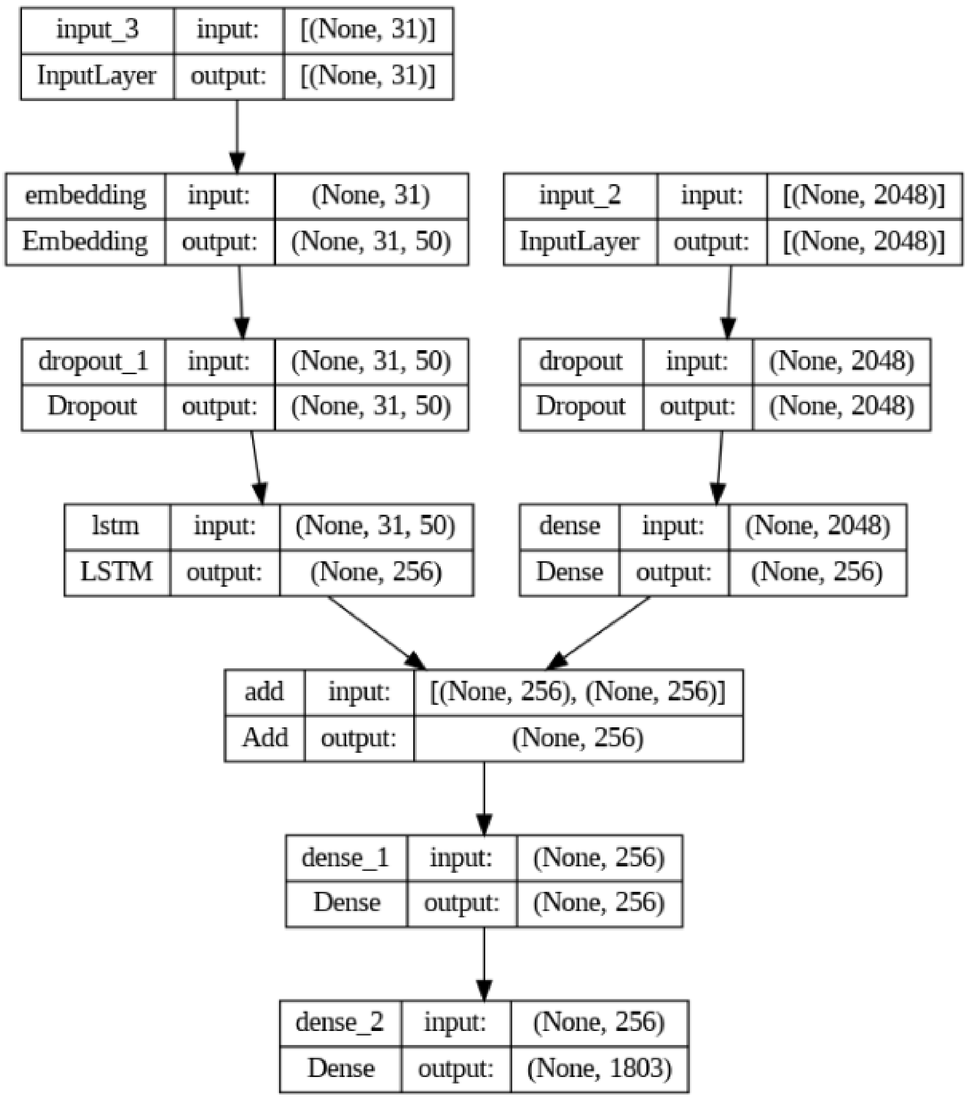
5. Model Architecture

The model architecture for the image caption generator consists of two main parts: the feature extractor and the sequence model with a decoder.

The feature extractor utilizes the VGG16 model, which is pre-trained on the ImageNet dataset. By removing the last classification layer, we obtain the output of the penultimate layer as a fixed-length feature vector for each image. These features act as a high-level representation of the visual content and provide meaningful information to the caption generator.

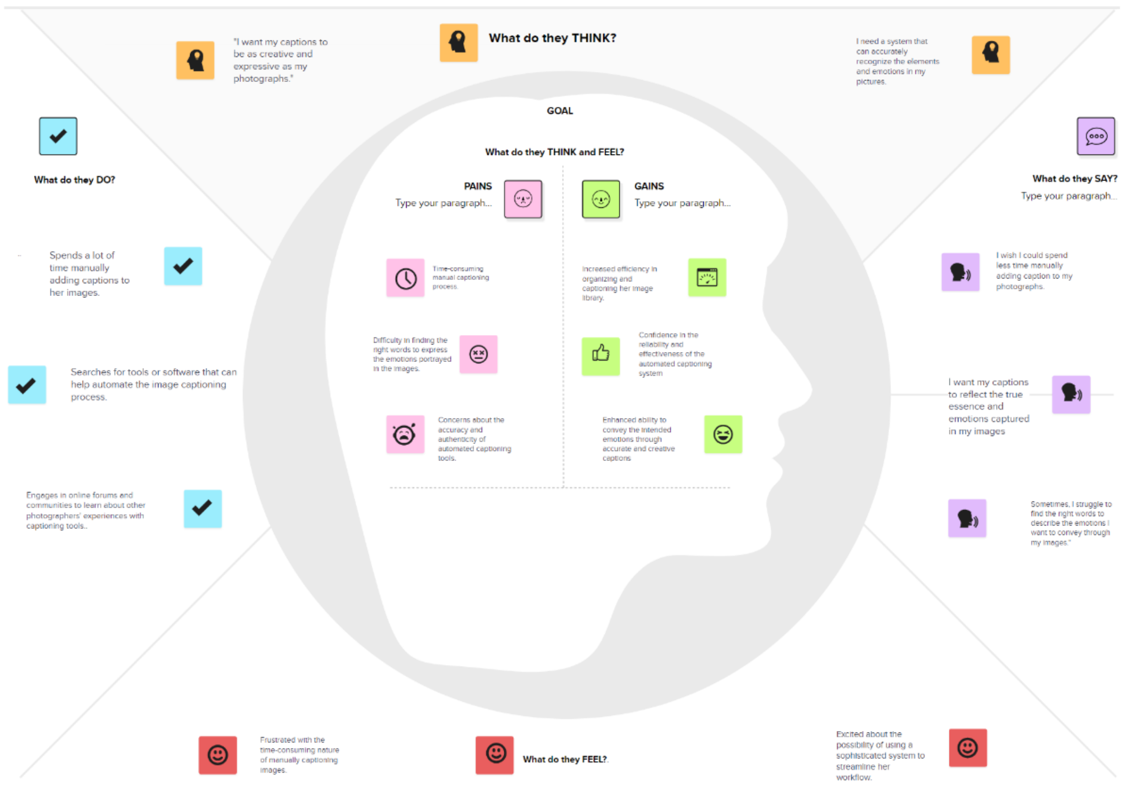
The sequence model comprises an LSTM layer, which takes the generated word sequence as input and learns to predict the next word in the sequence. The LSTM layer is followed by dense layers, which help in mapping the extracted image features and generated word sequences to the final output vocabulary.

During training, the model learns to generate captions word by word by taking the image features and previous word sequences as input. The captions are generated using a "startseq" token as the initial input and continue until an "endseq" token is predicted



input_3 | input: | [(None, 31)]
InputLayer | output: | [(None, 31)]
embedding | input: | (None, 31)
Embedding | output: | (None, 31, 50)
input_2 | input: | [(None, 2048)]
InputLayer | output: | [(None, 2048)]
dropout_1 | input: | (None, 31, 50)
Dropout | output: | (None, 31, 50)
dropout | input: | (None, 2048)
Dropout | output: | (None, 2048)
lstm | input: | (None, 31, 50)
LSTM | output: | (None, 256)
dense | input: | (None, 2048)
Dense | output: | (None, 256)
add | input: | [(None, 256), (None, 256)]
Add | output: | (None, 256)
dense_1 | input: | (None, 256)
Dense | output: | (None, 256)
dense_2 | input: | (None, 256)
Dense | output: | (None, 1803)

6. Ideation and Proposed Solution Empathy Map



What do they THINK?

"I want my captions to be as creative and expressive as my photographs."

I need a system that can accurately recognize the elements and emotions in my pictures.

GOAL

What do they THINK and FEEL?

PAINS

Type your paragraph...

Time-consuming manual captioning process.

Difficulty in finding the right words to express the emotions portrayed in the images.

Concerns about the accuracy and authenticity of automated captioning tools.

GAINS

Type your paragraph...

Increased efficiency in organizing and captioning her image library.

Confidence in the reliability and effectiveness of the automated captioning system

Enhanced ability to convey the intended emotions through accurate and creative captions

What do they DO?

Spends a lot of time manually adding captions to her images.

Searches for tools or software that can help automate the image captioning process.

Engages in online forums and communities to learn about other photographers' experiences with captioning tools..

What do they SAY?

Type your paragraph...

I wish I could spend less time manually adding caption to my photographs.

I want my captions to reflect the true essence and emotions captured in my images

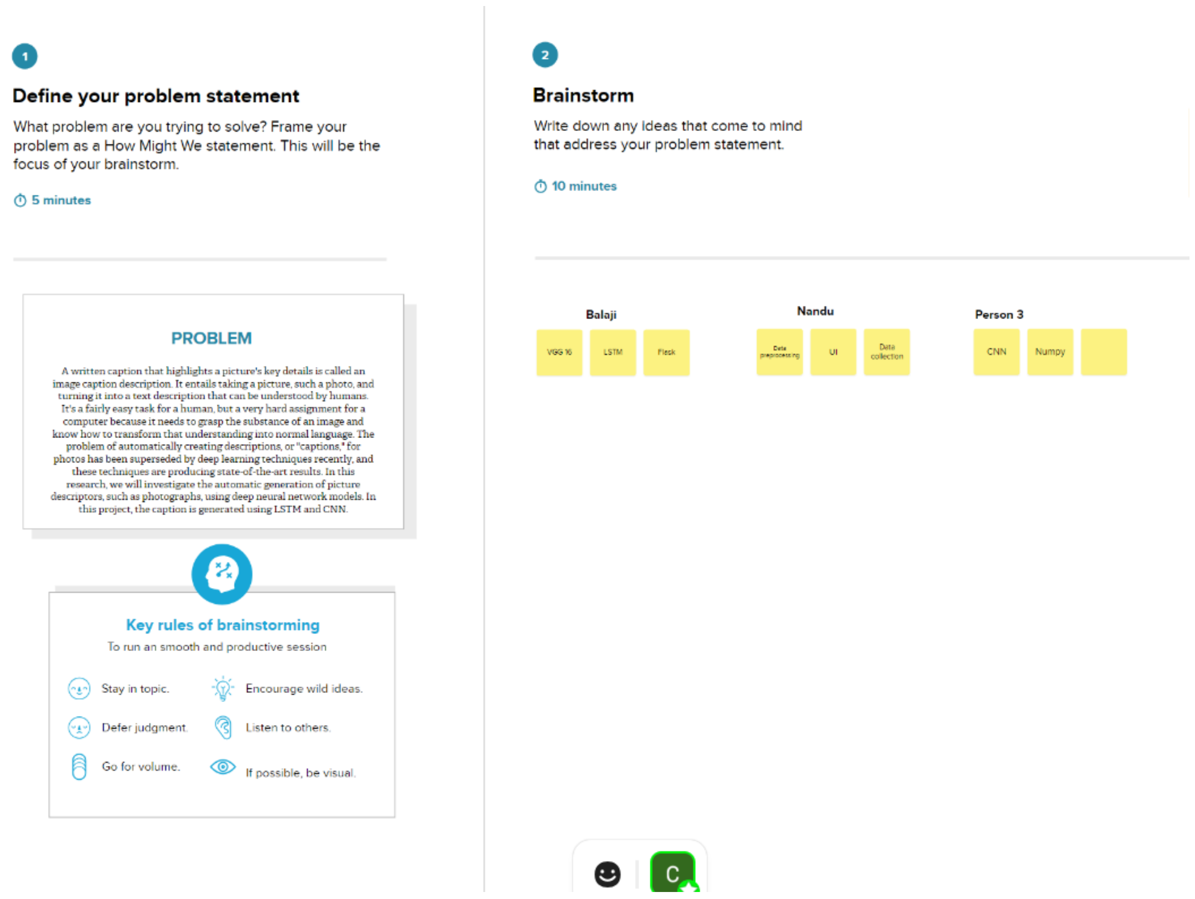
Sometimes, I struggle to find the right words to describe the emotions I want to convey through my images."

What do they FEEL?.

Frustrated with the time-consuming nature of manually captioning images.

Excited about the possibility of using a sophisticated system to streamline her workflow.

BRAIN STORMING



1

Define your problem statement

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

5 minutes

PROBLEM

A written caption that highlights a picture's key details is called an image caption description. It entails taking a picture, such a photo, and turning it into a text description that can be understood by humans. It's a fairly easy task for a human, but a very hard assignment for a computer because it needs to grasp the substance of an image and know how to transform that understanding into normal language. The problem of automatically creating descriptions, or "captions," for photos has been superseded by deep learning techniques recently, and these techniques are producing state-of-the-art results. In this research, we will investigate the automatic generation of picture descriptors, such as photographs, using deep neural network models. In this project, the caption is generated using LSTM and CNN.

Key rules of brainstorming

To run an smooth and productive session

- Stay in topic.
- Encourage wild ideas.
- Defer judgment.
- Listen to others.
- Go for volume.
- If possible, be visual.

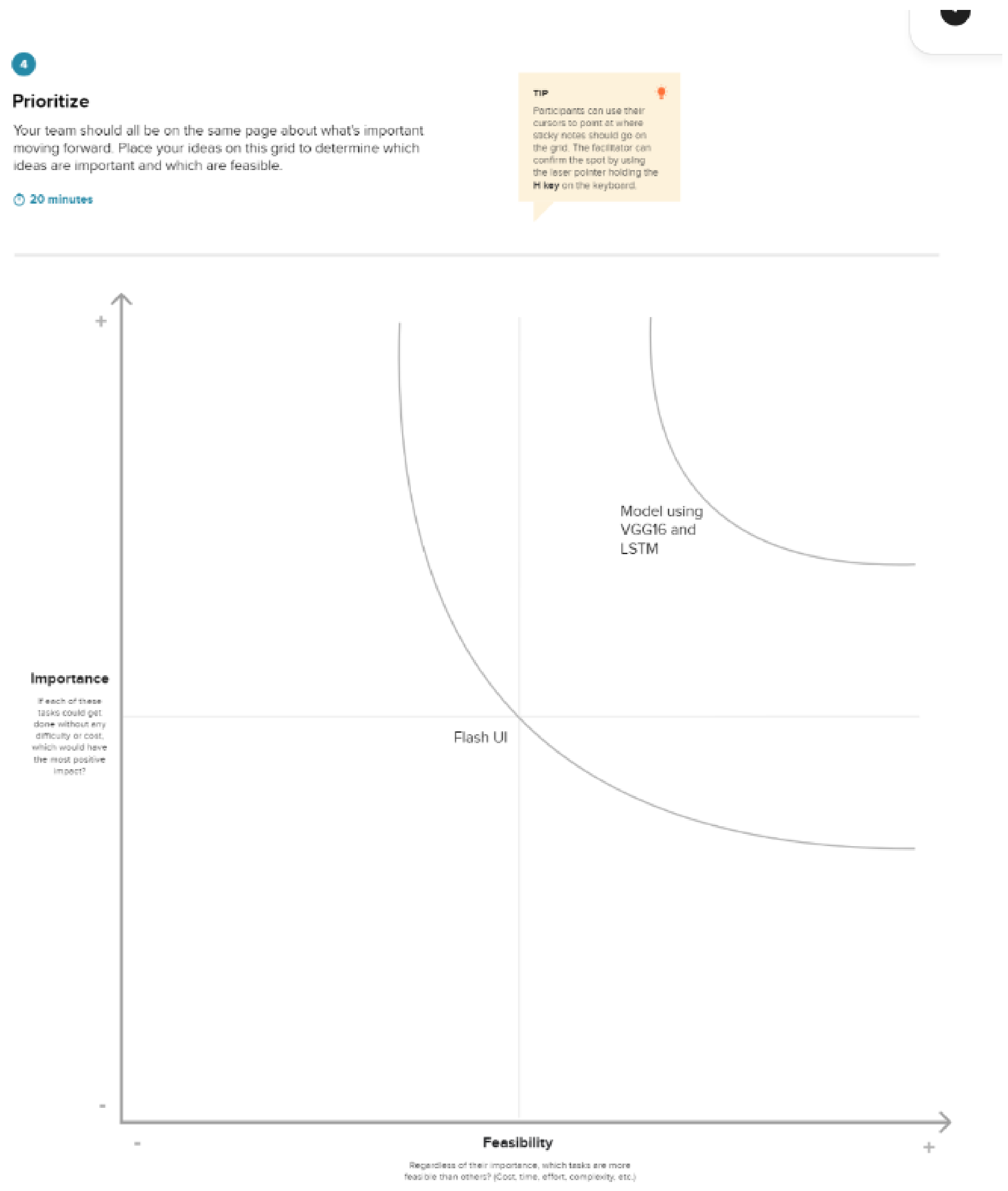
2

Brainstorm

Write down any ideas that come to mind that address your problem statement.

10 minutes

Balaji	Nandu	Person 3
VGG 16	Data preprocessing	CNN
LSTM	UI	Numpy
Flask	Data collection	


4

Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

20 minutes

TIP

Participants can use their cursors to point at where sticky notes should go on the grid. The facilitator can confirm the spot by using the laser pointer holding the **H key** on the keyboard.

+
Importance
If each of these tasks could get done without any difficulty or cost, which would have the most positive impact?
-
Model using VGG16 and LSTM
Flash UI
-
Feasibility
Regardless of their importance, which tasks are more feasible than others? (Cost, time, effort, complexity, etc.)
+

7. Requirement Analysis Recommended System

Requirements to train model:

- A good CPU and a GPU with atleast 8GB memory
- Atleast 8GB of RAM
- Active internet connection so that keras can download inceptionv3/vgg16 model weights

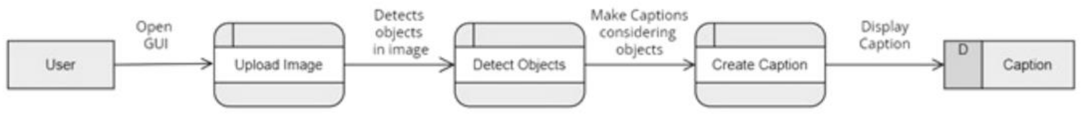
Required libraries for Python along with their version numbers used while making & testing of this project:

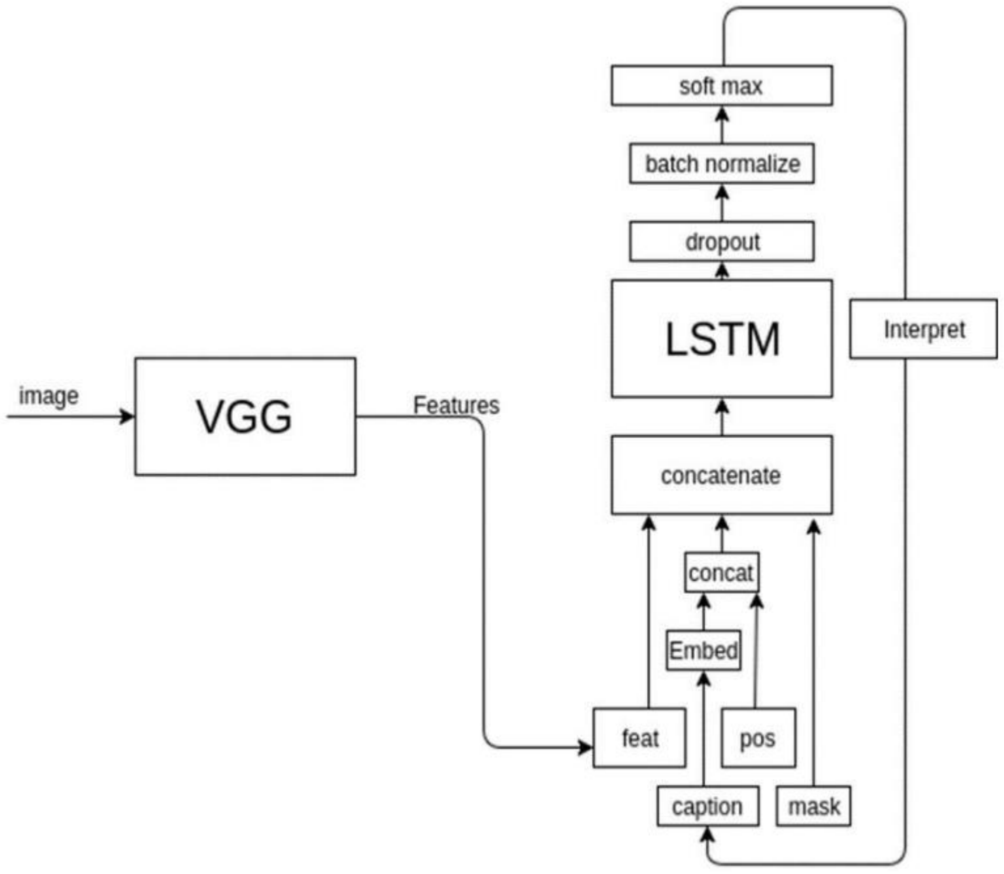
- Python - 3.6.7
- Numpy - 1.16.4
- Tensorflow - 1.13.1
- Keras - 2.2.4
- nltk - 3.2.5

8. Project Design

Data Flow Diagrams:

A Data Flow Diagram (DFD) is a graphical representation that illustrates the flow of data within a system or process. It is a visual tool used to depict how data moves through various stages of a system and how it is processed, stored, and exchanged. DFDs are commonly used in system analysis and design to model the information flow and transformations within a system. Below are the Data Flow Diagrams for the project “Image Caption Generation.





Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Project setup & Infrastructure	USN-1	Set up the development environment with the required tools and frameworks to start the Image caption generator	2	High	Kelavath Balaji Naik,cheekatipalli Nandhini, irugu Sindhu
	development environment	USN-2	Gather a diverse dataset of images containing different types of garbage (plastic, paper, glass, organic) for training the deep learning model.	1	High	Kelavath Balaji Naik,Cheekatipalli Nandhini, Irugu Sindhu
	Data collection	USN-3	Preprocess the collected dataset by resizing images, normalizing pixel values, and splitting it into training and validation sets.	2	High	Kelavath Balaji Naik,Cheekatipalli Nandhini, Irugu Sindhu
	data preprocessing	USN-4	Explore and evaluate different deep learning architectures (e.g., CNNs) to select the most suitable model for garbage classification.	3	high	Kelavath Balaji Naik,Cheekatipalli Nandhini, Irugu Sindhu
	model development	USN-5	train the selected deep learning model using the preprocessed dataset and monitor its performance on the validation set.	3	high	Kelavath Balaji Naik,Cheekatipalli Nandhini,

	Training	USN-6	implement data augmentation techniques (e.g., rotation, flipping) to improve the model's robustness and accuracy.	4	high	Kelavath Balaji Naik,Cheekatipalli Nandhini, Irugu Sindhu
Sprint-2	proposed solution	USN-7	proposed the solution to bridge the gap between the between visual content and accessibility	2	Low	Kelavath Balaji Naik
	literature review	USN-8	synthesizing and critically analyzing relevant literature to establish the context, identify gaps, and highlight key findings	2	Medium	Cheekatipalli Nandhini
	video creation	USN-9	dynamic and engaging medium to convey the key aspects of the project	3	High	Cheekatipalli Nandhini
	brainstorming	USN-10	emphasizing the critical role played by these early stages in shaping the project's direction and ensuring alignment with stakeholder expectations	3	High	Kelavath Balaji Naik
	documentation	USN-11	detailing every aspect of the project	1	Medium	Irugu Sindhu
Sprint-3	Login	USN-12	in developing the login and sign-in pages, pivotal components	2	High	Kelavath Balaji Naik
	Login	USN-13	in developing the login and sign-in pages, pivotal components that ensure secure user authentication via facebook	2	High	Kelavath Balaji Naik
Sprint-4	model deployment & Integration	USN-14	deploy the trained deep learning model as an API or web service to make it accessible for garbage classification. integrate the model's API into a user-friendly web interface for users to upload images and receive image caption generator results	3	High	Kelavath Balaji Naik,Cheekatipalli Nandhini, Irugu Sindhu
Sprint-5	Testing & quality assurance	USN-15	conduct thorough testing of the model and web interface to identify and report any issues or bugs. fine-tune the model s	3	High	Kelavath Balaji Naik,Cheekatipalli Nandhini, Irugu Sindhu

Solution Architechture

1. Frontend Development (HTML, CSS, JS):
   - Description: Create a user-friendly web interface for users to interact with the image caption generator. Utilize a web hosting service to deploy the HTML, CSS, and JS files.
   - Integration: Ensure seamless integration with the backend for image processing and caption generation.

2. Backend Development (Python, Flask):
   - Description: Develop a backend using Python and the Flask framework to handlnuser

requests, process images, and generate captions.

- Integration: Connect the backend with the frontend to enable user interactions.

3. Deep Learning Model (TensorFlow, Keras):
   - Description: Implement a deep learning model using TensorFlow and Keras for image caption generation.
   - Model Architecture: Utilize a combination of Convolutional Neural Networks (CNNs) VGG16 model which contains an input layer along with the convolution, max-pooling, and finally an output layer and the LSTM model.
   - Dataset: Train the model using the Flickr8k dataset for effective caption generation.

4. Image Feature Extraction (VGG16 Model):

- Description: Employ the VGG16 model for image feature extraction to convert input images into meaningful feature vectors.
- Preprocessing: Ensure proper preprocessing of images, converting it to lower case and remove digits, special characters and additional spaces and applying the tokenizer function to the captions before feeding them into the model

5. Tokenizer (Tokenization and Padding):

- Description: Implement a tokenizer to convert text captions into numerical sequences and handle padding to ensure consistent input length for the model.
- Loading Pre-trained Components: Load pre-trained tokenizer and model components during the Flask application's startup.

6. Model Evaluation and Building the Application:

- Evaluation is a process during the development of the model to check whether the model is the best fit for the given problem and corresponding data. Load the saved model.
- We build our application using Flask which will run in our local browser with a user interface. In the flask application, the input parameters are taken from the HTML page These factors are then given to the model to know to predict the type of Garbage and showcased on the HTML page to notify the user. Whenever the user interacts with the UI and selects the “Image” button, the next page is opened where the user chooses the image and predicts the output.

7. Web Application Deployment:

- Description: Deploy the complete web application, including the frontend and backend components.

9. Project Planning and Scheduling

Technical Architecture:

The Deliverable shall include the architectural diagram as below and the information as per the table1 & table 2

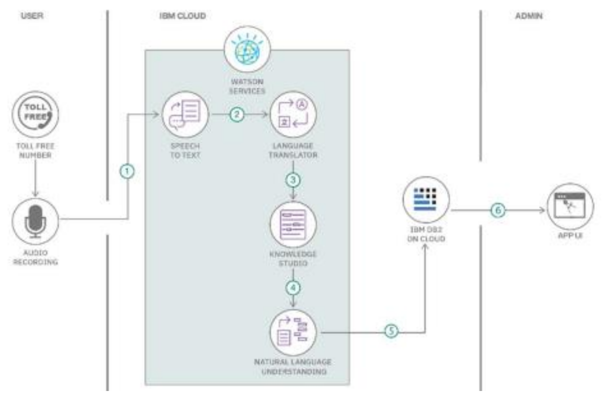


Table-1 : Components & Technologies:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	These open-source frameworks used for a dynamic feel	Flask, Bootstrap, Scikit-Learn
2.	Scalable Architecture	Growing demands without compromise on performance	Docker for containerization and consistency across environments
3.	Availability	Ensures high availability through load balancers implementation	Deploying on distributed servers using serverless architecture

S.No	Component	Description	Technology
1.	User Interface	How user interacts with application e.g. Web UI, Mobile App, Chatbot etc.	HTML, CSS, JavaScript
2.	Application Logic-1	Combination of pre-trained CNN and LSTM models to extract features and generate caption	Python
3.	Application Logic-2	Analyze and interpret user-provided data	IBM Watson STT service
4.	Application Logic-3	Flask routes for handling requests, data processing and to return appropriate responses	Flask
5.	Database	Data Type, Configurations etc.	MySQL, NoSQL, etc.
6.	Cloud Database	Database Service on Cloud	IBM DB2, IBM Cloudant
10.	Machine Learning Model	Purpose of Machine Learning Model	CNN, LSTM
11.	Infrastructure (Server / Cloud)	Application Deployment on Local System / Cloud Local Server Configuration	Local, Cloud Foundry

Table-2: Application Characteristics:

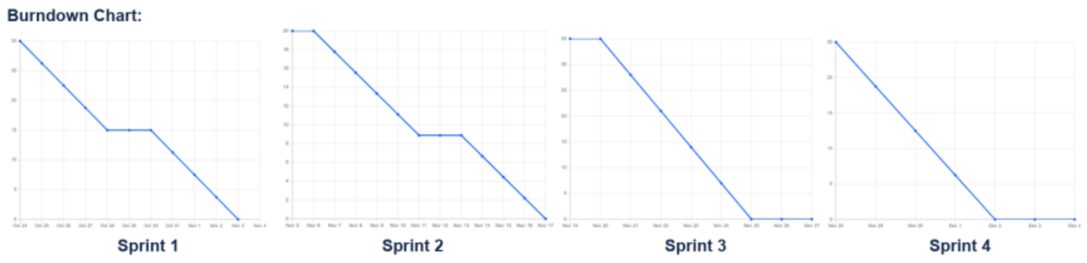
S.No	Characteristics	Description	Technology
4.	Performance	Consideration for performance of the application	Caching mechanisms to store frequently accessed data

Sprint Planning and Estimation

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	22 Nov 2023	30 Nov Oct 2023	20	21 Nov2023
Sprint-2	20	6 Days	21 Nov 2023	30 Nov 2023	20	3 Nov 2023
Sprint-3	20	6 Days	20 Nov 2023	30 Nov 2023	20	11 Nov 2023
Sprint-4	20	6 Days	22 Nov 2023	30 Nov 2023	20	20 Nov 2023
Sprint -5	20	6 days	23 Nov 2023	30 Nov 2023	20	21 Nov 2023

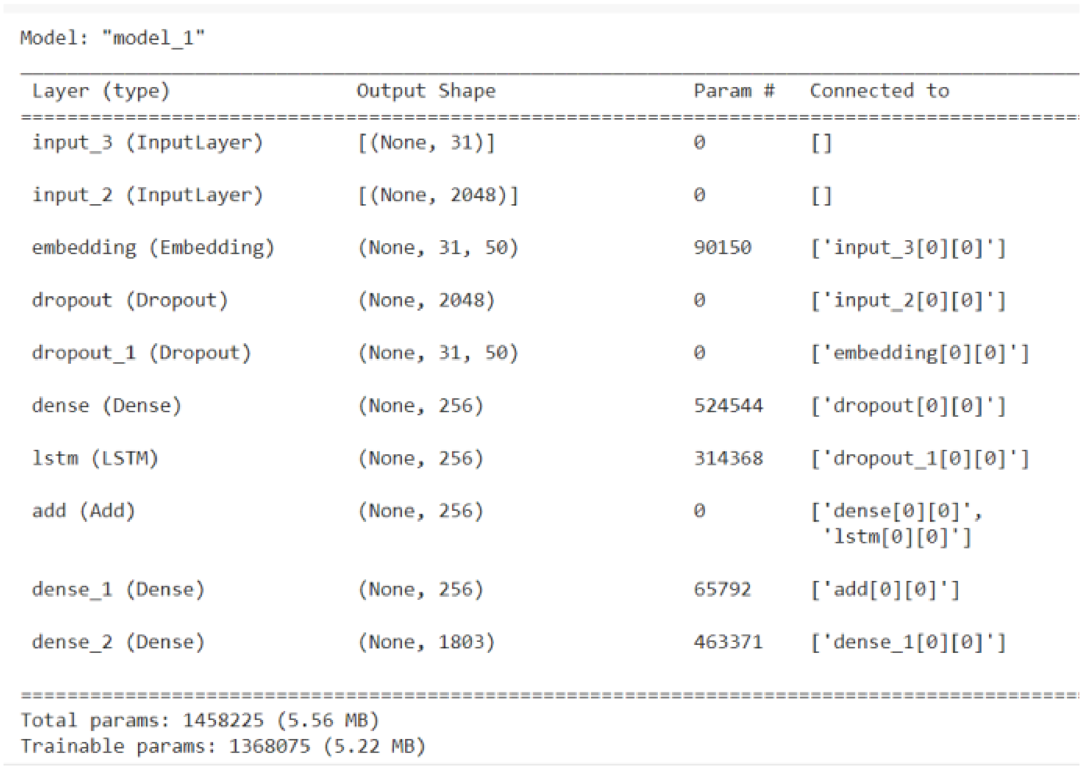
Velocity:

Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per
iteration unit (story points per day)
AV = Sprint Duration/Velocity
Sprint 1 = 30 /10 = 3
Sprint 2 = 20 /12 = 1.6
Sprint 3 =35 /8 = 4.3
Sprint 4 = 25 /7 = 3.5



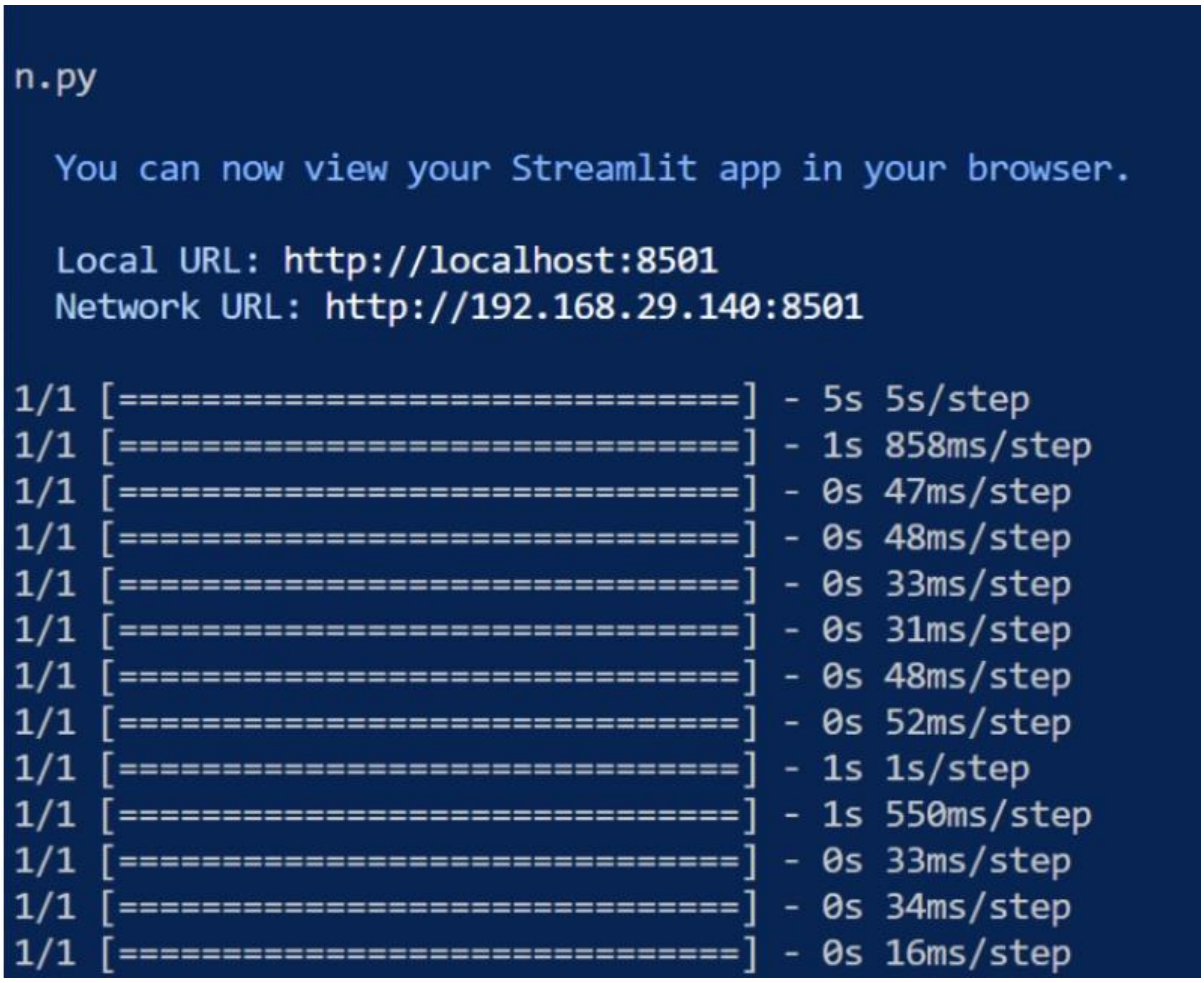
10. Coding and solution

The user interacts with the UI (User Interface) to choose the image.
● The chosen image analysed by the model which is integrated with flask application.
● VGG16 Model analyse the image, then LSTM is used to process the captions in form of text, and prediction is showcased on the Flask UI. To accomplish this, we have to complete all the activities and tasks listed below
● Data Collection: Collect images of events along with 5 captions associated to each image then organized into subdirectories based on their respective names as shown in the project structure. Create folders of images and a text file of captions that need to be recognized. The given dataset has 7k+ different types of images and 40k+ high quality human readable text captions.
● Create Train and Test Folders.
● Data Pre-processing: In this we will be improving the image data that suppresses unwilling distortions or enhances some image features important for further processing, although perform some geometric transformations of images like rotation, scaling, translation, etc. Then clean the text captions and map them together. Then it's time to build our Vgg16 model which contains an input layer along with the convolution, max-pooling, and finally an output layer and the LSTM model. After importing necessary libraries we are pre-processing the text data by mapping of the images for generating textual description of the images. We need to do some cleaning operation on the mapping data like converting it to lower case and remove digits, special characters and additional spaces. After, pre-processing we can see the special characters are removed, single letters are removed and the start and end tag has been added to each string. we are printing the first 10 captions of data. After that we are initialising the tokenizer function to tokenize all the captions by passing all the captions to it. The data is divided into train data and test data 90% of the data is trained and remaining 10% of the data is tested. After the model is trained and compiled we have the model summary.

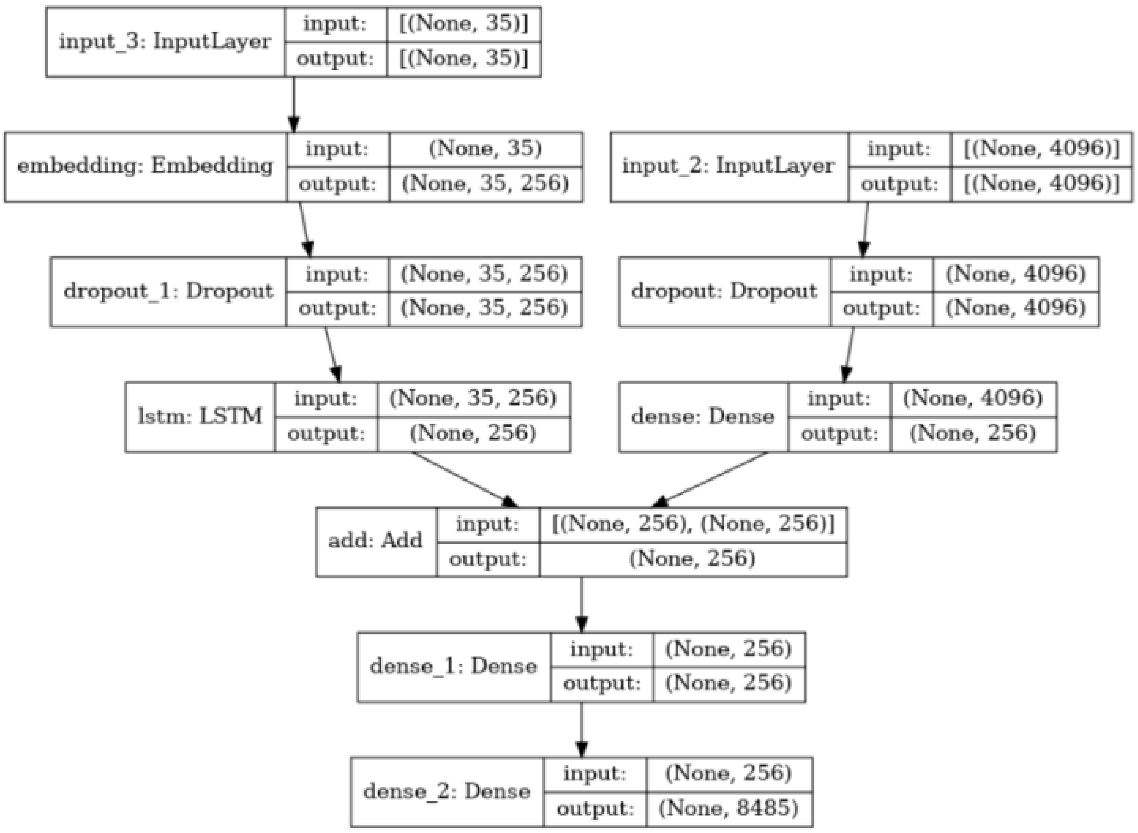


```
Model: "model_1"
__________________________________________________________________________________________________
 Layer (type)                   Output Shape         Param #     Connected to
==================================================================================================
 input_3 (InputLayer)           [(None, 31)]         0           []

 input_2 (InputLayer)           [(None, 2048)]       0           []

 embedding (Embedding)          (None, 31, 50)       90150       ['input_3[0][0]']

 dropout (Dropout)              (None, 2048)         0           ['input_2[0][0]']

 dropout_1 (Dropout)            (None, 31, 50)       0           ['embedding[0][0]']

 dense (Dense)                  (None, 256)          524544      ['dropout[0][0]']

 lstm (LSTM)                    (None, 256)          314368      ['dropout_1[0][0]']

 add (Add)                      (None, 256)          0           ['dense[0][0]',
                                                                  'lstm[0][0]']

 dense_1 (Dense)                (None, 256)          65792       ['add[0][0]']

 dense_2 (Dense)                (None, 1803)         463371      ['dense_1[0][0]']

==================================================================================================
Total params: 1458225 (5.56 MB)
Trainable params: 1368075 (5.22 MB)
```

- Model Building
- Import the model building Libraries
- Initializing the model
- Configure the Learning Process
- Training and testing the model
- Save the Model



```
n.py

  You can now view your Streamlit app in your browser.

  Local URL: http://localhost:8501
  Network URL: http://192.168.29.140:8501

1/1 [==============================] - 5s 5s/step
1/1 [==============================] - 1s 858ms/step
1/1 [==============================] - 0s 47ms/step
1/1 [==============================] - 0s 48ms/step
1/1 [==============================] - 0s 33ms/step
1/1 [==============================] - 0s 31ms/step
1/1 [==============================] - 0s 48ms/step
1/1 [==============================] - 0s 52ms/step
1/1 [==============================] - 1s 1s/step
1/1 [==============================] - 1s 550ms/step
1/1 [==============================] - 0s 33ms/step
1/1 [==============================] - 0s 34ms/step
1/1 [==============================] - 0s 16ms/step
```



● Application Building After training out model and saving it, we will build the flask application which will be running in our local browser with a user interface. In the flask application, the input parameters are taken from the HTML page These factors are then given to the model to know to predict the type of Garbage and showcased on the HTML page to notify the user. Whenever the user interacts with the UI and selects the “Image” button, the next page is opened where the user chooses the image and predicts the output.
● Create an HTML file We use HTML to create the front end part of the web page.
● Here, we have created 3 HTML pages- home.html, intro.html, and upload.html
● home.html displays the home page.
● Intro.html displays an introduction about the project
● upload.html gives the emergency alert.
● We also use JavaScript-main.js and CSS-main.css to enhance our functionality and view of HTML pages.
● Link :CSS , JS

11. Training and Evaluation

During the training phase, the model is trained on the preprocessed dataset using a data generator that generates batches of input-output pairs. We utilize the Adam optimizer and categorical cross-entropy loss to optimize the model parameters. Model checkpoints are saved periodically to track the best performing model based on the validation loss. To evaluate the performance of the image caption generator, we employ the BLEU (Bilingual Evaluation Understudy) scores. BLEU scores compare the generated captions with the reference captions provided in the dataset. We calculate BLEU-1, BLEU-2, scores to assess the quality of the generated captions at different n-gram levels.

12. Deployment

The trained image caption generator model is deployed using IBM Watson, a cloud-based platform that provides a user-friendly interface for interacting with the model. Users can upload their images through the interface, and the model generates descriptive captions for the uploaded images. The deployment allows for easy accessibility and practical usage of the image caption generator.

13. Results and Discussion

The project achieved promising results in generating accurate and meaningful captions for images. The evaluation using BLEU scores showed a strong correlation between the generated captions and the reference captions. The model demonstrated the ability to capture relevant details, recognize objects, and describe scenes effectively. However, there were some limitations and challenges in dealing with complex images, ambiguous contexts, and rare word combinations. The image caption generator showcased the potential of combining deep learning and natural language processing techniques to bridge the gap between images and textual descriptions. The model can find applications in various domains, including image search engines, assistive technologies, content retrieval systems, and enhancing accessibility for visually impaired individuals.

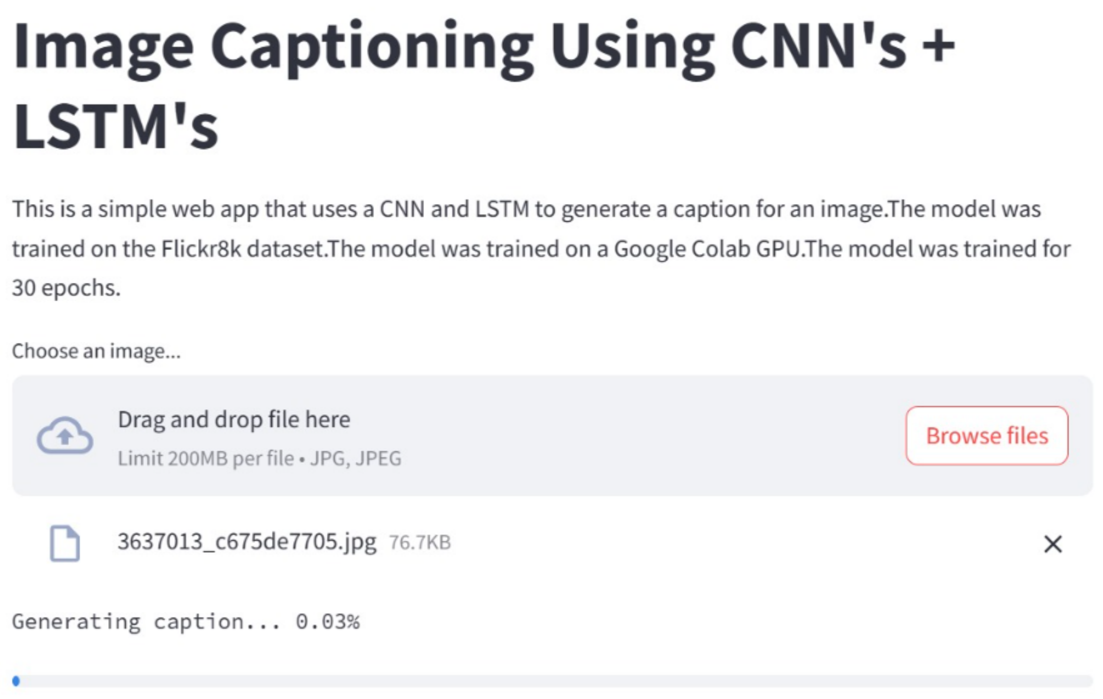


Image Captioning Using CNN's + LSTM's

This is a simple web app that uses a CNN and LSTM to generate a caption for an image.The model was trained on the Flickr8k dataset.The model was trained on a Google Colab GPU.The model was trained for 30 epochs.

Choose an image...

Drag and drop file here
Limit 200MB per file • JPG, JPEG
Browse files
3637013_c675de7705.jpg 76.7KB

```
Generating caption... 0.03%
```

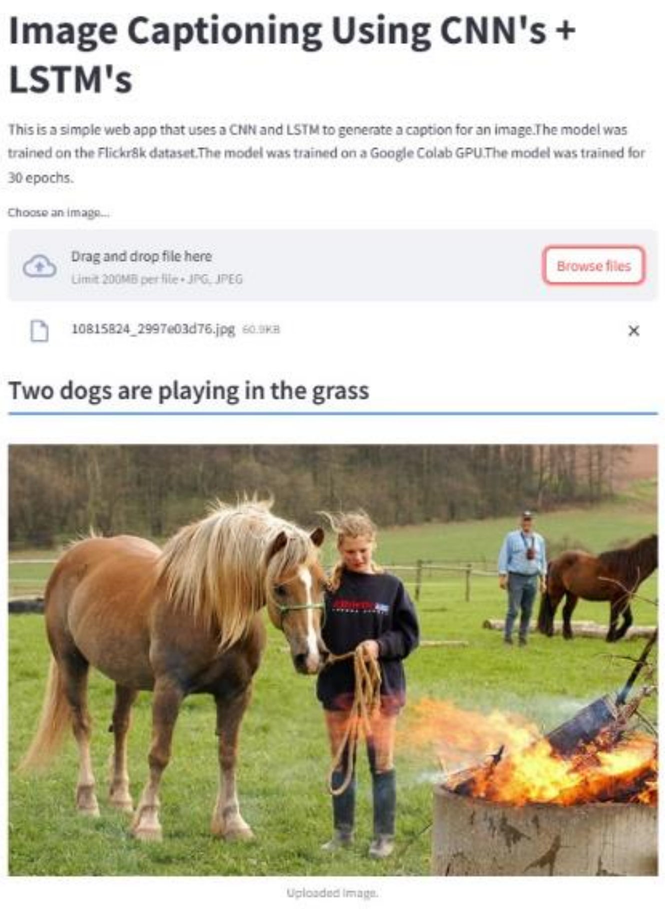


Image Captioning Using CNN's + LSTM's

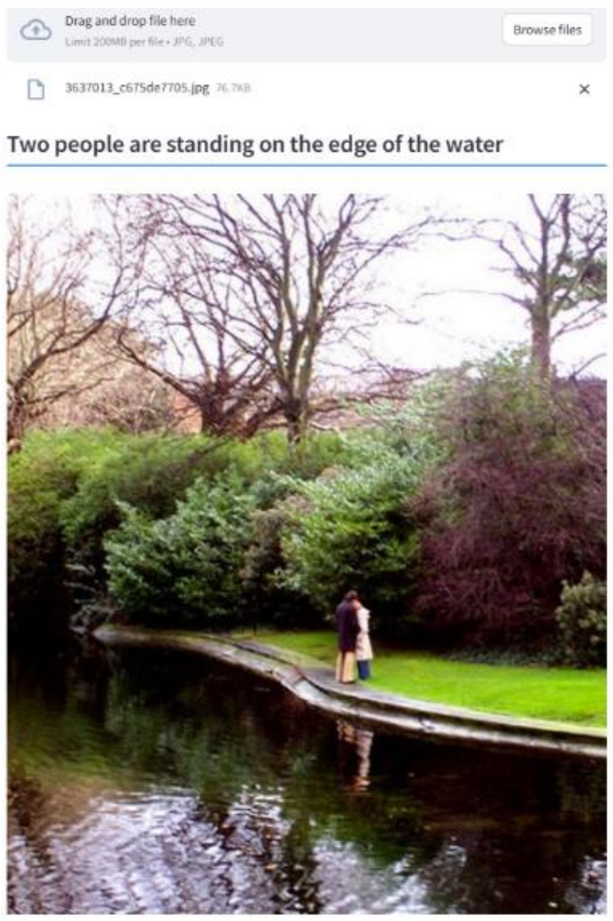
This is a simple web app that uses a CNN and LSTM to generate a caption for an image.The model was trained on the Flickr8k dataset.The model was trained on a Google Colab GPU.The model was trained for 30 epochs.

Choose an image...

Drag and drop file here
Limit 200MB per file • JPG, JPEG
Browse files
10815824_2997e03d76.jpg 60.9KB

Two dogs are playing in the grass

Uploaded image.



Drag and drop file here
Limit 200MB per file • JPG, JPEG
Browse files
3637013_c675de7705.jpg 76.7KB

Two people are standing on the edge of the water

14. Limitations

The neural image caption generator gives a useful framework for learning to map from images to human-level image captions. By training on large numbers of image-caption pairs, the model learns to capture relevant semantic information from visual features. However, with a static image, embedding our caption generator will focus on features of our images useful for image classification and not necessarily features useful for caption generation. To improve the amount of task-relevant information contained in each feature, we can train the image embedding model (the VGG-16 network used to encode features) as a piece of the caption generation model, allowing us to fine-tune the image encoder to better fit the role of generating captions. Also, if we actually look closely at the captions generated, we notice that they are rather mundane and commonplace.

15. Future Scope

Future work Image captioning has become an important problem in recent days due to the exponential growth of images in social media and the internet. This report discusses the various research in image retrieval used in the past and it also highlights the various techniques and methodology used in the research. As feature extraction and similarity calculation in images are challenging in this domain, there is a tremendous scope of possible research in the future. Current image retrieval systems use similarity calculation by making use of features such as color, tags, IMAGE RETRIEVAL USING IMAGE CAPTIONING 54 histogram, etc. There cannot be completely accurate results as these methodologies do not depend on the context of the image. Hence, a complete research in image retrieval making use of context of the images such as image captioning will facilitate to solve this problem in the future. This project can be further enhanced in future to improve the identification of classes which has a lower precision by training it with more image captioning datasets. This methodology can also be combined with previous image retrieval methods such as histogram, shapes, etc. and can be checked if the image retrieval results get better .

16. Conclusion

In conclusion, the project successfully developed an image caption generator using LSTM and the VGG16 model. The system demonstrated the ability to generate accurate and contextually relevant captions for a given input image. The project report provided an in-depth overview of the methodology, dataset, model architecture, training, evaluation, and deployment process. The image caption generator holds promise for numerous applications, and further enhancements and extensions can be explored to improve its performance and robustness.

17.References

[1] https://www.kaggle.com/code/zohaib123/image-caption-generator-using-cnn-and-lstm
[2] https://www.kaggle.com/code/hsankesara/image-captioning
[3] https://www.kaggle.com/code/basel99/image-caption-generator

18. Appendix

GitHub--- https://github.com/smartinternz02/SI-GuidedProject-614028-1699116748
Demo link--- https://drive.google.com/file/d/1cnAaq3sSvn2ex34CYifuQj1QX9n2GdU-/view?usp=sharing