A PRELIMINARY REPORT ON

A DATASET AND FRAMEWORK FOR MULTILINGUAL AND CODE-MIXED VISUAL QUESTION ANSWERING

SUBMITTED TO THE SAVITRIBAI PHULE PUNE UNIVERSITY, PUNE IN THE PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE

OF

BACHELOR OF ENGINEERING (COMPUTER ENGINEERING)

SUBMITTED BY

STUDENT NAME	EXAM NUMBER.
1. Sunanda Somwase	B150074274
2. Shivam Rajput	B150074267
3. Soham Pawar	B150074263
4. Tejas Shah	B150074270

DEPARTMENT OF COMPUTER ENGINEERING

PVG's COLLEGE OF ENGINEERING AND TECHNOLOGY &; G K PATE(WANI) INSTITUTE OF MANAGEMENT

44, VIDYANAGARI, PARVATI, PUNE 411009
SAVITRIBAI PHULE PUNE UNIVERSITY
2021-2022

CERTIFICATE

This is to certify that the project report entitled

A DATASET AND FRAMEWORK FOR MULTILINGUALAND CODE-MIXED VISUAL QUESTION ANSWERING

Submitted by

STUDENT NAME	EXAM NUMBER.
1. Sunanda Somwase	B150074274
2. Shivam Rajput	B150074267
3. Soham Pawar	B150074263
4. Tejas Shah	B150074270

is a record of bonafide work carried out by him/her, in the partial fulfillment of the Term-work of Fourth year in Computer Engineering of Savitribai Phule Pune University at Pune Vidyarthi Griha's College of Engineering and Technology & G.K. Pate (Wani) Institute of Management, Pune under Savitribai Phule Pune University, Pune. This work is done during the academic year 2020-21.

Prof.Ms.M.V.Marathe.

(Project Guide)

Date: - 20/05/2022

Place: - Pune

ACKNOWLEDGMENT

It gives us immense pleasure to present our preliminary project report on ”A Dataset and Framework for Multilingual and Code-Mixed Visual Question Answering”. We would like to take this opportunity to thank our internal guide Prof. M. V. Marathe for giving us all the help and guidance which we needed.

We are grateful for her valuable suggestions and continuous support. We would also like to express our special thanks of gratitude to Mr. Atharva Kulkarni and Mr. Amey Hengle (Alumnae of Computer Department, PVG) for mentoring us throughout this journey for finding better solutions. We would also like to thank our reviewers Prof. B. V. Julme and Prof. M. S. Pokale for giving valuable suggestions and time.

We are also grateful to Prof. D. D. Sapkal (Head of the Computer Engineering Department) and Prof. M. V. Marathe for their constant support and suggestions.

Tejas Shah

Soham Pawar

Shivam Rajput

Sunanda Somwase

ABSTRACT

Problems at the intersection of vision and language are of significant importance both as challenging research questions and for the rich set of applications they enable. Visual Question Answering (VQA) is a computer vision task where a system is given a text-based question about an image, and it must infer the answer. The switching in between the languages is called code-switching or code-mixing, depending upon the type of mixing. The majority of the existing techniques on Visual Question Answering (VQA) focus on English questions only. However, many applications such as medical imaging, tourism, and visual assistants require a multilinguality-enabled module for their widespread usages. Implementing a framework for Multilingual and Code-Mixed Visual Question Answering would allow us to tap into the huge potential of applications. Furthermore, research in Indian regional languages is scarce and mostly limited to Hindi. In our work, we focus on VQA in Marathi and Marathi+English code mix language. We aim to create a robust dataset first since no such dataset is available. Based on this dataset, we aim to deploy a unified framework that answers visual questions with satisfactory accuracy.

Contents

Chapter 1

INTRODUCTION

Visual Question Answering (VQA):

In simple words, Visual Question Answering means answering a question based on the image provided along with it. This infant domain of computer science is an amalgamation of Natural Language Processing(NLP) and Computer Vision. Combining Natural Language Processing with Computer Vision for high-level scene interpretation is a recent trend, e.g., image captioning. These works have benefited from the rapid development of deep learning for visual recognition (object recognition and scene recognition and have been made possible by the emergence of large image datasets and text corpus. Beyond image captioning, a natural next step is a visual question answering. Progress in VQA will lead to improved chatbots, better inclusion of people with disabilities and better aid for autistic individuals.

VQA is a challenging problem that requires complex reasoning and algorithms over visual elements to provide an accurate answer to a natural language question. An efficient VQA system can be used to build an Artificial Intelligence (AI) agent which takes a natural language question and predicts the decision by analyzing the complex scene(s). VQA requires language understanding, fine-grained visual processing, and multiple steps of reasoning to produce the correct answer. Hence training such models usually requires high computational systems and a large, comprehensive dataset. [7]

Code Mixing or Code Mixed Languages:

Multilingual speakers often switch back and forth between their native and foreign languages to express themselves. This phenomenon of embedding the morphemes, words, phrases, etc, of one language into another is popularly known as code-mixing and such a mixture of languages(two or more than two) is know as a code mixed language. With the unimaginable growth of social media, the ease of instant messaging and the onset of informality, there has been a rapid rise in acceptance of code mixed languages in today's global society.

Traditionally, some studies have viewed the mixing of two independent codes as a lack of fluency of the segment of the population in either of the languages. However, an alternate perspective argues that the mixing of two traditionally isolated linguistic codes potentially creates a third legitimate code. Researchers have also presented several socio-cultural reasons and motivations for switching. There have been studies to depict the usage of a particular language based on the emotional attachment and the sentiment of the person towards that topic. Hence, from a linguistics point of view, studying code mixing becomes absolutely necessary. [6] [3] [10]

1.1 Motivation

Sectors like tourism, food, education, marketing, etc. have recently started using code-mixed languages in their advertisements to attract a larger consumer base. To build an AI agent which can serve multilingual end users, a VQA system should be put in place that would be language agnostic and tailored to deal with the code-mixed and multilingual environment. It is worth studying the VQA system in these settings which would be immensely useful to a very large number of the population who speak and write in more than one language. Such a situation can be prominently observed in India where there are more than 250 regional languages. Combining these together or adding a foreign language to the mix can create a vast number of code mixed languages. While at this point in time it is difficult to handle all such languages, we would at least like to focus on the major ones. But a lack of dataset hinders progress considerably. Moreover, there is no such dataset available for Marathi + English code mixed language which can be utilized for building a VQA system. Additionally, due to the lack of a gold standard dataset and other reasons, the VQA system for Marathi + English code mixed language is yet to be built. As much as we can, we aim to improve this situation.

1.2 Problem Definition

As stated in the previous section, there is no gold standard dataset and the VQA system present for Marathi + English code mixed language. Hence the problem is naturally divided into two parts.

- In the first part, we aim to create a gold standard dataset for Marathi + English code mixed language, which can be employed in the development of a VQA system.
- In the second part, we aim to develop a gold standard model for generating fairly accurate answers for Marathi, English and Marathi + English code mix questions accompanied by an image.

1.3 Project Scope

Administrators and users are stakeholders for this project. The system will take an input from the user which will contain an image and a question in either Marathi, English or Marathi + English code mixed language. A key component of the project will be an AI-based model i.e VQA system, which will be able to produce answers to the question. The development, training, and deployment of the VQA system is the key component of the project, hence the scope of the project is defined based on the efficiency of the VQA system itself. For this, first and foremost a comprehensive dataset will need to be created. It also makes a fair share of the scope. This project can be termed as a research and development project which can have varied scope.

1.4 Limitations

- Accuracy of translated and transliterated text inputs:

  - According to a survey conducted by Google in 2017, the average accuracy of Google translate is 85
  - The accuracy of translation and transliteration models is one of the limitations of this project we have employed these tools to obtain our gold standard dataset.

- Lack of powerful image processing tools:
  - The dataset we are working with is as huge as 80 thousand images. Processing such an amount of dataset itself is a limitation.
- Lack of powerful Computation tools:
  - This is a student project, hence we are limited by our computational power. Limited computation power might hinder us from deploying complex models which in turn will affect the credibility of the answers generated.

Chapter 2

LITERATURE SURVEY

Visual Question Answering (VQA) is a challenging task that has received increasing attention from both the computer vision and the natural language processing communities. Given an image and a question in natural language, it requires reasoning over visual elements of the image and general knowledge to infer the correct answer.
[1]

Lots of VQA models are proposed over time which are focused on datasets such as VQA [2], MS-COCO [8], Flicker30k [9] etc. These VQA systems are limited to the type of dataset used to train the system. There are different techniques for different types of images like natural images, artificial images, mathematical plots, etc. The performance of the models also depend on the type of question asked. Hence the work done in this domain is generally varied and diverse.

2.1 Approaches for solving a VQA task:

Various models have been proposed which tackle the problem from different angles. These are the approaches that we surveyed.

2.1.1 Compositional VQA Models :

In compositional approach the questions are interpreted as composition of multiple subtasks. For example - what color shirt the boy is wearing? can be interpreted as composition of finding the boy and locating the shirt and then naming the color of the shirt. There are different frameworks available for processing the tasks in sub-steps.

2.1.2 Attention Based Models:

Attention mechanisms form the basis of a lot of models out there. Attention models are popular as they are based on the fact that certain regions in image and certain words in text are more important than others. But for attention models to work, the dataset must include the bounding-box annotations that contain information about the different objects in different regions in the image. This restricts the model from working on any dataset. There can be two ways of encoding local features in an image. One is to divide images in uniform grid-size and then specify grid-wise features and the other is by defining bounding box annotation around different objects in the images. [4]

The VQA models which use annotation box as local feature encoding needs to pass an image and the question along with annotation box while training. An improved approach known as the Focused Dynamic Attention model was introduced which suggests using only those bounding box whose labels match with the words in the question. In this way a lot of redundant processing is saved with a little extra effort of matching the labels. Later the matched labels are classified using the ResNet. In some cases, this might affect the accuracy of the generated answer.
As attention based models work with only specialized datasets, it is necessary to know which datasets have been designed to cater to such models. Openly available Datasets are:

- DAQUAR
- COCO-QA
- VQA (v1 v2)

2.2 Approaches for solving Code Mixing Problems:

Code mixing problems by their inherent nature are very intricate. Any problem that can be posed in a pure language can be posed for any code mix language but unfortunately, the same cannot be said for converting pure language solutions to code mixed solutions.

In our survey, we tried to narrow our field of relevance to question answering problems in code mix languages and problems in Indian regional languages.

Kushagra Singh et al created a corpus for Hindi English code mix Part of Speech(POS) tagging. This helped understand the unique grammar people tend to use in code mixed ”Hinglish” and shed light upon which POS is generally in Hindi and which in English. Sahil Swami et al further built on this to generate a comprehensive dataset for Sarcasm detection in Hinglish. Thanks to their efforts, Hindi english code mix language is now fairly well understood.

A lack of work becomes apparent when one turns their attention to code-mix VQA. Very few gold standard datasets are available to build advanced complex systems based off of. Since a lot of code mix combinations of languages are possible it becomes exponentially harder to construct a well-rounded dataset.

Deepak Gupta et. al. have developed an effective deep learning framework for multilingual and code- mixed visual question answering.Their proposed model is capable of predicting answers from the questions in Hindi, English or Code-mixed (Hinglish: Hindi-English) languages. As there is no available dataset in English-Hindi VQA, they firstly create a Hindi and Code-mixed VQA datasets by exploiting the linguistic properties of these languages.

In our extensive survey, nowhere did we encounter work done in Marathi + English code mix. This came as a surprise since tens of millions of people speak Marathi. A lot of advanced work is hindered by the absence of a corpus. By the end of the project, we hope to have made a difference.

Chapter 3

SOFTWARE REQUIREMENT SPECIFICATION

3.1 Introduction

The system requirements of a research project are varied and hierarchical. In this project, the requirements needed for training the model, especially the hardware requirements differ vastly than the requirements on the user's end. In this section of the report, the requirements are specified in great detail.

3.1.1 Project Scope

Administrators and users are stakeholders for this project. The system will take an input from the user which will contain an image and a question in either Marathi, English or Marathi + English code mixed language. A key component of the project will be an AI-based model i.e VQA system, which will be able to produce answers to the question. The development, training, and deployment of the VQA system is the key component of the project, hence the scope of the project is defined based on the efficiency of the VQA system itself. For this, first and foremost a comprehensive dataset will need to be created. It also makes a fair share of the scope. This project can be termed as a research and development project which can have varied scope.

3.1.2 User Classes and Characteristics

Computer vision (CV) and Natural language processing (NLP) are the emerging fields in recent years so the class of users will be limited because technology is not yet used in most of the fields. VQA is directly applicable to a variety of applications of high societal impact that involve humans eliciting situationally-relevant information from visual data; where humans and machines must collaborate to extract information from pictures.

Here is a look at those use cases:

1. An aid to the visually impaired.
2. Surveillance.
3. Interaction with personal assistants.

3.1.3 Assumptions and Dependencies

The user is assumed to have a fair idea of how to use a web-based GUI for proper output. It is assumed that the system can be compatible with more users at the same time. Users are assumed to have a fair estimate of execution times, to get accurate results with less chance of errors.

3.2 External Interface Requirement

List of Modules and Functionalities:

3.2.1 User Interface

Web-Based GUI where the users can upload an image and ask the question in the text area related to the image. The question may be in pure English, pure Marathi, or code-mix Marathi + English format. The GUI has a “Send” button that is used to send data. After receiving the data, the predicted answer is displayed on the screen.

3.2.2 Hardware Interface

Training Phase

Dataset consists of 80k images and 100k questions and answers from the MS-COCO dataset during training the model so we require 4GB to GB Graphics Processing Unit(GPU) and up to 32GB of RAM.

Deployment Phase

The model will be deployed on Heroku web server and will be accessible from any device (for example: mobiles, personal desktops) with an internet connection.

3.2.3 Software Interface

The system uses the following software interfaces:

- Python 3.8
- Google Collab
- Heroku Cloud Services
- TensorFlow
- NLTK

3.2.4 Communication Interface

Web-based GUI that takes the image and question from the user. That Image and question are given to the image processing unit and text processing unit respectively to extract the relevant features from it and return the correct answer to the end-user in a language that prefers while writing the question. For example, the user provides an image and asks a question(“Bench var koni baslay ka? ”). Then the model interprets the question with an image and produces a relevant response as (“Ho” or “Nahi”). The main objective is to create a CodeMix VQA system on the regional language but especially our focus is on (Marathi) because no prior work is done on it.

3.3 Non-functional Requirements

3.3.1 Performance Requirements

- Response Time- It will depend on the user's internet speed as well as a few other factors.

$$ResponseTime = T_u + T_p + T_r + T_c$$

- Capacity- The system will be able to cater to 150 people at any given time as this is the maximum limit of the basic Heroku Web Server.

3.3.2 Safety Requirements

The System has been tested on varieties of input and it is ready to use. There is as such no safety requirement, but the only thing that needs to be taken care of is that the relevant question has to be asked to the system to get an accurate answer. For example the question is " kiti mule ground var khelat ahet?" it should not be "ahet var kiti ground mule ?". The question should be semantically correct. Similarly, to get a valid answer, the image and the question asked should be visibly related. If the question asked is "Where is the Banana?", then it is expected that the image contains at least one banana.

Furthermore, care will be taken that none of the images and questions are stored in any database to protect the privacy of users.

3.4 System Requirements

3.4.1 Functional Requirements

- Users must be able to upload image and the question.
- Model must be able to generate a group of relatable answers.

- User must get a valid answer.

3.4.2 Software Requirements

- Operating System: Windows / Linux / MacOS
- IDE: PyCharm, Jupyter Notebook, VS Code
- ML tools: TensorFlow, Keras, scikit-learn
- Data Handling and Visualization: Pandas, Numpy, Matplotlib, NLTK
- Hosting Provider: Heroku

3.4.3 Hardware Requirements

- Processor: Intel Core i5 or i7 (recommended)
- RAM: Minimum 32 GB (recommended)
- Graphics: 4GB
- GPU: An NVIDIA GPU with CUDA support and at least 12 GB of VRAM

3.5 Analysis Models: SDLC Model to be applied

3.5.1 Iterative SDLC Model

The iterative SDLC model does not need the full list of requirements before the project starts. The development process may start with the requirements to the functional part, which can be expanded later. The process is repetitive, allowing us to make new versions of the product for every cycle. Every iteration includes the development of a separate component of the system, and after that, this component is added to the functional developed earlier. Speaking with math terminology, the iterative model is a realization of the sequential approximation method; that means a gradual closeness to the planned final product shape.

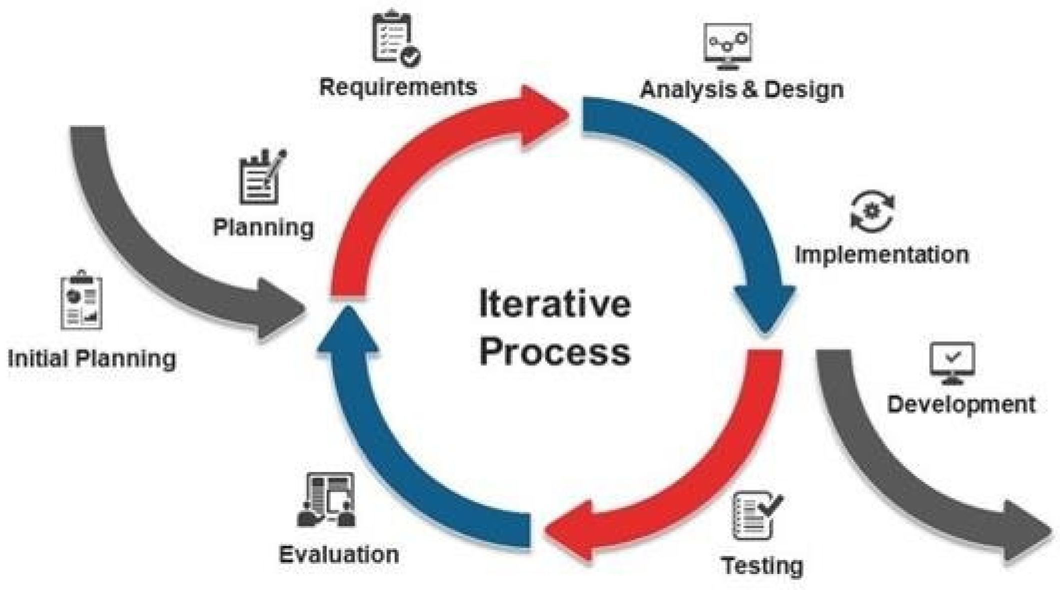


Figure 3.1: Iterative Process

The key to successful use of an iterative software development life cycle is rigorous validation of requirements, and verification and testing of each version of the software against those requirements within each cycle of the model. As the software evolves through successive cycles, tests must be repeated and extended to verify each version of the software

3.5.2 Requirements Gathering

All the functional and non-functional requirements of the project were identified. Interaction with the users and all other stakeholders of the project was conducted to identify all the requirements starting from important features like maintaining an audit trail, to basic features like the look and the feel of the user interface. The different requirements mainly fall into categories:

- System features
- User requirements
- User interface

3.6 Design

The first step was database design. A complete database required for the implementation of this project was designed. The second step was project design. The project was designed based on a framework. The framework uses three layers:

- Business entities layer:It identifies all the entities used in the project.
- Business logic layer:This layer operates on the business entity to achieve the goals.
- Data access layer: his layer serves as an interface between backend and the service

Chapter 4

SYSTEM DESIGN

4.1 System Architecture

This diagram shows the inner working of the proposed model. It will be a modified attention based model. Image feature extraction and text feature extraction will be handled differently by parallel pipelines. These features will later be combined to generate a coherent and valid answer.

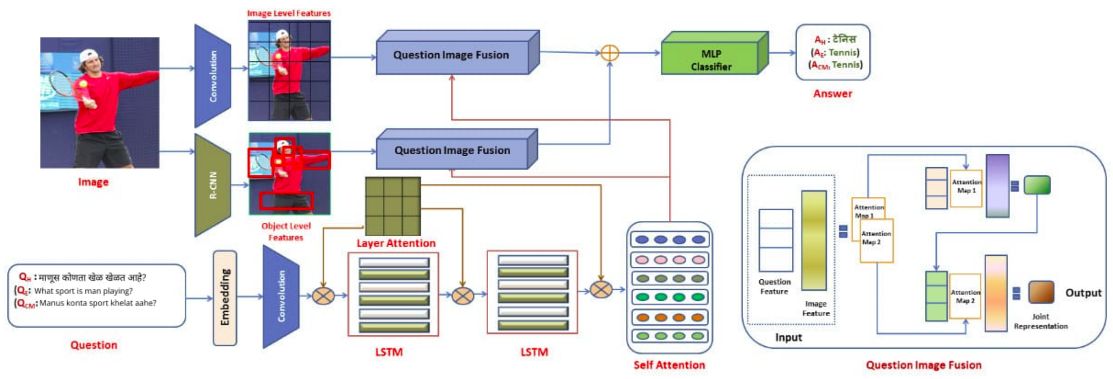


Figure 4.1: System Architecture Diagram

4.2 Data Flow Diagram

A data flow diagram shows the way information flows through a process or system. It includes data inputs and outputs, data stores, and the various sub-processes the data moves through. DFDs are built using standardized symbols and notation to describe various entities and their relationships.

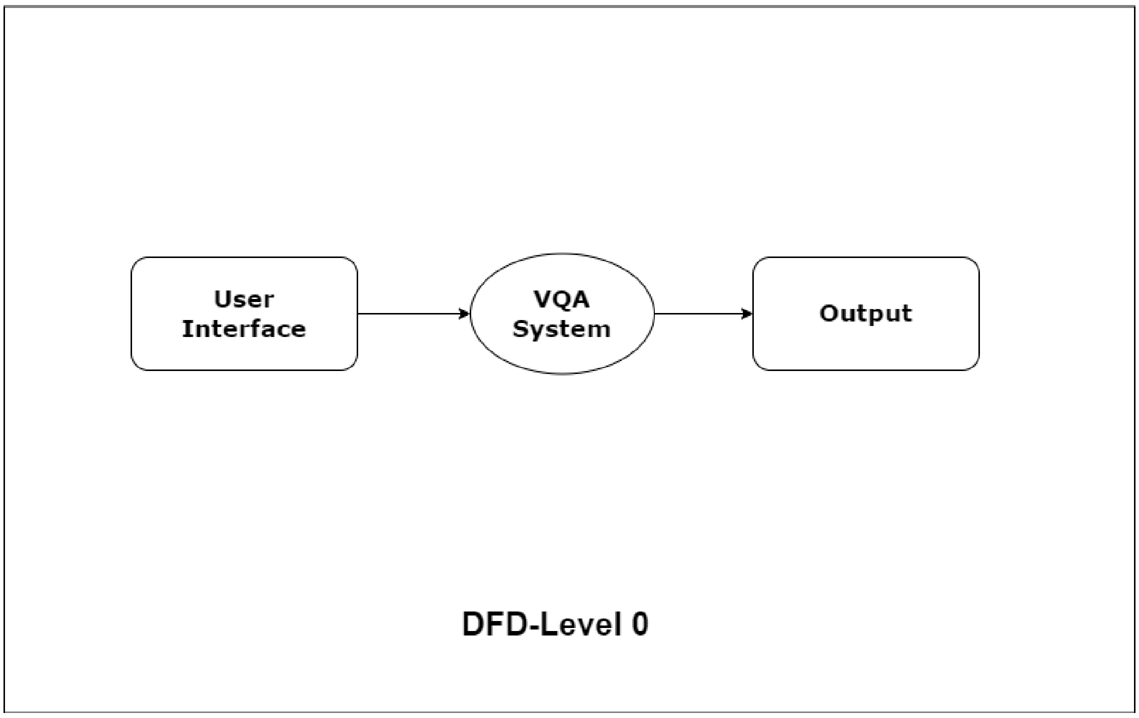


Figure 4.2: Data Flow Diagram

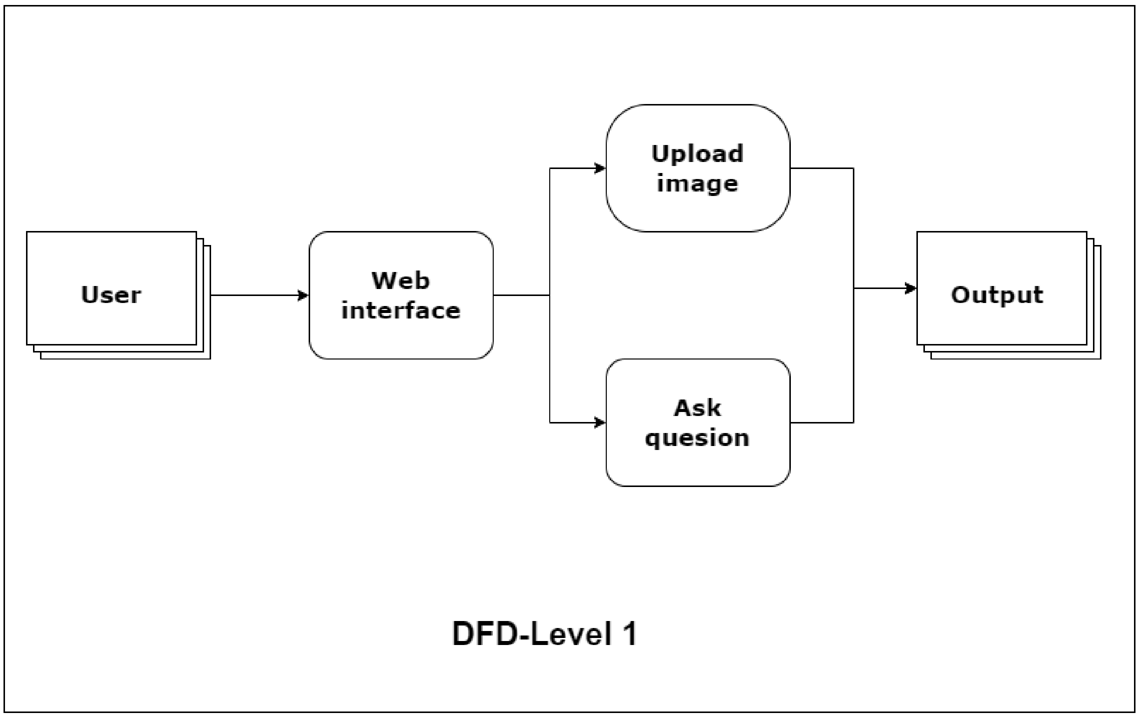


Figure 4.3: Data Flow Diagram

4.3 UML Diagrams

Software industry uses some well-defined process model for developing software. This model is called Software Engineering Process(SEP) or Software Development Process (SDP). The unified Software Development Process (USDP) is a popular, widely used industry standard SDP. In short it is also called as Unified Process (UP). Unified Process co-exists with Unified Modelling Language(UML) which is a virtual language part of the system or project, UP addresses the process part of software development. UML is standard modeling language and notation for object oriented analysis and design.These notations are used to visualize the design of system and widely accepted among the software development organizations.

4.3.1 Sequence Diagram

Sequence Diagrams are used to show the interaction between objects when system becomes operational. The primary focus in a sequence diagram is the sequencing of events. Time runs top to bottom and events take place in the time line. The diagram contains a very unique element called the object lifeline and shows the object's interaction with other objects with respect to the object's lifeline. Instance lifeline is denoted by sketching the dashed line below the object instance. In a sequence diagram, the focus of control shifts from one object to another as one object activates the other. The entire line denotes the duration of instance over time.

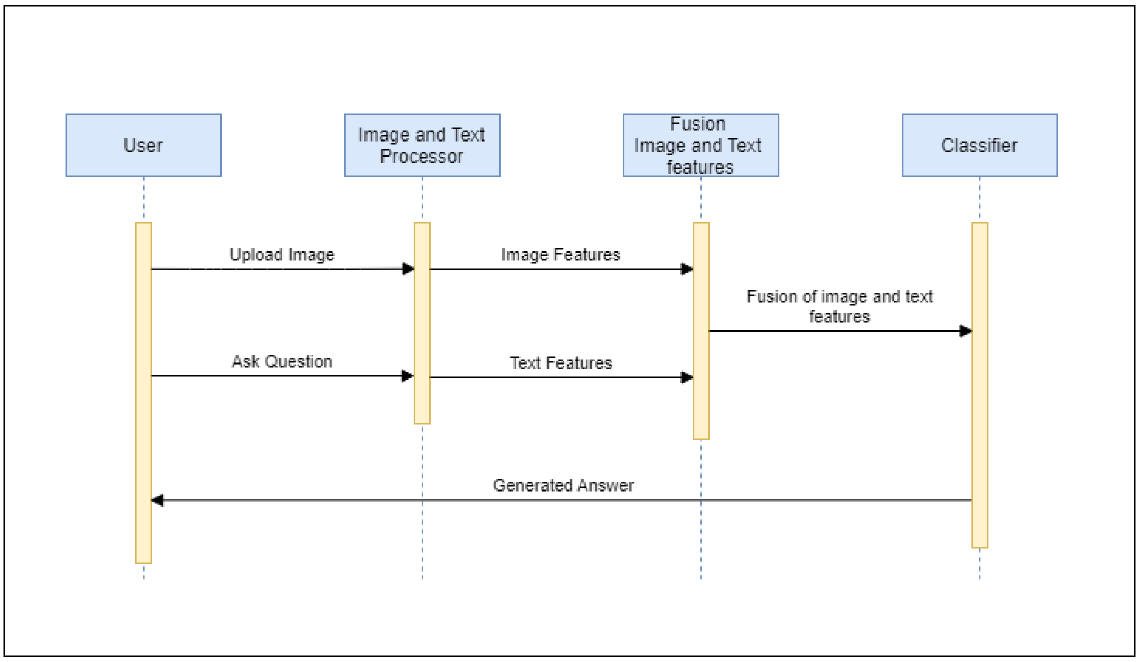


Figure 4.4: Sequence Diagram

4.3.2 Activity Diagram

Activity diagrams are used to represent flow of different activities in the system. It shows the flow of control and sequencing among different possible activities. Like use case, activity diagram also capture the user's perspective of the system. Activity diagrams do not explore system design or system's processing logic, instead focuses on flow of activities that a user of the system can experience. Activity diagram are mostly used for work flow modelling in Web Services and SOA (Service-Oriented Architecture) applications. Activity diagrams can be considered as refinement over Use case diagrams, where the scenario is captured more precisely.

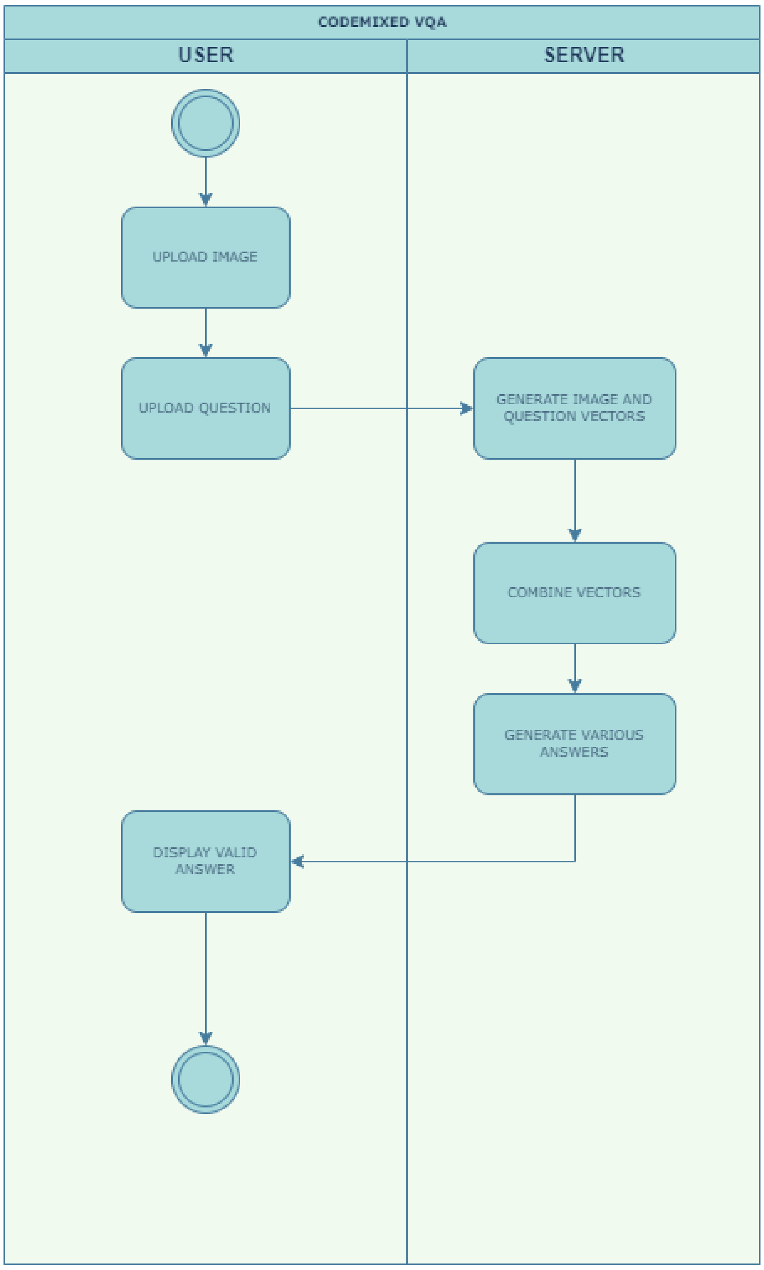


Figure 4.5: Activity Diagram

4.4 Dataset Design Pipeline

This Diagram illustrates the steps we took to create a valid, coherent and usable corpus. Istead of creating a dataset from scatch, we borrowed the images and English questions from the VQA dataset. We then translated the questions to marathi. These translated questions were artificially code mixed by employing a code mix language generator. Then these code mixed questions were transliterated to roman script.

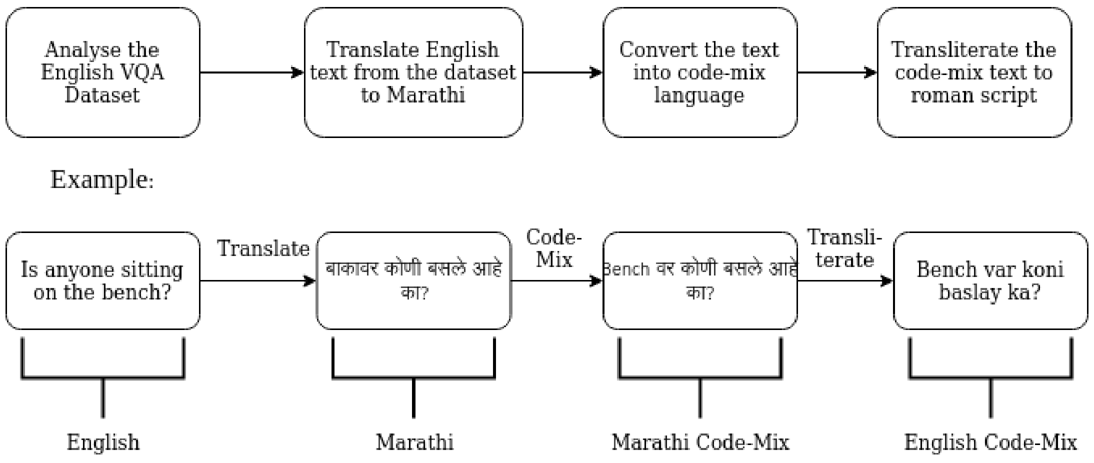

Analyse the English VQA Dataset
Translate English text from the dataset to Marathi
Convert the text into code-mix language
Transliterate the code-mix text to roman script
Example:
Is anyone sitting on the bench?
Translate
बाकावर कोणी बसले आहे का?
Code-Mix
Bench वर कोणी बसले आहे का?
Transli-terate
Bench var koni baslay ka?
English
Marathi
Marathi Code-Mix
English Code-Mix

Figure 4.6: Steps for creating the Dataset

Chapter 5

PROJECT PLAN

5.1 Project schedule

5.1.1 Project Task Set

The database creation was the first task which we started around mid of November. We collected the MSCOCO English Codemix Dataset for our further work. Data collection and pre-processing was done till mid January.Simultaeously we started learning about various Visual Question Answering and Codemix processing models which we used for our model creation. Our Main Task Set was divided into 2 parts- Dataset Creation And Model Building.

We started building the Codemix Dataset around mid January and on the other hand, we researched about LSTM and CVV Model concepts which we used for building our VQA model.

For Dataset Creation, we studied about different concepts regarding English to Marathi Translation, Part Of Speech Tagging for Marathi sentences using python, Word to Word Parallel Alignment for English And Marathi Sentences. This all work was done during mid january and February. We started building Codemix VQA Model in february and completed the training phase in mid of february. We tried different combinations and came up with attributes required for the Model such as filter size, stride, dimension of input image etc.

We tried different estimators and optimizers to increase accuracy of the model. We the tweaked our model to train it on larger dataset to increase the accuracy. We added record analysis with different criteria i.e daily,weekly and monthly analysis in mid of march. This way the project was completed till the march as per the project plan.

5.1.2 Task Network

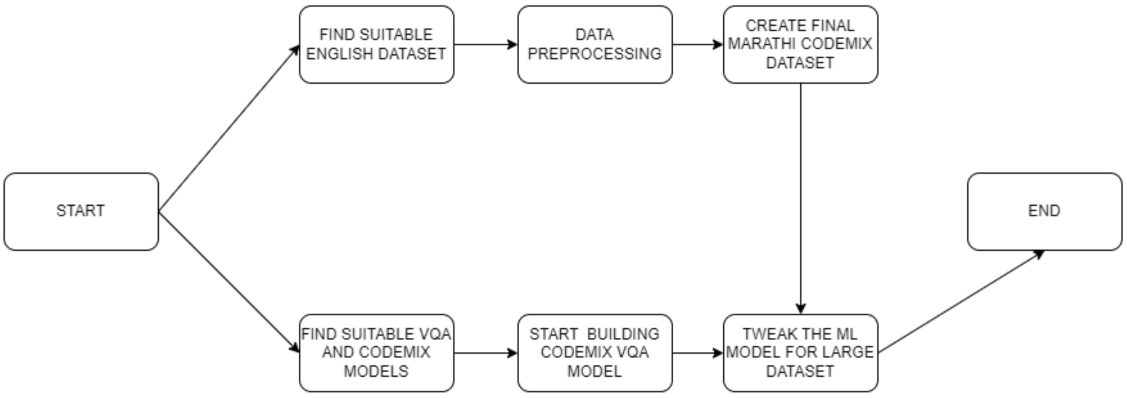
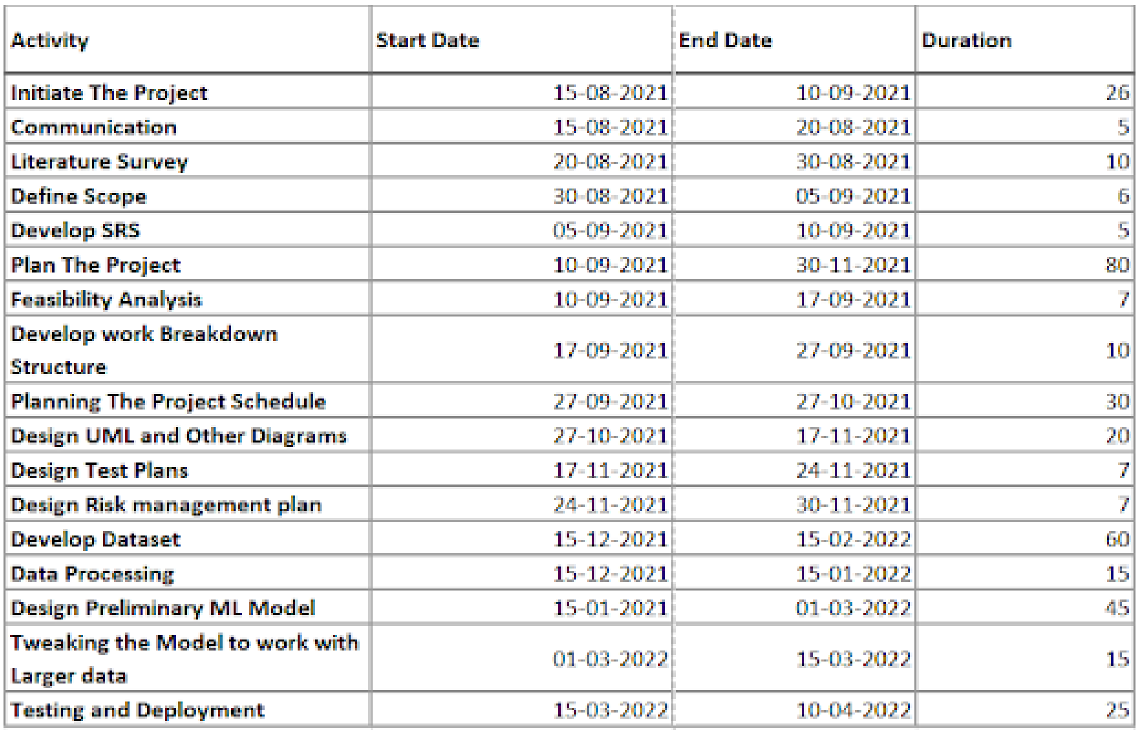

START
FIND SUITABLE ENGLISH DATASET
DATA PREPROCESSING
CREATE FINAL MARATHI CODEMIX DATASET
FIND SUITABLE VQA AND CODEMIX MODELS
START BUILDING CODEMIX VQA MODEL
TWEAK THE ML MODEL FOR LARGE DATASET
END

Figure 5.1: Task Network

5.1.3 Timeline Chart



Activity	Start Date	End Date	Duration
Initiate The Project	15-08-2021	10-09-2021	26
Communication	15-08-2021	20-08-2021	5
Literature Survey	20-08-2021	30-08-2021	10
Define Scope	30-08-2021	05-09-2021	6
Develop SRS	05-09-2021	10-09-2021	5
Plan The Project	10-09-2021	30-11-2021	80
Feasibility Analysis	10-09-2021	17-09-2021	7
Develop work Breakdown Structure	17-09-2021	27-09-2021	10
Planning The Project Schedule	27-09-2021	27-10-2021	30
Design UML and Other Diagrams	27-10-2021	17-11-2021	20
Design Test Plans	17-11-2021	24-11-2021	7
Design Risk management plan	24-11-2021	30-11-2021	7
Develop Dataset	15-12-2021	15-02-2022	60
Data Processing	15-12-2021	15-01-2022	15
Design Preliminary ML Model	15-01-2021	01-03-2022	45
Tweaking the Model to work with Larger data	01-03-2022	15-03-2022	15
Testing and Deployment	15-03-2022	10-04-2022	25

Figure 5.2: Timeline Chart

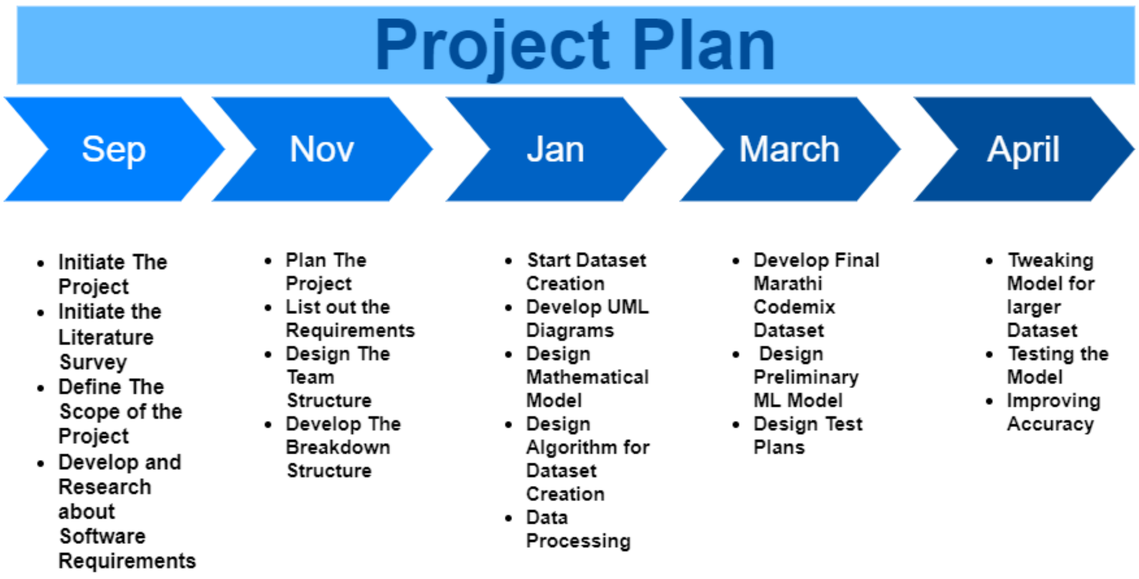
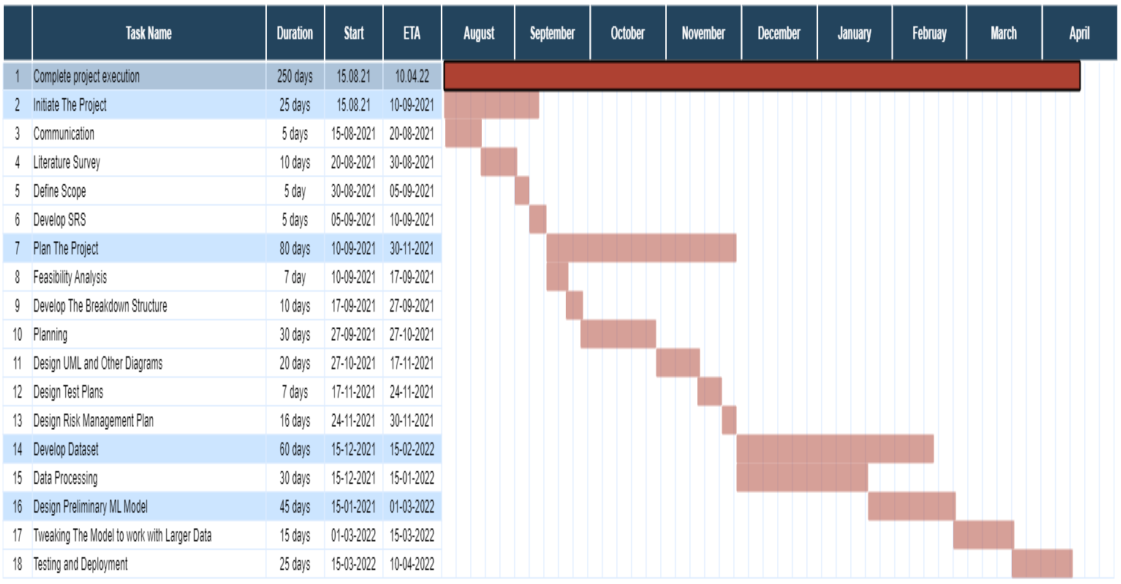

Project Plan
Sep
Nov
Jan
March
April
Initiate The Project
Initiate the Literature Survey
Define The Scope of the Project
Develop and Research about Software Requirements
Plan The Project
List out the Requirements
Design The Team Structure
Develop The Breakdown Structure
Start Dataset Creation
Develop UML Diagrams
Design Mathematical Model
Design Algorithm for Dataset Creation
Data Processing
Develop Final Marathi Codemix Dataset
Design Preliminary ML Model
Design Test Plans
Tweaking Model for larger Dataset
Testing the Model
Improving Accuracy

Figure 5.3: Project Plan



	Task Name	Duration	Start	ETA
1	Complete project execution	250 days	15.08.21	10.04.22
2	Initiate The Project	25 days	15.08.21	10-09-2021
3	Communication	5 days	15-08-2021	20-08-2021
4	Literature Survey	10 days	20-08-2021	30-08-2021
5	Define Scope	5 day	30-08-2021	05-09-2021
6	Develop SRS	5 days	05-09-2021	10-09-2021
7	Plan The Project	80 days	10-09-2021	30-11-2021
8	Feasibility Analysis	7 day	10-09-2021	17-09-2021
9	Develop The Breakdown Structure	10 days	17-09-2021	27-09-2021
10	Planning	30 days	27-09-2021	27-10-2021
11	Design UML and Other Diagrams	20 days	27-10-2021	17-11-2021
12	Design Test Plans	7 days	17-11-2021	24-11-2021
13	Design Risk Management Plan	16 days	24-11-2021	30-11-2021
14	Develop Dataset	60 days	15-12-2021	15-02-2022
15	Data Processing	30 days	15-12-2021	15-01-2022
16	Design Preliminary ML Model	45 days	15-01-2021	01-03-2022
17	Tweaking The Model to work with Larger Data	15 days	01-03-2022	15-03-2022
18	Testing and Deployment	25 days	15-03-2022	10-04-2022

Figure 5.4: Timeline Chart

5.2 Requirement Analysis

The main goal of our project is to provide a Visual Question Answering Model for Marathi Codemix sentences which is a first of its kind Model. This VQA model has many applications and can be used in many systems which are dependent on local languages. Our problem can be defined as, Using Machine Learning to predict the output of Codemix Question asked to the model based on the input image . Software requirement is a functional or non-functional need to be implemented in the system. Functional means providing particular service to the user. The main functional requirement of our system is to give the required and accurate answer to the user based on the image and question asked. The Functional requirements are concerned with the functionality of the system. Regarding the no functional requirements of our model, the main functionality is the accuracy of the dataset created, the codemix dataset must be accurate which will directly increase the accuracy of the model.

5.3 Team Orgainization

As we were in a group of 4, the modules were to be divided amongst all team members equally. The members needed tasks to be distributed to them that they were capable of handling. Therefore, the team can be said to have followed the organic structure. The tasks were divided accordingly and all the people reported to a single member and all the changes were updated on one machine. The Main 2 Modules of the Project were divided in the team of 2 each, 2 people handled the Dataset Creation Part and 2 People handled the Model Building Part.

5.4 Risk Assessment

Risk is an expectation of loss, a potential problem that may or may not occur in the future.It is generally caused due to lack of information, control or time. A possibility of suffering from loss in software development process is called a software risk. Loss can be anything, increase in production cost, development of poor quality software, not being able to complete the project on time. Software risk exists because the future is uncertain and there are many known and unknown things that cannot be incorporated in the project plan

5.4.1 Market Risk

The Model is designed to provide the a Marathi-English Codemix VQA system which can answer the question asked accurately based on the Image. There are many Hindi-English Codemix VQA system available in market, so therefore our model have a accuracy constraint that it should more accurate than the other models.

5.4.2 Schedule Risk

Time required to train a Machine Learning model is unpredictable and a complex task. Every layer has to be trained properly else it might result in false results.

5.4.3 Technical Risk

To gather as many images with the relevant question is complex task.Also Transforming that dataset in codemix language is tricky .But we did all the integration in the right way to create the final dataset.

5.4.4 Operational Risk

Insufficient Resources: Datasets and platforms available are insufficient to achieve maximum accuracy. Hardware constraints for Model training also decreased the chance to get the maximum accuracy.

Chapter 6

PROJECT IMPLEMENTATION

6.0.1 Details Of Modules

Dataset Creation

To make a robust and flexible model, it is important to train it on a vast and diverse dataset. Since we are only a team of 4, creating a dataset manually was out of the question. Therefore, we implemented a novel, automated dataset creation technique.
Briefly, we tried to convert a pre-existing, gold standard VQA dataset in English Language, into a Marathi+English Codemix dataset. This is a challenging task because a lot of steps need to be followed, in a precise order to achieve the goal. Furthermore, at every step, there is a chance of error being introduced, which as the algorithm proceeds will multiply, resulting in unnatural and practically useless data. Hence to tackle this, strict survelliance and error detection and correction needs to be applied at every step of the way.
The algorithm created and employed by us is elaborated below. Every step on the way will be explained in great detail further in this report.

Explanation of Algorithm 1

We need to convert every question in the English VQA dataset to Marathi+English Codemix Dataset. This endeavor requires first the translation of every English sentence into Marathi. For this, we use Google Translate API, which among all the possible translation tools has

Algorithm 1 High Overview of Dataset Creation

```
Require: Load English VQA Dataset
 1: $n \leftarrow 1$ question + answer
 2: $N \leftarrow n$
 3: while $N \neq empty$ do
 4:     $t \leftarrow translate\ n$
 5:     if $t$ is valid then
 6:         $c \leftarrow codemix\ t$
 7:         if $c$ is valid then
 8:             $tr \leftarrow transliterate\ c$
 9:             $T \leftarrow tr$
10:         end if
11:     end if
12: end while
```

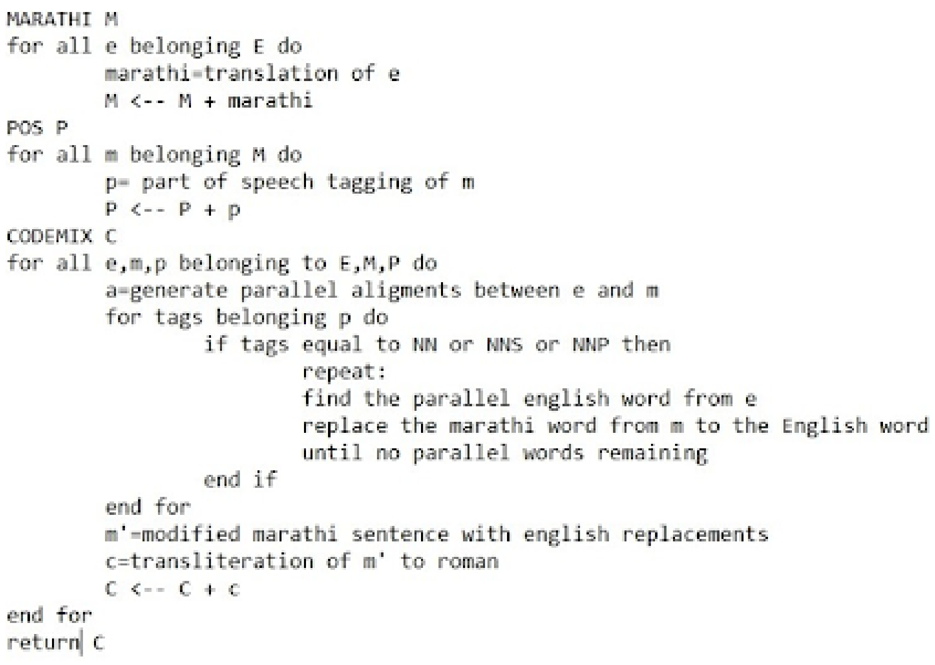
the best accuracy and reliability. Once ensured that the translated sentence is up-to the mark, a code mix sentence, consisting of words in Roman and Devanagari script is generated.This feat is achieved by word to word alignment from both the English and Marathi sentences. Fastalign and Simalign are the two tools we used for this step. Based on this codemix sentence, we get our final form ie. the Devnagari words are transliterated to Roman script using Aksharmukha API. Thus, one sentence of a large and extensive dataset is created. We have created this pipeline to cater to the process of creating almost any codemix dataset.

Model Training

6.0.2 Algorithm Details

Dataset Algorithm (or Implementation)

The popular VQA dataset released by Antol et al. (2015) contains images, with their corresponding questions (in English) and answers (in English). This is a challenging large scale dataset for the VQA task. So on the base of this dataset, we created the first marathi-english codemix dataset using extensive data cleaning and transformation The MSCOCO dataset contains 250000 english questions and the related answers in english with 80000 images. First step in creating synthetic codemix questions is to translate the english questions into marathi questions. For this task, we used official google translate api which gave us the accurate english to marathi translation.

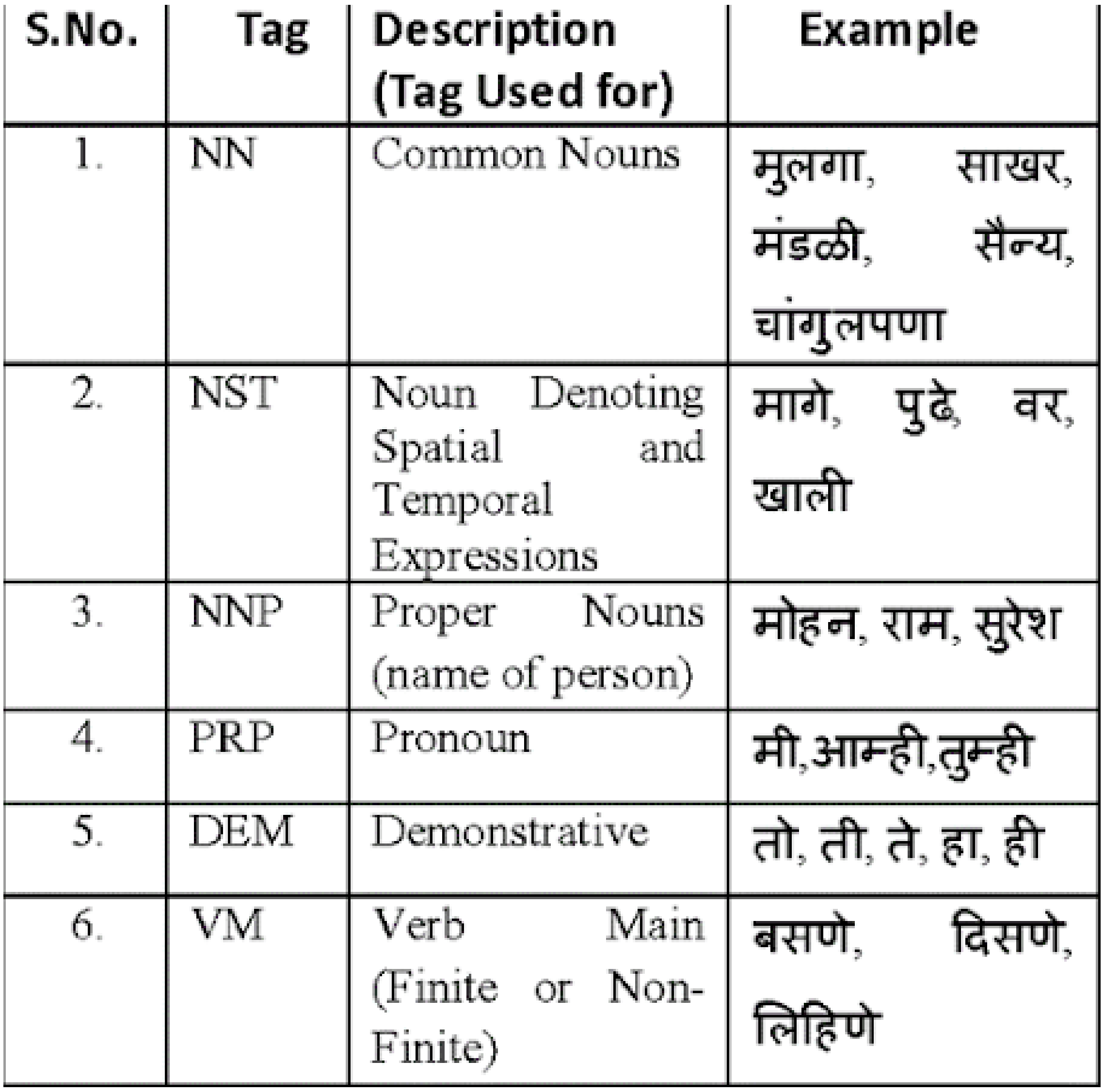


```
MARATHI M
for all e belonging E do
        marathi=translation of e
        M <-- M + marathi
POS P
for all m belonging M do
        p= part of speech tagging of m
        P <-- P + p
CODEMIX C
for all e,m,p belonging to E,M,P do
        a=generate parallel aligments between e and m
        for tags belonging p do
                if tags equal to NN or NNS or NNP then
                        repeat:
                        find the parallel english word from e
                        replace the marathi word from m to the English word
                        until no parallel words remaining
                end if
        end for
        m'=modified marathi sentence with english replacements
        c=transliteration of m' to roman
        C <-- C + c
end for
return C
```

Figure 6.1: Detailed Algorithm for dataset creation

- POS Tagging:

  Part-of-speech (POS) tagging is a popular Natural Language Processing process which refers to categorizing words in a text (corpus) in correspondence with a particular part of speech, depending on the definition of the word and its context. After Translating the English sentences to the Marathi Sentences, we POS Tag the Marathi sentences to get the Nouns and adjectives from the Marathi Sentences. For this process we used the nltk library and we built a basic pos tagger for the marathi language.



S.No.	Tag	Description (Tag Used for)	Example
1.	NN	Common Nouns	मुलगा, साखर, मंडळी, सैन्य, चांगुलपणा
2.	NST	Noun Denoting Spatial and Temporal Expressions	मागे, पुढे, वर, खाली
3.	NNP	Proper Nouns (name of person)	मोहन, राम, सुरेश
4.	PRP	Pronoun	मी,आम्ही,तुम्ही
5.	DEM	Demonstrative	तो, ती, ते, हा, ही
6.	VM	Verb Main (Finite or Non-Finite)	बसणे, दिसणे, लिहिणे

Figure 6.2: POS Tagging Marathi

- Parallel Word Alignment:

  After the POS Tagging, we align the English and Marathi Sentences Parallelly and get the Alignments between them using SimAlign. These alignments help us to identify the word's position in both the sentences having the same meaning.

  After getting the alignments, we use the POS tagging results and select out the Nouns and adjectives from the Marathi sentences and we place them in the English sentences replacing the English nouns and adjectives.

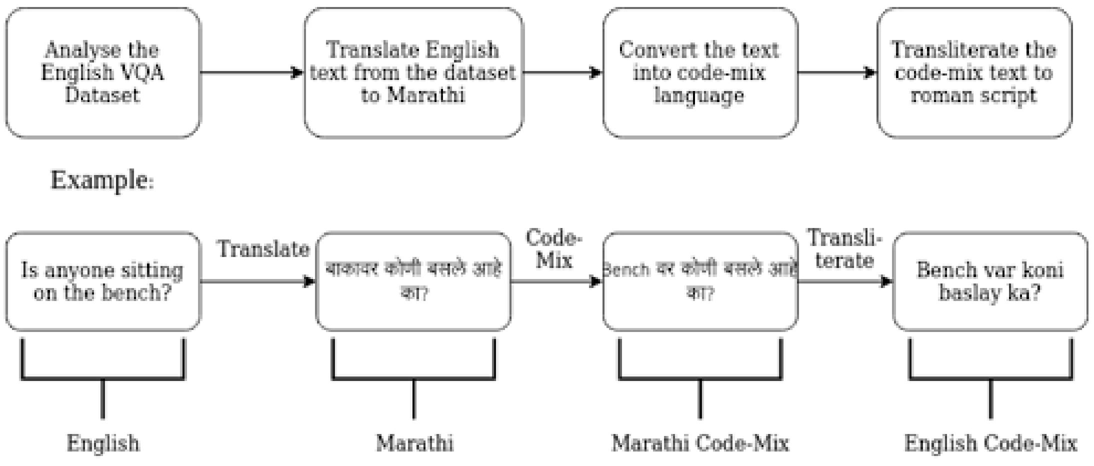

Analyse the English VQA Dataset
Translate English text from the dataset to Marathi
Convert the text into code-mix language
Transliterate the code-mix text to roman script
Example:
Is anyone sitting on the bench?
Translate
बाकावर कोणी बसले आहे का?
Code-Mix
Bench वर कोणी बसले आहे का?
Transli-terate
Bench var koni baslay ka?
English
Marathi
Marathi Code-Mix
English Code-Mix

Figure 6.3: Example Pipeline

- Transliteration: Transliteration is the process of transferring a word from the alphabet of one language to another. Transliteration helps people pronounce words and names in foreign languages.

  After getting the question in a mixed script we transliterate it in devanagari script using Aksharamukha Library which gives us our final Marathi English Codemix Question.

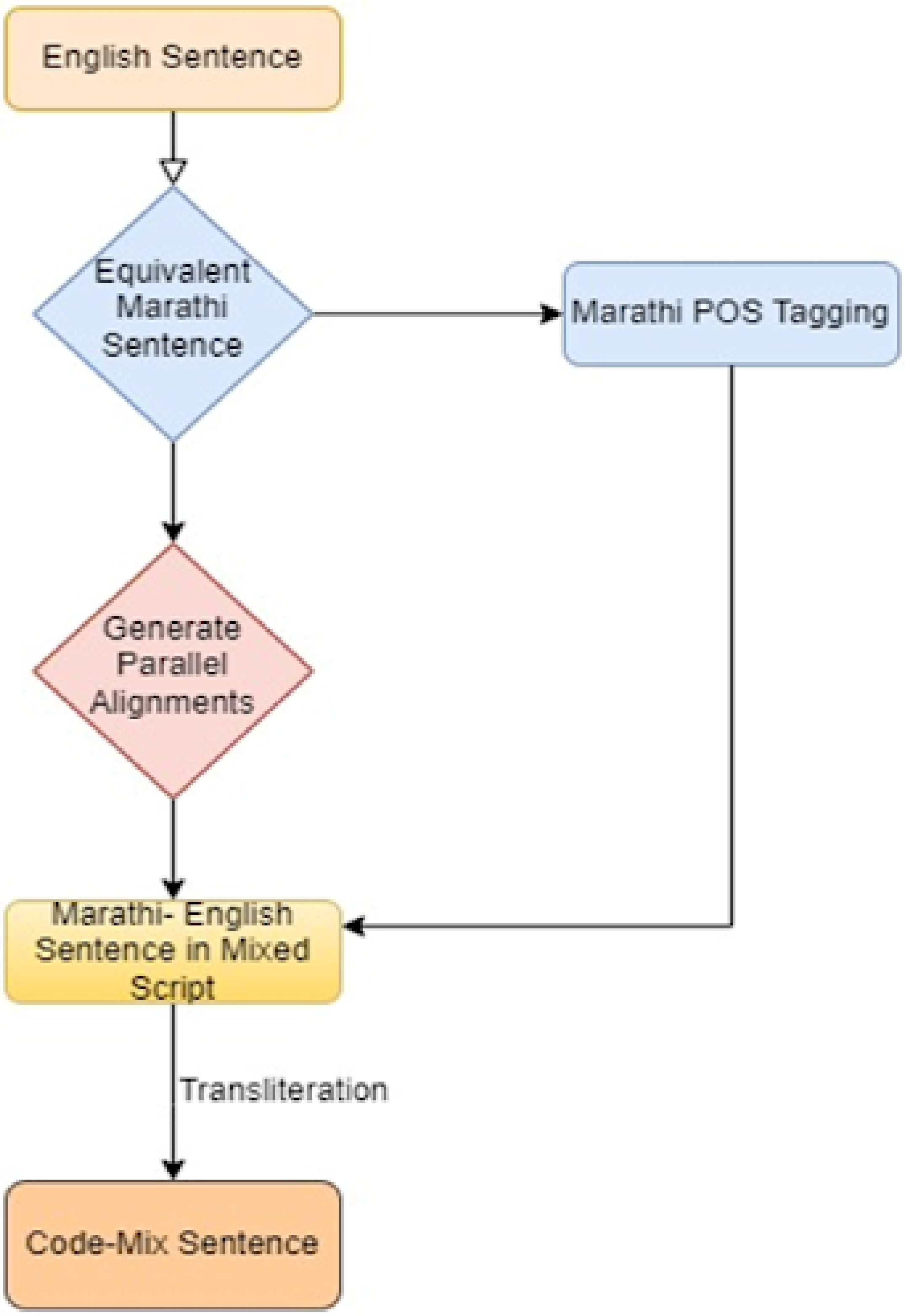

English Sentence
Equivalent Marathi Sentence
Marathi POS Tagging
Generate Parallel Alignments
Marathi- English Sentence in Mixed Script
Transliteration
Code-Mix Sentence

Figure 6.4: Work Flow

Model Devlopment

We developed 2 channels Vision(image)+question(language) with softmax over k possible output.K value represents the top K answers among the total answer dictionary.

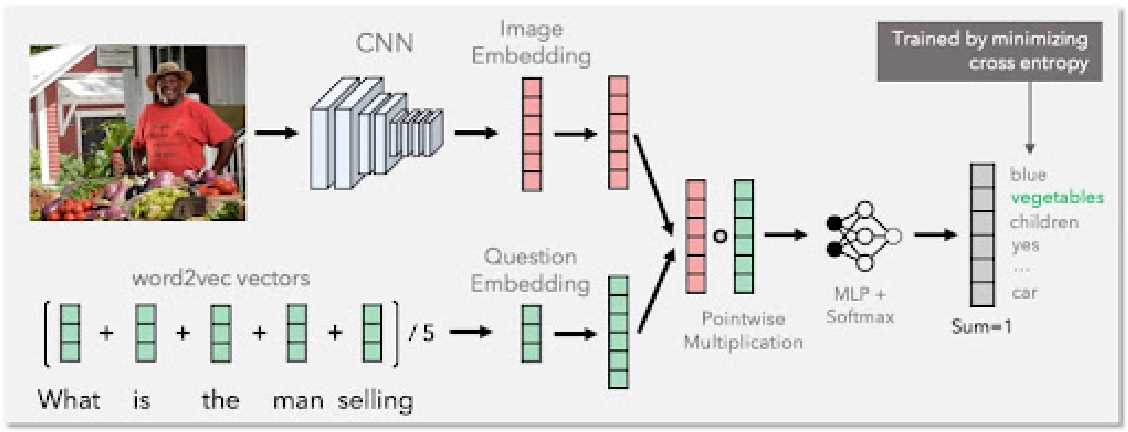


Figure 6.5: Work Flow

- VGG16:

  VGG16 is a convolution neural net (CNN) architecture. It is considered to be one of the excellent vision model architecture till date. Most unique thing about VGG16 is that instead of having a large number of hyper-parameter they focused on having convolution layers of 3x3 filter with a stride 1 and always used same padding and maxpool layer of 2x2 filter of stride 2. It follows this arrangement of convolution and max pool layers consistently throughout the whole architecture. In the end it has 2 FC (fully connected layers) followed by a softmax for output. The 16 in VGG16 refers to it has 16 layers that have weights. This network is a pretty large network and it has about 138 million (approx.) parameters.

- Question Channel:

  The question asked by the user was in textual formal which was human-understandable but data must be in system understandable format to feed for the model

- Word embedding:

  – Bag-of-words:

    A bag of words is a representation of text that describes the occurrence of words within a document. We just keep track of word counts and disregard the grammatical details and the word order. A Dictionary of words was created by iterating over each question. During the encoding phase of the question, each word of the question was encoded with its corresponding dictionary index.

- Word2Vec:

  word2vec is a family of model architectures and optimizations that can be used to learn word embeddings from large datasets. Embeddings learned through word2vec have proven to be successful on a variety of downstream natural language processing tasks.Word2Vec created vectors of the words that were numerical representations of word features. These word features could comprise words that represent the context of the individual words present in our vocabulary. Each word of the corpus has a unique representation in vector space. Word vectors are positioned in the vector space such that words that share common contexts in the corpus are located close to one another in the space. Two different model architectures that can be used by Word2Vec to create the word embeddings are the Continuous Bag of Words (CBOW) model the Skip-Gram model.

- fastText: fastText is a word embedding method that is an extension of the word2vec model. Instead of learning vectors for words directly, fastText represents each word as an n-gram of characters. In FastText, each word is modelled by a sum of vectors, with each vector representing an n-gram. There is a high possibility that any word that is not seen during training may occur during testing time, in that cases, Bag-of-words and Word2Vec are not capable to obtained correct vector representation for word. This is the main advantage of the fastText over the 2 techniques.

  Feature extraction was an important step which is done by LSTM(Long short-term memory) for the question channel. Long Short Term Memory Network is an advanced RNN, a sequential network, that allows information to persist. It is capable of handling the vanishing gradient problem faced by RNN. An LSTM with one hidden layer is used to obtain embedding for the question. The standard approach to performing VQA is something like this,

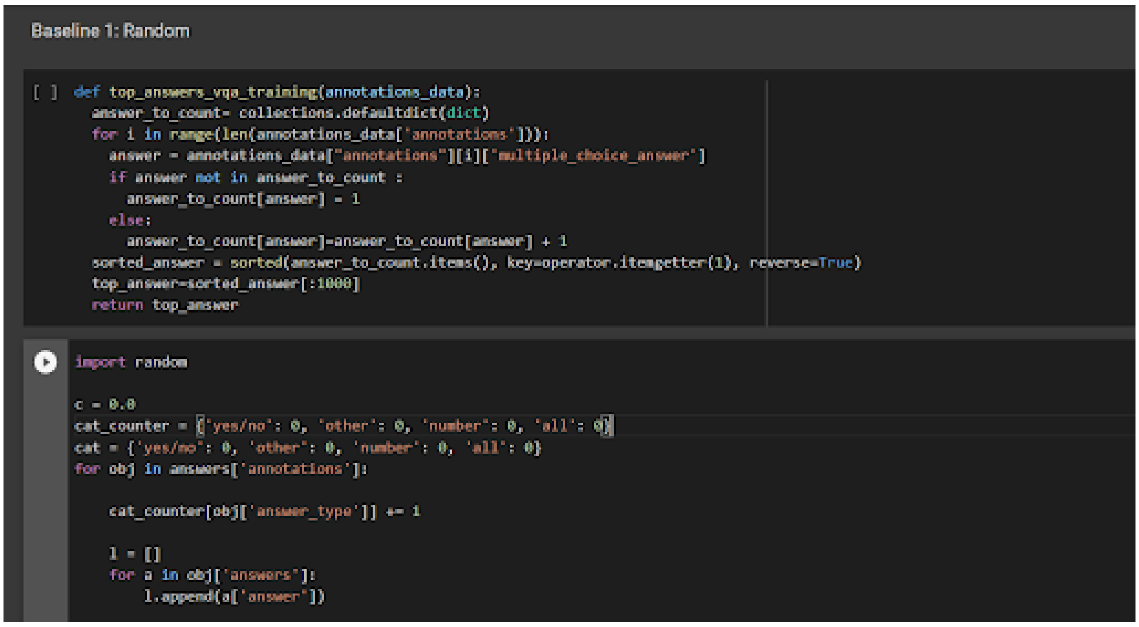
  * 1) Process the Image.

  - ∗ 2) Process the Question.
  - ∗ 3) Combine Features obtained from both Pipelines.
  - ∗ 4) Predict Accurate answer to the question.

- **B**aseline models(Algorithm):

  We implemented some baseline models on our novel codmix dataset which were set as the benchmark to deploy enlightened visual question answering(VQA) systems. The main goal is the creation of a robust dataset used to implement models.

  – Random: Top 1k answers from the training dataset were selected as finalized answers set and one among 1k answers was selected as an answer for each question.



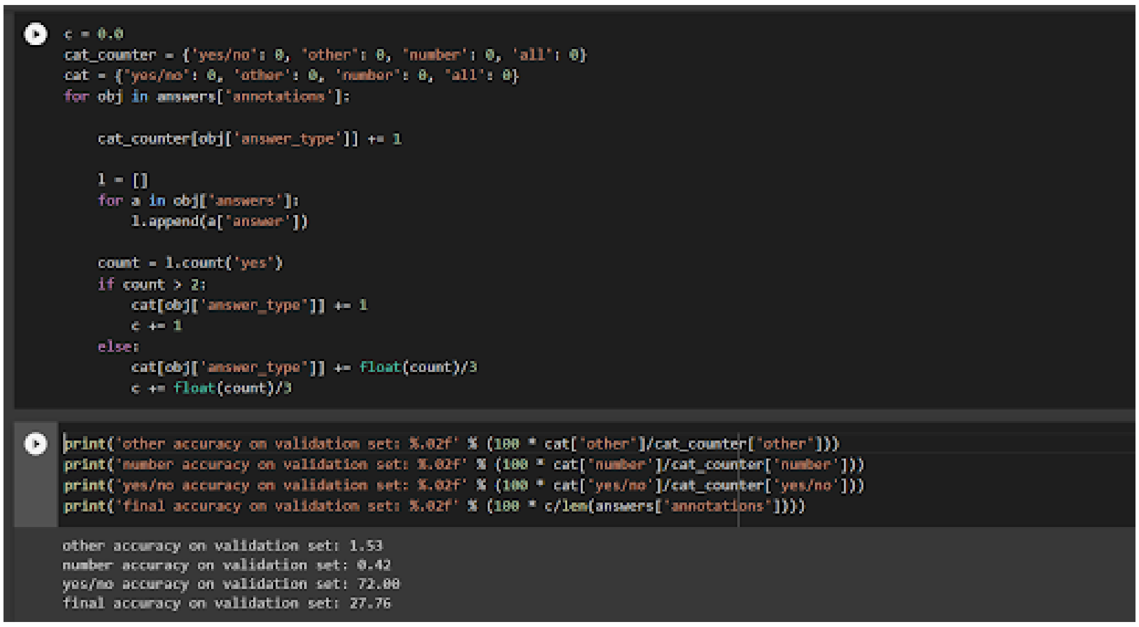
Baseline 1: Random

```
[ ] def top_answers_vqa_training(annotations_data):
      answer_to_count= collections.defaultdict(dict)
      for i in range(len(annotations_data['annotations'])):
        answer = annotations_data["annotations"][i]['multiple_choice_answer']
        if answer not in answer_to_count :
          answer_to_count[answer] = 1
        else:
          answer_to_count[answer]=answer_to_count[answer] + 1
      sorted_answer = sorted(answer_to_count.items(), key=operator.itemgetter(1), reverse=True)
      top_answer=sorted_answer[:1000]
      return top_answer
```

```
import random

c = 0.0
cat_counter = {'yes/no': 0, 'other': 0, 'number': 0, 'all': 0}
cat = {'yes/no': 0, 'other': 0, 'number': 0, 'all': 0}
for obj in answers['annotations']:

    cat_counter[obj['answer_type']] += 1

    l = []
    for a in obj['answers']:
        l.append(a['answer'])
```

Figure 6.6: Random Model

  – Prior (“yes”): “YES” is the most common answer among the training set which is selected as a default answer for each validate set question.



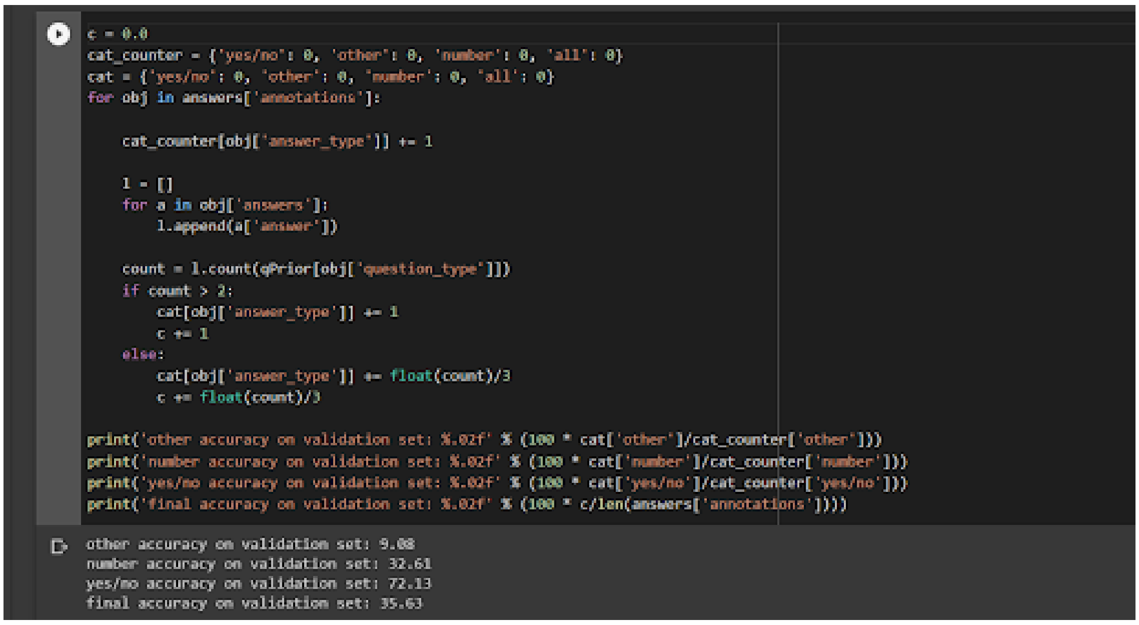
```
c = 0.0
cat_counter = {'yes/no': 0, 'other': 0, 'number': 0, 'all': 0}
cat = {'yes/no': 0, 'other': 0, 'number': 0, 'all': 0}
for obj in answers['annotations']:

    cat_counter[obj['answer_type']] += 1

    l = []
    for a in obj['answers']:
        l.append(a['answer'])

    count = l.count('yes')
    if count > 2:
        cat[obj['answer_type']] += 1
        c += 1
    else:
        cat[obj['answer_type']] += float(count)/3
        c += float(count)/3
```

```
print('other accuracy on validation set: %.02f' % (100 * cat['other']/cat_counter['other']))
print('number accuracy on validation set: %.02f' % (100 * cat['number']/cat_counter['number']))
print('yes/no accuracy on validation set: %.02f' % (100 * cat['yes/no']/cat_counter['yes/no']))
print('final accuracy on validation set: %.02f' % (100 * c/len(answers['annotations'])))
```

```
other accuracy on validation set: 1.53
number accuracy on validation set: 0.42
yes/no accuracy on validation set: 72.00
final accuracy on validation set: 27.76
```

Figure 6.7: Random Model

- Per Q-type prior: Define different categories of questions based on starting a few words from the question. For each question, the most frequent answer can be selected among all answers for that category



```
c = 0.0
cat_counter = {'yes/no': 0, 'other': 0, 'number': 0, 'all': 0}
cat = {'yes/no': 0, 'other': 0, 'number': 0, 'all': 0}
for obj in answers['annotations']:

    cat_counter[obj['answer_type']] += 1

    l = []
    for a in obj['answers']:
        l.append(a['answer'])

    count = l.count(qPrior[obj['question_type']])
    if count > 2:
        cat[obj['answer_type']] += 1
        c += 1
    else:
        cat[obj['answer_type']] += float(count)/3
        c += float(count)/3

print('other accuracy on validation set: %.02f' % (100 * cat['other']/cat_counter['other']))
print('number accuracy on validation set: %.02f' % (100 * cat['number']/cat_counter['number']))
print('yes/no accuracy on validation set: %.02f' % (100 * cat['yes/no']/cat_counter['yes/no']))
print('final accuracy on validation set: %.02f' % (100 * c/len(answers['annotations'])))
```

```
other accuracy on validation set: 9.08
number accuracy on validation set: 32.61
yes/no accuracy on validation set: 72.13
final accuracy on validation set: 35.63
```

Figure 6.8: Random Model

- **T**ransformer based models(Algorithm):

  We implemented some baseline models on our novel codmix dataset which were set as the benchmark to deploy enlightened visual question answering(VQA) systems. The main goal is the creation of a robust dataset used to implement models.

- VisualBERT:

  VisualBERT is a multi-modal vision and language model. It can be used for visual question answering, multiple choice, visual reasoning and region-to-phrase correspondence tasks. VisualBERT uses a BERT-like transformer to prepare embeddings for image-text pairs. Both the text and visual features are then projected to a latent space with identical dimension.

  To feed images to the model, each image is passed through a pre-trained object detector and the regions and the bounding boxes are extracted. The authors use the features generated after passing these regions through a pre-trained CNN like ResNet as visual embeddings. They also add absolute position embeddings, and feed the resulting sequence of vectors to a standard BERT model. The text input is concatenated in the front of the visual embeddings in the embedding layer, and is expected to be bound by [CLS] and a [SEP] tokens, as in BERT. The segment IDs must also be set appropriately for the textual and visual parts.

- Uniter Base: Joint image-text embedding is the bedrock for most Vision-and-Language (V+L) tasks, where multimodality inputs are jointly processed for visual and textual understanding. In this paper, we introduce UNITER, a UNiversal Image-TExt Representation, learned through large-scale pre-training over four image-text datasets (COCO, Visual Genome, Conceptual Captions, and SBU Captions), which can power heterogeneous downstream V+L tasks with joint multimodal embeddings. We design three pre-training tasks: Masked Language Modeling (MLM), ImageText Matching (ITM), and Masked Region Modeling (MRM, with three variants). Different from concurrent work on multimodal pre-training that apply joint random masking to both modalities, we use conditioned masking on pre-training tasks (i.e., masked language/region modeling is conditioned on full observation of image/text). Comprehensive analysis shows that conditioned masking yields better performance than unconditioned masking. We also conduct a thorough ablation study to find an optimal setting for the combination of pre-training tasks. Extensive experiments show that UNITER achieves new state of the art across six V+L tasks (over nine datasets), including Visual Question Answering, Image-Text Retrieval, Referring Expression Comprehension, Visual Commonsense Reasoning, Visual Entailment, and NLVR2

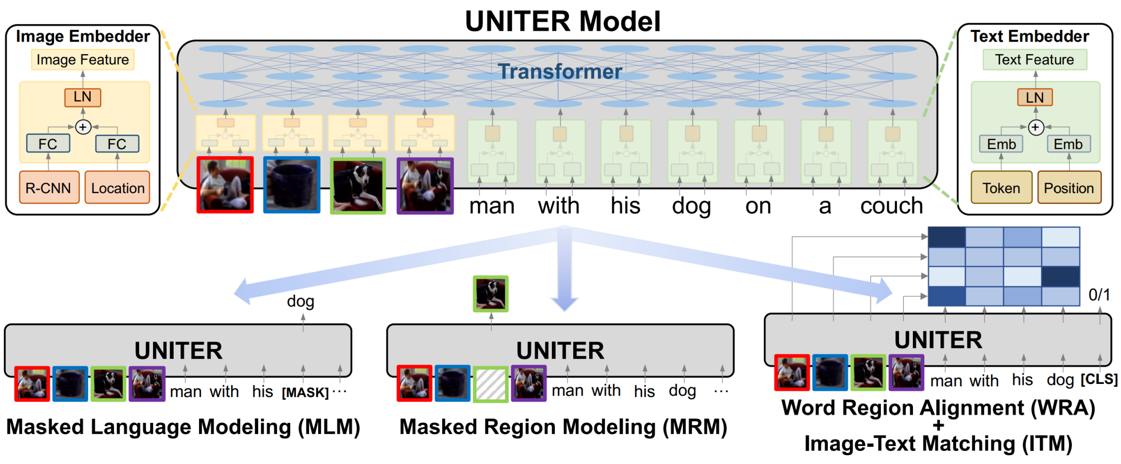


Figure 6.9: Random Model

– LXMERT: Vision-and-language reasoning requires an understanding of visual concepts, language semantics, and, most importantly, the alignment and relationships between these two modalities. We thus propose the LXMERT (Learning Cross-Modality Encoder Representations from Transformers) framework to learn these vision-and-language connections. In LXMERT, we build a large-scale Transformer model that consists of three encoders: an object relationship encoder, a language encoder, and a cross-modality encoder. Next, to endow our model with the capability of connecting vision and language semantics, we pre-train the model with large amounts of image-and-sentence pairs, via five diverse representative pre-training tasks: masked language modeling, masked object prediction (feature regression and label classification), cross-modality matching, and image question answering. These tasks help in learning both intra-modality and cross-modality relationships. After fine-tuning from our pre-trained parameters, our model achieves the state-of-the-art results on two visual question answering datasets (i.e., VQA and GQA). We also show the generalizability of our pre-trained cross-modality model by adapting it to a challenging visual-reasoning task, NLVR2, and improve the previous best result by 22

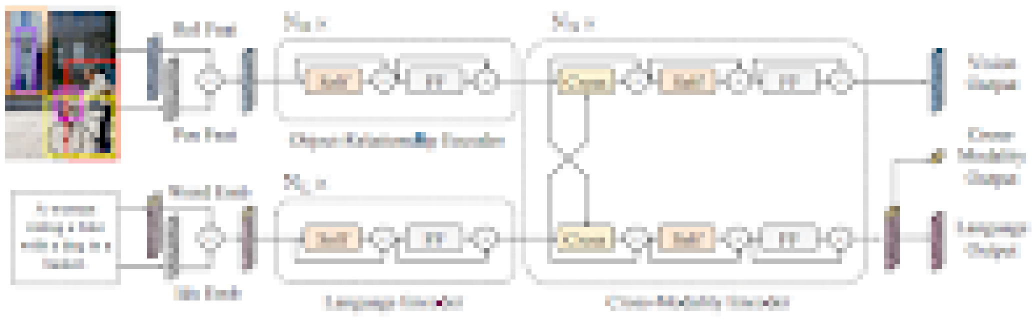


Figure 6.10: LXMERT

- **E**valuation Metrics

Over the years, several metrics have been suggested to determine the quality of codemixed text. The ones that we have calculated for our dataset are the following-

  - **Degree of Code-mixing (DCM):** The code-mixing index (CMI) proposed by (Das andGambäck, 2014) measures the degree of code-mixing in a text based on language tags of the participating tokens. The objective of DCM follows CMI with an exception of explicit to- ken language identification to measure the degree of code-mixing in the text. The DCM score can vary between 0 to 10. A DCM score of 0 corresponds to the monolingual sentence with no code-mixing, whereas the DCM score of 10 suggests a high degree of code-mixing. The formula for CMI is shown in figure 7.3.

$$\text{CMI} = \begin{cases} 100 \times [1 - \frac{max\{w_i\}}{n-u}] & : n > u \\ 0 & : n = u \end{cases}$$

Figure 6.11: Calculation of CMI

To calculate our CMI score, we randomly selected 1500 questions from the English VQA dataset. Then all four of use, manually generated the code-mix questions for all 1500 selected questions. Based on this data, we applied the above mentioned formula. Our results are shown in table 7.1.

- **Readability (RA):** RA score can vary between 0 to 10. A completely unreadable sentence due to many spelling mistakes, no sentence structuring, or meaning yields a RA score of 0. A RA score of 10 suggests a highly readable sentence with clear semantics and easy-to-read words
  To calculate our RA score, we randomly selected 1500 questions from the English VQA dataset. Then all four of use, manually generated the code-mix questions for all 1500 selected questions. Based on this data, we calculated the RA score. Our results are shown in table 7.1.

- **Bilingual Evaluation Understudy Score(BLEU):** BLEU, or the Bilingual Evaluation Understudy, is a score for comparing a candidate translation of text to one or more reference translations. Although developed for translation, it can be used to evaluate text generated for a suite of natural language processing tasks. A perfect match results in a score of 1.0, whereas a perfect mismatch results in a score of 0.0.

```python
# two references for one document
from nltk.translate.bleu_score import corpus_bleu
references = [[['this', 'is', 'a', 'test'], ['this', 'is' 'test']]]
candidates = [['this', 'is', 'a', 'test']]
score = corpus_bleu(references, candidates)
print(score)
```

Figure 6.12: Calculating BLEU score using NLTK

The Python Natural Language Toolkit library, or NLTK, provides an implementation of the BLEU score that you can use to evaluate your generated text against a reference. It's example is shown in figure 7.4. To calculate our BLEU score, we randomly selected 1500 questions from the English VQA dataset. Then all four of use, manually generated the code-mix questions for all 1500 selected questions. On this data, we used the NLTK module.

- Transformer based model score

The transformer based models were trained for multiple choice questions. Where each question has multiple options. Every option is assigned a score to it. If the option is the correct answer then 1 is the score assigned to the option. If the option is partially correct answer then it is assigned as 0.5 score and if the answer is incorrect then 0 is the score. After predicting the answer using the model our evaluator adds the corresponding score of the predicted answer and takes the average of scores for all the answers.

Chapter 7

RESULTS

7.0.1 Dataset Results

To understand if a dataset is up-to a certain standard, a fair bit of analysis needs to be done. Further is set of predefined metrics needs to be calculated. In this section, the results of the dataset analysis are elaborated.

Word Cloud

Word clouds (also known as text clouds or tag clouds) work in a simple way: the more a specific word appears in a source of textual data (such as a speech, blog post, or database), the bigger and bolder it appears in the word cloud. A word cloud is a collection, or cluster, of words depicted in different sizes. The bigger and bolder the word appears, the more often it's mentioned within a given text and the more important it is. Also known as tag clouds or text clouds, these are ideal ways to pull out the most pertinent parts of textual data.
Hence, for our dataset, we can use a simple word cloud to understand the distribution of the most frequent words. From figure 7.1, it is clear that the Marathi questioning words ”Ka”, ”Kay”, ”Kuthe” etc appear prominently. This is no fluke as the part of speech tagging and transliteration techniques employed by us transliterated Marathi verbs.

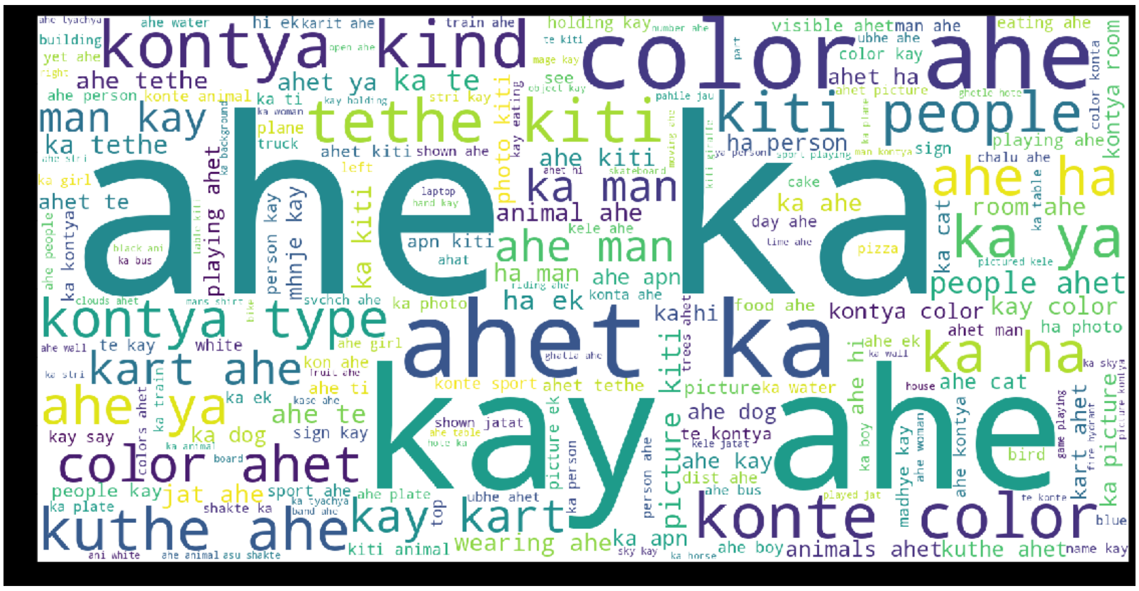


Figure 7.1: Word Cloud of all 250k Questions

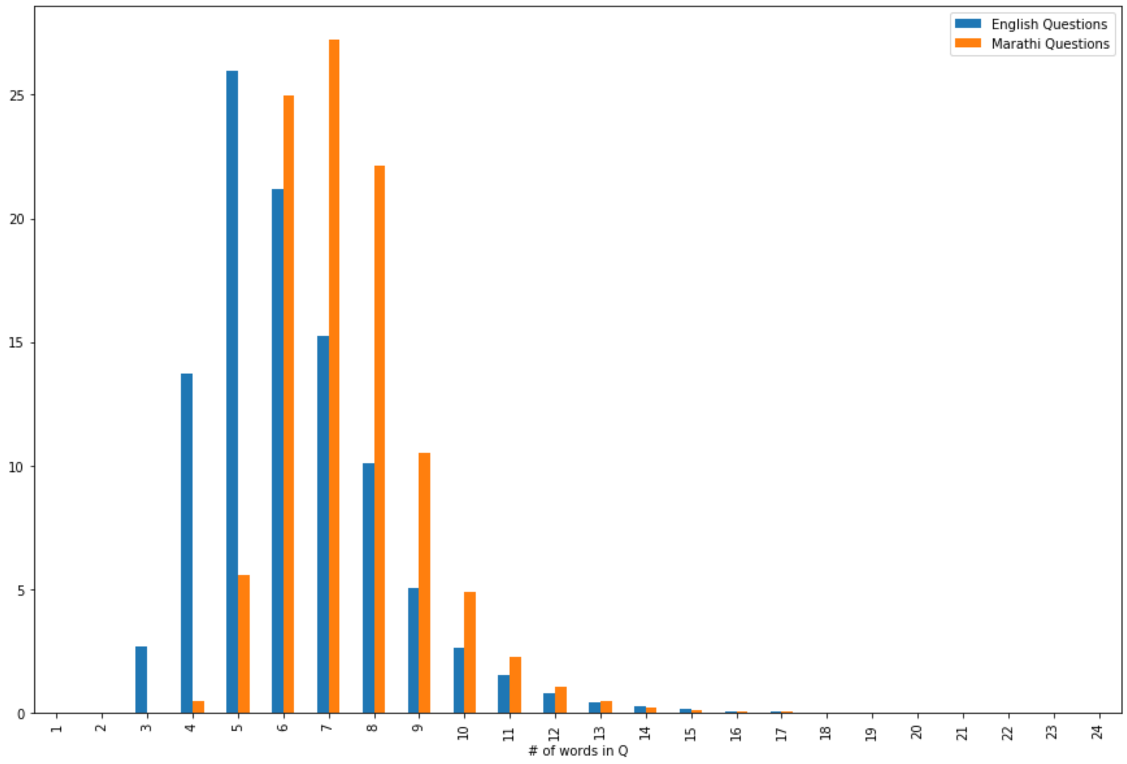


Figure 7.2: Number of words in Questions

Question Length Distribution

It is pertinent to know how the words per question distribution changes as we switch from English to Marathi + English Codemix. As is evident from figure 7.2, the number of words per question increases in Code mix Question with more than 25% questions having 7 words in them. This change can be attributed to the fact that Marathi language has joint words which when converted to Roman script are broken down into multiple words thus increasing the word count. Furthermore, since our dataset shows this fundamental property of the languages it is trying to synthesis, it can be considered proof of the dataset's quality.

Our Results for dataset

Table 7.1 shows the results of the various metrics detailed above. Our dataset achieves an excellent Degree of Code-mixing score and a more than satisfactory Readability score. Although our dataset lacks in the BLEU score department, we think it can be partially attributed to the bias of our annotators

Degree of Code-mixing	7.411086817341725
Readability	6.091884339445364
BLEU	0.242

Our results for model training

Base Line Model	Accuracy
Random	0.14
Prior ("yes")	27.76
Per Q-type prior	35.63

APPENDIX A

7.1 Feasibility Analysis

7.1.1 Operational Feasibility

- The intended database is easier to understand and used which is in standard Json format.
- The intented system requires strong computation model to further work on this domain.

7.1.2 Technical Feasibility

- Most of the components used are open source and freely available to reuse.
- Created dataset is open to use.

7.1.3 Economic Feasibility

The financial resources to build this project are small and within the budget. Although the excessive training of model will be requiring high computation power which can be achieved by using GPUs.

7.2 Computational Complexity

7.2.1 Polynomial(P)

- The output is given by the system in xed polynomial time. Applications of polynomial type are rare.
- One such example is hashtable. The hashtable finds an index for a data to be inserted in a fixed amount of time because a hash function is used to find the index. So for finding an index for one or a hundred, the time is fixed.
- P is a set of problems that can be solved by a deterministic Turing machine in Polynomial time.
- Our application is not of type P because it does not give results in xed polynomial time

7.2.2 Non-Polynomial(NP)

- NP-Hard

  There are two subtypes of NP Problems: The system accepts input, but there is no guarantee that we will get the output. Such systems do not exist because no one will use the system if there is no guarantee the system works for any inputs. Hence our application is again not of NP-Hard type because we want to build a system that never fails and guarantees output. Example: Turing Machine Halting problem

- NP-Complete

  NP complete problems are problems whose status is unknown. No polynomial time algorithm has yet been discovered for any NP complete problem, nor has anybody yet been able to prove that no polynomial-time algorithm exists for any of them. Thus, currently systems for such problems can solve them in a nonpolynomial time.

  Example: Boolean Satisfiability Problem, Travelling Salesman problem. Our System will take the user's voice as input and convert it in text format and generate prescription at a nonpolynomial time. Thus, our problem does not belong to the

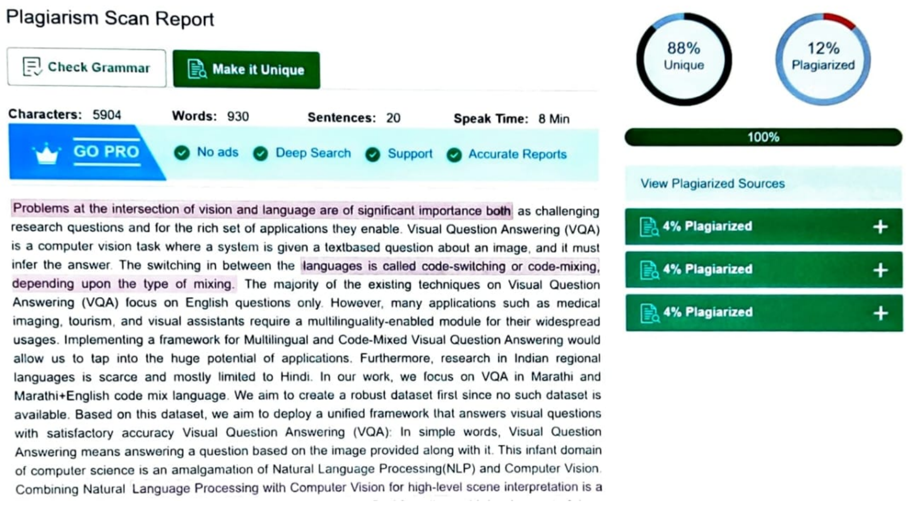
P-class of problems. Our system should guarantee a result, hence the problem, despite taking a non-polynomial time to complete, does not belong to the NP-hard class of problems either. As our system guarantees a result but executes and the solution can be verified in non-polynomial time, it is of NP-complete type. Hence, our problem statement is NP-complete

APPENDIX B

Deepak Gupta, Pabitra Lenka, Asif Ekbal, and Pushpak Bhattacharyya. 2020. A Unified Framework for Multilingual and Code-Mixed Visual Question Answering. In Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing, pages 900–913, Suzhou, China. Association for Computational Linguistics. [5]

In this paper, an effective deep learning framework for multilingual and code- mixed visual question answering is proposed. The proposed model is capable of predicting answers from the questions in Hindi, English or Code- mixed (Hinglish: Hindi-English) languages. The majority of the existing techniques on Visual Question Answering (VQA) focus on English questions only. However, many applications such as medical imaging, tourism, visual assistants require a multilinguality-enabled module for their widespread usages. As there is no available dataset in English-Hindi VQA, we firstly create Hindi and Code-mixed VQA datasets by exploiting the linguistic properties of these languages. They propose a robust technique capable of handling the multilingual and code-mixed question to provide the answer against the visual information (image). They perform extensive evaluation and ablation studies for English, Hindi and Code- mixed VQA. The evaluation shows that the proposed multilingual model achieves state-of-the-art performance in all these settings.

APPENDIX C



Characters: 5904 **Words:** 930 **Sentences:** 20 **Speak Time:** 8 Min

Problems at the intersection of vision and language are of significant importance both as challenging research questions and for the rich set of applications they enable. Visual Question Answering (VQA) is a computer vision task where a system is given a textbased question about an image, and it must infer the answer. The switching in between the languages is called code-switching or code-mixing, depending upon the type of mixing. The majority of the existing techniques on Visual Question Answering (VQA) focus on English questions only. However, many applications such as medical imaging, tourism, and visual assistants require a multilinguality-enabled module for their widespread usages. Implementing a framework for Multilingual and Code-Mixed Visual Question Answering would allow us to tap into the huge potential of applications. Furthermore, research in Indian regional languages is scarce and mostly limited to Hindi. In our work, we focus on VQA in Marathi and Marathi+English code mix language. We aim to create a robust dataset first since no such dataset is available. Based on this dataset, we aim to deploy a unified framework that answers visual questions with satisfactory accuracy Visual Question Answering (VQA): In simple words, Visual Question Answering means answering a question based on the image provided along with it. This infant domain of computer science is an amalgamation of Natural Language Processing(NLP) and Computer Vision. Combining Natural Language Processing with Computer Vision for high-level scene interpretation is a

Figure 7.3: Plagiarism Report

Bibliography

[1] Aishwarya Agrawal et al. “VQA: Visual Question Answering”. In: *Int. J. Comput. Vision* 123.1 (May 2017), pp. 4–31. ISSN: 0920-5691. DOI: 10.1007/s11263-016-0966-6. URL: https://doi.org/10.1007/s11263-016-0966-6.

[2] Stanislaw Antol et al. “VQA: Visual Question Answering”. In: *CoRR* abs/1505.00468 (2015). arXiv: 1505.00468. URL: http://arxiv.org/abs/1505.00468.

[3] Aditya Bohra et al. “A Dataset of Hindi-English Code-Mixed Social Media Text for Hate Speech Detection”. In: *Proceedings of the Second Workshop on Computational Modeling of People’s Opinions, Personality, and Emotions in Social Media*. New Orleans, Louisiana, USA: Association for Computational Linguistics, June 2018, pp. 36–41. DOI: 10.18653/v1/W18-1105. URL: https://aclanthology.org/W18-1105.

[4] Kan Chen et al. “Abc-cnn: An attention based convolutional neural network for visual question answering”. In: *arXiv preprint arXiv:1511.05960* (2015).

[5] Deepak Gupta et al. “A Unified Framework for Multilingual and Code-Mixed Visual Question Answering”. In: *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*. Suzhou, China: Association for Computational Linguistics, Dec. 2020, pp. 900–913. URL: https://aclanthology.org/2020.aacl-main.90.

[6] Didar Hossain and Kapil Bar. “A case study in code-mixing among Jahangirnagar University students”. In: *International Journal of English and Literature* 6.7 (2015), pp. 123–139.

[7] Kushal Kafle and Christopher Kanan. “Visual question answering: Datasets, algorithms, and future challenges”. In: *Computer Vision and Image Understanding* 163 (2017), pp. 3–20.

[8] Tsung-Yi Lin et al. “Microsoft COCO: Common Objects in Context”. In: *CoRR* abs/1405.0312 (2014). arXiv: `1405.0312`. URL: `http://arxiv.org/abs/1405.0312`.

[9] Bryan A. Plummer et al. “Flickr30k Entities: Collecting Region-to-Phrase Correspondences for Richer Image-to-Sentence Models”. In: *CoRR* abs/1505.04870 (2015). arXiv: `1505.04870`. URL: `http://arxiv.org/abs/1505.04870`.

[10] Sahil Swami et al. “A Corpus of English-Hindi Code-Mixed Tweets for Sarcasm Detection”. In: *ArXiv* abs/1805.11869 (2018).