School of Computing

FACULTY OF ENGINEERING

UNIVERSITY OF LEEDS

Deep Learning for under-resourced languages and dialects

Garima Vatsa

Submitted in accordance with the requirements for the degree of
MSc Advanced Computer Science (Data Analytics)

2018/2019

The candidate confirms that the following have been submitted*:*

Items	**Format**	**Recipient(s) and Date**
Deliverables 1, 2, 3	*Report*	*SSO (09/13/19)*
Deliverable 4	*Software codes or URL*	*Supervisor, assessor (09/13/19)*

Type of Project: Empirical Investigation

The candidate confirms that the work submitted is their own and the appropriate credit has been given where reference has been made to the work of others.

I understand that failure to attribute material which is obtained from another source may be considered as plagiarism.

(Signature of student) ____________________________

Summary

Over the past decade, linguistic computing for under-resourced languages has witnessed significant progress. This paper presents an analytical review of existing challenges and approaches in the field of machine translation for many spoken languages lacking digital text resources. We currently have numerous models available for well-resourced languages. However, building a natural language translation model for under-resourced language is challenging since huge textual data is the basis of Natural Language Processing. The aim of this research study is to explore methods to adapt or transfer word embedding models learnt from well-resourced languages, having extensive corpus available in digital format, to handle related under-resourced languages.

Marathi and Sanskrit have been considered for this project. Sanskrit and Marathi are ancient languages with more than 2000-year-old history. However, these languages lack the presence of digital text data. Both, Sanskrit and Marathi, are written in Devanagari script and also considered to be sibling languages. (Wiki, n.d.).

Concepts of data mining, machine learning, data mining, and information visualisation have been applied during the different phases of this project while following CRISP-DM methodology approach which is a standard research methodology for data mining project.

Acknowledgements

I have received a great deal of assistance throughout the realization of my master’s project. First of all, I would like to thank my supervisor, Dr Haiko Muller, for his invaluable support and encouragement. Dr Muller helped me with understanding the aim of this project and his constant feedback was crucial to the improvisation to achieve the objectives. His directions have been crucial for this project.

I would also like to express my gratitude towards my professors, colleagues, administrative staff of the School of Computing, and my family to help me get to this point. Your constant understanding is truly appreciated.

Table of Contents

List of Figures

List of Tables

List of Abbreviation

MT	Machine Translation
NLP	Natural Language Processing
LSA	Latent Semantic Analysis
Q1	Dataset for Marathi language
Q2	Dataset for Sanskrit language
TF-IDF	Term Frequency, Inverse Document frequency
RBMT	Rule-Based Machine Translation
IMT	Interlingual Machine Translation
TMT	Transfer-based Machine Translation
CBMT	Corpus-Based Machine Translation
SMT	Statistical Machine Translation
EBMT	Example-based Machine Translation
HMT	Hybrid Machine Translation
NMT	Neural Machine Translation
CBOW	Continuous Bag of Words
LTSM	Long short-term memory
GRU	Gated recurrent unit
RNN	Recurring neural network
BLUE	Bilingual evaluation understudy

Chapter 1
Introduction

This paper concerns and contributes to the research area of Text analytics and proposes the challenging task of training resources and linguistic models for under-resourced languages. The objective of this research is to apply deep learning to a text corpus of an under-resourced language to produce word embedding models that are universal in nature and can be learned from any language. Our proposed approach emphasizes that a transfer-learning approach can be used to share lexical and sentence level representations across closely-related languages into one target language. In this paper, we propose a universal neural machine translation approach focusing primarily on low resource languages to leverage the capabilities of NMT while overcoming the shortcoming. In chapter 1, the objective of this research, as well as the research methodology, has been explained in detail.

1.1 Understanding the problem

The objective of this project is to build a machine learning linguistic model which is learnable from any language, but the training resources and linguistic models are well-developed only for some languages. We have such models available quite conveniently for well-resourced languages but applying the same model on under-resourced languages is not productive due to the existing challenges with under-resourced languages. Before deep-diving into the existing challenges, It’s important to establish clarity on the definition of under-resourced languages since the term “under-resourced” is easily misinterpreted.

The term “under-resourced languages” was introduced by Krauwer [2003] and adjected by Berment [2004]. They define certain measures to consider a language as an under-resourced language such as: lack of a unique writing system or stable orthography, limited presence on the web, lack of linguistic expertise, lack of electronic resources for speech and language processing, such as monolingual corpora, bilingual electronic dictionaries, transcribed speech data, pronunciation dictionaries, vocabulary lists, etc. Additionally, underdeveloped terminology, lack of parallel data, and complex structure of language are valid standards to consider a language as under-resourced language. Under-resourced languages are also commonly referred to as “low-density languages”, “less-resourced languages”, “resource-poor languages”, “low-data languages” etc as well. It is interesting to notice that under-resourced languages aren’t the language that is spoken by a small percentage of the population. There are more than 6000 languages in existence and even languages spoken by millions of speakers can lack the resources we need to build language technologies.

One of the main challenges with Information extraction (IE) and natural language processing (NLP) is that we don’t have large text corpora on which deep learning can be applied to produce word embedding models and meaning of a particular word, sentence, or phrase can be apprehended. One way to deal with this challenge is to implement transfer learning by considering resources available for closely-related languages. For this project, Sanskrit and Marathi have been chosen because not only these languages share the same script, but Marathi shares a large corpus with Sanskrit. There have been claims that Marathi existed close to 2000 years ago alongside Sanskrit as a sister language. (Wiki, n.d.).

Another way to handle the challenge is to explore methods to adapt or transfer word embedding models learnt from “big languages” to handle related “small languages” (Sharoff 2018, Adams et al 2017) and for the same reason, extracted text dataset from the Bible has been taken for both Sanskrit and English languages. We have vast amounts of new data and information being generated that possess valuable economic and societal value due to the extensive use of online platforms. According to the International Data Group (IDG), unstructured data is growing at a rapid rate of 62% per year. IDG also suggests that approximately 93% of digital data would be unstructured by the year 2022 which explains the surge in the field of research in text analytics. Text Mining Techniques like Information Retrieval (IR), categorisation, clustering, and summarisation can exploit the potential of the data available in an unstructured format. Specifically, there is a significantly increased interest in finding linguistic features of under-resourced languages in the field of corpus linguistics and text/data analytics.

Another important challenge with extracting knowledge from under-resourced languages with the lack of availability of tools required to implement an accurate machine translation model. One of the reasons for the success of machine translation models for adequately-resourced or well-resourced languages (E.g. English) is the efficient tools used in the process. The language portability of these tools is limited to only a few languages having enough digital resources available.

1.2 The project aim

This research involves cross-lingual text modelling and adaptation to facilitate the implementation of a universal machine learning language translation model that can further be used for any other under-resourced language. Different approaches for building such model, using deep learning at the core, were explored and various methods have been considered to implement the idea. We also investigated the approaches to understand whether parallel corpora can compensate for the lack of resources for an under-resourced language.

The motivation behind the research comes from the fact that many (some of them) under-resourced languages are endangered and require immediate attention to encourage linguistic diversity with the help of technology. In the end, this project aims to constitute methods that are adjustable to new language and independent of language where possible. Also, with recent advancements in the field of computational linguistics, the demand for tools and study related to under-resourced language is witnessing rapid surge to accommodate a larger audience

1.3 Objectives

Objectives of this research are as follows:

- Understand the objective by perceiving the definition of under-resourced (UR) languages and existing challenges with building word embedding models for UR languages.

- Collect Sanskrit and Marathi language datasets.

- Pre-process collected data using steps like cleaning, tokenization, standardization, selection, transformation and selection to transform raw data into an understandable format. The purpose is to make sure that data has the appropriate format to be used as input in the machine translation model.

- Prepare training and testing data to machine learning language models.

- Evaluate the outcome using evaluation metrics to understand the performance of developed machine learning language model.

- Analyse the implementation of the proposed solution approach.

1.4 Minimum requirements and deliverables

As part of preparing project planning, minimum requirements and deliverables for this paper was established after careful study of aim defined for our research.

- Establish the comprehension of under-resourced languages.

- Explore and use a useful translation model to perform machine translation between natural languages.

- Present delivering data, software, and results to external clients in the School of Languages and/or School of English, to aid their research into under-resourced languages and for researchers in the field into under-resourced languages.

- Analyse the performance of different approaches and explain their respective benefits and shortcomings applicable to relevant scenario.

- Evaluate the outcome using evaluation metrics to understand the performance of the developed machine learning language model.

- At the end of the project, the dissertation should provide information that explains background research as well as all entire process of model development for language translation.

- Identify and evaluate the model manually by comparing English text samples from the Marathi and Sanskrit text samples.

1.5 Degree Relevance

This project concerns multiple areas of interests and skillsets that have been taught as part of my Advanced Computer Science (Data Analytics) course.

Module Code	**Module Name**	**Topics/tools/skills relevance to project**
COMP5122M	Data Science	Data acquisition, cleaning, profiling, quality investigation
COMP3611	Machine Learning	Deep learning algorithms, Anaconda
COMP5840M	Data Mining and Text Analytics	Text and data analytics, NLTK, LTSM, Word embedding models
COMP3736	Information Visualization	Use of different visualizations to present data pictorially.

Table 1.1: Degree relevance

Studying all four modules, as mentioned in the table above, has enabled me to comprehend all aspects of this project. Furthermore, this project is a great opportunity for me to implement acquired knowledge and skills from different modules intersecting with one another and get overall understanding.

1.6 Research methodology

This project deals with broad fields of study(Text and data analytics, Machine Learning, Data Mining etc.) and has many tools and techniques involved in its problem-solving process. Due to the nature of its complexity and overlapping of concepts/fields, It is important to approach the problem statement in a systematic, structured, and logical way which is simple to understand, follow and duplicate if needed.
CRISP-DM (Cross- Industry Standard Process for Data Mining) is the commonly used methodology for such implementations projects in data mining because it is neutral to industry, tools, or applications. CRISP-DM was developed, in 1999, by an association of data mining companies and purveyors.

CRISP-DM (Cross-Industry Standard Process for Data Mining) methodology has been followed throughout the realization of this project.

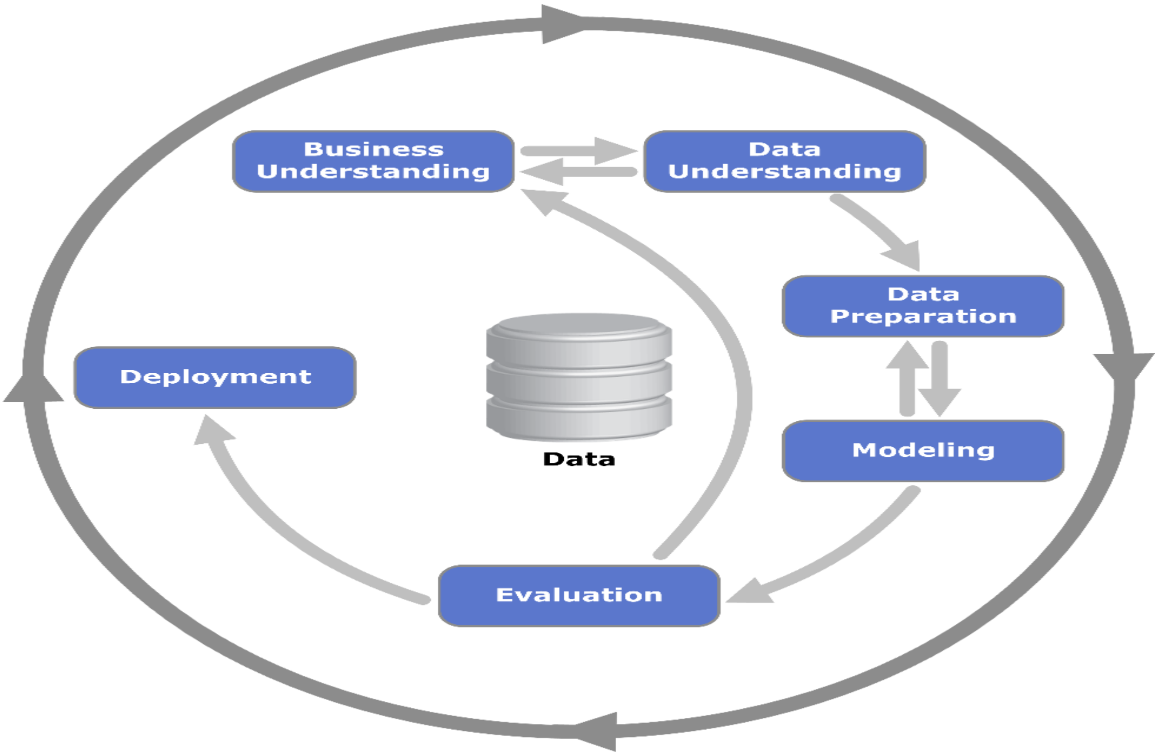


Figure 1.1 Cross-Industry Standard Process for Data Mining (Shearer, 2000)

Following the CRISP-DM methodology, our project life cycle can be divided into six main phases as mentioned below:

• **Business understanding:** This phase defines the problem statement and its purpose to the client. Tasks involved in this phase are setting objective, produce a project plan and finalizing business success criteria along with finding list the resources available to the project, assessing any data security concerns etc.
In this case, we also needed clarity on the definition of under-resourced language since there are too many versions available and the term "under-resourced" can be easily be misinterpreted. Also, understanding the purpose of this project was important for client gives clarity on the overall understanding of the project.

• **Data understanding:** This phase deals with understanding the dataset that is going to be used throughout for modelling. Tasks involved in this phase are describing and exploring collected data and verifying data quality. This phase became crucial in this project because we don't have enough data available for under-resourced languages. Ethical issues also need to be addressed to make sure that licensed data isn't being used without required approval. Second step was to perform rigorous investigation and select appropriate dataset which aligns with the overall task. Third and most important task in this phase was to explore and assess the quality of datasets that have been collected for this project.

• **Data preparation:** Third phase of our project is data understanding and like any other data mining project this process takes up most of the time due to the criticality that structure, quality, accuracy, and level of cleanliness decide overall quality and accuracy of the model. Tasks involved in this phase are cleaning and constructing the required data. Data cleaning, profiling, constructing, integrating, formatting, and other NLP techniques for data pre-processing techniques were applied in this phase of the project.

• **Modelling**: Modelling phase consists of selecting an appropriate modelling technique, generating a test plan, and building the model, and assessing the model. This phase involves performing text mining tasks on data and exploring the optimal solution by designing/applying applicable machine learning algorithms and models best suited to the objective. Next step is to deciding split the data into training and testing data for the evaluation step.

• **Evaluation:** In the evaluation phase, we assess whether the developed model fulfils the aim and objective for overall projects. Output of the previous step i.e. modelling provides us with a developed model trained on Sanskrit, English, and Marathi. This model is capable of learning from any other under-resourced dataset, but the model needs to be evaluated on the evaluation metric to understand how "good "are the results. This phase involved evaluating the developed model. We have various evaluation metrics available such as precision, accuracy, recall, F1 score and AUC. Also, we Summarise the process review and establish the tasks that should have been included and those that should be exercised in future as well. Finally, we determine next steps.

• **Deployment**: The final phase of this project is deployment. It involves Summarising the monitoring and maintenance strategy, including the involved steps and instruction on how to replicate them. Documentation is an essential aspect of this phase. This Masters dissertation is the outcome of the deployment phase for this project.

1.7 Thesis Outline

This paper is organised into five chapters. Starting with the Introduction chapter, research problem, aim and objective of the project is presented precisely. Requirements and deliverables of this study have been established precisely in this chapter. Also, we discuss the underpinning research methodology, thesis outline, and project plan followed over the entire course are discussed in detail.
Chapter 2 explains the background research about under-resourced languages, text mining, natural language processing, evolution of machine translation, applications, word embedding techniques, and approaches for natural language translation.
In chapter 3, we discuss the approach, which is based on the Sequence-to-Sequence model and deep learning. Also, working and underlying structure of Seq2seq model has been explained.
Chapter 5 discusses the evaluation of the developed translator model and any other relevant outcome.
Eventually, Chapter 6 concludes this paper by giving overall idea of all phases and stages of this project and discussing existing challenges, potential solutions and possibilities to take this research further.

1.8 Project Plan

Preliminary work for this project started in March 2019 but considering the study and semester examination commitments, the project took its proper shape around June.
The timeframe of June-September was completely devoted to this project.
Scoping and planning for this project was done within three weeks of project start date. However, the plan was revised to accommodate additional background research to be able to select appropriate tools and techniques.

Data collection cab under-resourced languages, while making sure that ethical issues are addresses, took more time than expected and scheduled.
For Example, natural data processing requires huge text corpus to process. After data, collection, pre-processing, and training, it was found out that dataset isn't large enough and the entire process needs to be repeated with much bigger dataset.
Figure 1.2, as mentioned below, shows the project plan.

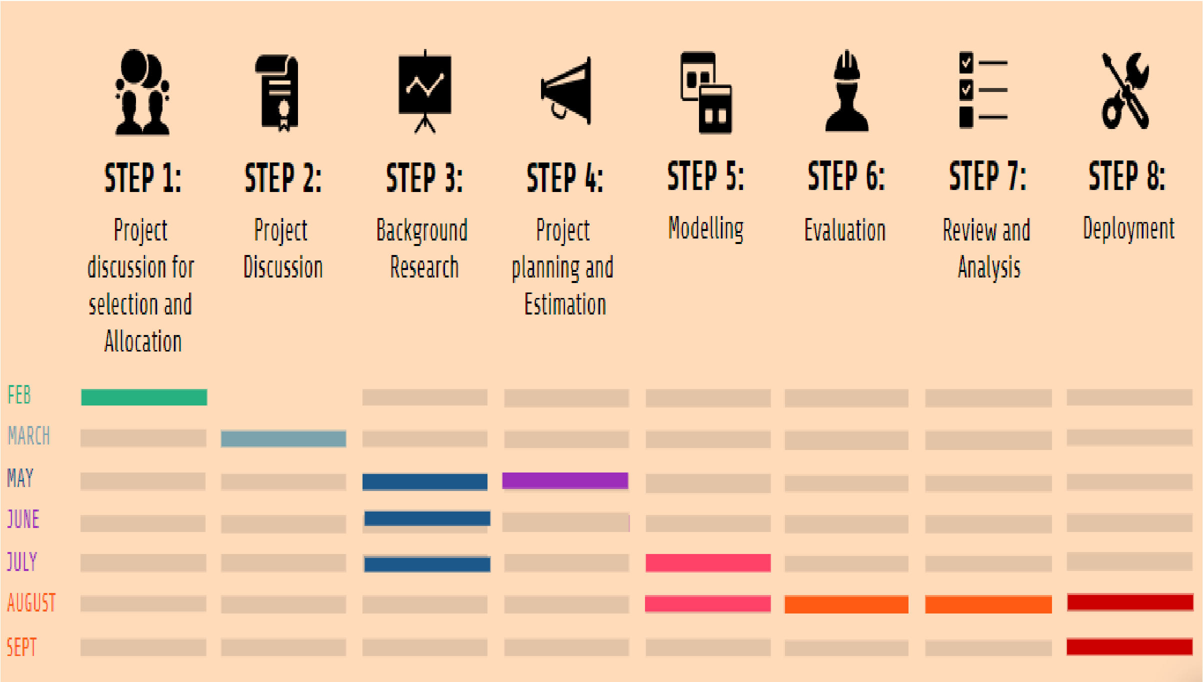


Figure 1.2 : Project plan

Chapter 2

Background research

2.1 Under-resourced languages language identification

We need to select three language to achieve our objective keeping in mind that our project objective is achieved by combining two ideas as mentioned below:

- **Small or UR languages can be learned by big or well-resourced language**

  Sanskrit and English have been selected as small and big language respectively for this project. Sanskrit is an Indo-Aryan language (Wikipedia suggests) with close to 3000-year-old history. The study of syntax has been found as early as the 4th-c. BC in Sanskrit grammar of the Indian grammarian Pāṇini. A lot of philosophical and religious texts have been written in classical Sanskrit. There is various other text available for music, drama, technical, etc. as well. However, Sanskrit has a quite limited presence on digital platforms and it's challenging to find a dataset that both extensive and accurate at the same time.

- **Small languages can be learned by closely-related languages**

  Marathi has been considered as the third language since Sanskrit and Marathi can be considered as closely-related languages. Marathi is an Indo-European language family and is the fourth widely-spoken language of India. Similar to Farsi and Arabic union, Marathi and Sanskrit are considered to be belonging to the same category. Sanskrit and Marathi have many common words that are semantically and morphologically same. For e.g., (Tatsam, तत्सम) holds the same meaning and reference in both Sanskrit and Marathi. Table 3.2, in chapter 3, has an elaborate list of such words. Sanskrit and Marathi are both written in Devanagari script and also considered to be sister languages. Many grammar rules, the pronunciations, the alphabets, etc. are directly derived from Sanskrit.

2.2 Semantics of English and Sanskrit

English is a well-resourced as well as well-structured language, whereas Sanskrit is an ancient language and considered to be the origin of many Indian languages. The basic structure of sentence formation in the English language follows Subject+verb+object format.

Unlike English, Sanskrit doesn't have a fixed format. However, the grammatical meaning of each sentence is conserved by the change introduced by the change of ordering of words.

The Sanskrit language has forty-two character, also known as varanas, in its alphabet. At the other hand, the English language has twenty-six characters in its alphabets. Sanskrit have nine vowels, also called swaras (a, aa, i, ii, u, uu, re, ree and le), and thirty three consonants, also known as, vyanjanas whereas English has five vowels (a, e, i, o, u) and twenty-one consonants while Sanskrit has nine vowels or swaras (a, aa, i, ii, u, uu, re, ree and le) and thirty-three consonants or yanjanas.

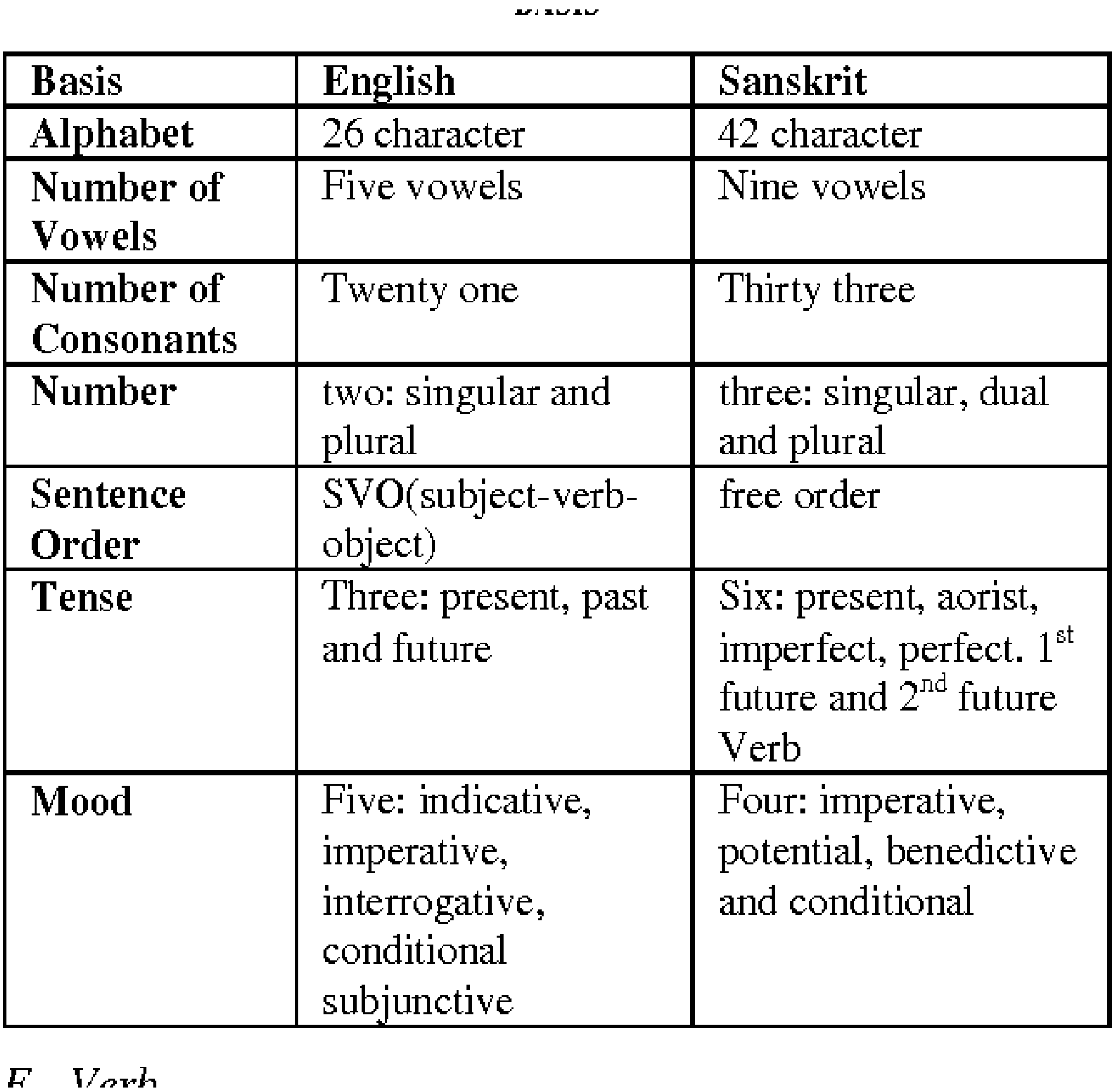
We have tried to understand simple grammar and rules for both languages to have a basic understanding before we proceed to the modelling of our translation model.

- **Gender**

  Any noun has three genders: masculine, feminine, and neuter. In English, there are two groupings of numbers: Singular and plural, whereas Sanskrit has three groups of numbers: singular, dual, and plural.

- **Pronoun**

  Sanskrit follows Paninian Grammar and it suggests that Sanskrit has 35 pronouns. These pronouns have been categorised into nine classes. All of these pronouns have different accentuation forms.



Basis	English	Sanskrit
Alphabet	26 character	42 character
Number of Vowels	Five vowels	Nine vowels
Number of Consonants	Twenty one	Thirty three
Number	two: singular and plural	three: singular, dual and plural
Sentence Order	SVO(subject-verb-object)	free order
Tense	Three: present, past and future	Six: present, aorist, imperfect, perfect. 1^{st} future and 2^{nd} future Verb
Mood	Five: indicative, imperative, interrogative, conditional subjunctive	Four: imperative, potential, benedictive and conditional

E Verb

Table 2.1: Comparative study of languages Sarita G. Rathod, Shanta Sondur (2012)

Table 2.1 establishes the fundamental difference between both languages, English and Sanskrit, in terms of how sentences are structured and the other basic rules of grammar. The purpose of this brief syntactic and morphologic analysis can also to understand possible category ambiguity issues (Same word can be used as a noun, verb, or an adjective depending on the context. For e.g. “light”).

2.3 Text analytics and Data Mining

Text analytics emerged in the late 1990s as “text mining” or “text data mining”. Text mining is the process of capturing structured and valuable information from huge collection of text data with use of well-defined tools and techniques. Though the end result is quite similar, text mining/analytics and data mining are different. Data mining is a spectrum of multiple approached clubbed together to find patterns and relationships in data.
It deals with all possible formats of data and is not limited to text data alone like text mining. The biggest difference between data mining and text mining is that text mining extract information from unstructured data (Natural language text) whereas data mining operates on structured data from various data sources.

Text mining, along with other technologies, uses natural language processing (NLP) to transform natural(human) language into normalized, structured data appropriate for further analysis and machine learning (ML) algorithms. The structured data created by text mining can be integrated into databases or any similar storage meant for storing structured data and used for predictive, descriptive, or prescriptive analytics.

2.4 Natural Language Processing (NLP)

Natural language processing (NLP) is a field of artificial intelligence that enables machines to read, interpret, and capture information from natural or human languages. Human languages are complex, and each language has its own set of established vocabulary, rules, grammar that overall construct the language.

Data generated from interactions, discussions, declarations, books, web pages, mobile applications, and numerous social media platforms is completely unstructured. Unstructured data doesn’t fit into traditional relational database format. Majority of the real-world data is unstructured and recent advances in machine learning field enables machine to capture information based on the meaning of words rather than keyword itself.
The goal of building systems using NLP is to be able to develop systems that process and contemplate text like a human mind which is by forming a representation of text with understanding of objects, goals, relationships, beliefs etc.

This goal is quite difficult to with existing challenges of NLP like:

- Language models start learning from scratch and lack common sense that human mind puts to use when processing a text statement.

- Inability of any model to process emotions which is an essential part of processing texts in human minds

- Lack of embodied learning i.e., learning from experiences and interactions.

> Though it is often discussed that using an agent embedded in an extensive environment, with an appropriate reward structure, could replicate the understanding achieved by humans due to embodied learning. However, the compute for such environment can be expensive and difficult to implement.

In addition to all the challenges mentioned above, NLP with Under-resourced languages have to deal with a few more issue.

Universal language model: Building a universal language model which exploits the cohesion of various languages is the idea that translations models are expected to implement. However, not having enough data makes it difficult to create such model. This is close to the idea of having a cross-lingual transformer language model.

Cross-lingual representations: In recent times, under-resourced languages have been getting a lot of attention in the NLP community. However, there are close to 1,250-2,100 languages in Africa alone, most of which haven’t been worked on(Stephan Gouws, 2018). Cross-lingual datasets align word embedding tasks efficiently to perform tasks like text similarity or subject classification but don't allow for tasks such as machine translation.

Benefits and impact: Even with under-resourced languages, We want to eventually build models that not only encourages lingual diversity with the help of technology but also enables people to read books that was not originally published in their language, ask queries about their health in case of not having immediate accessibility to their healthcare systems, etc.

NLP still is one of the most powerful techniques used in text mining and it enables the recognition and prediction of diseases, sentiment analysis, cognitive assistant, stopping spam, identifying fake news, voice-driven interfaces etc. There are most frequently used NLP algorithms such as Bag of Words, tokenization, stop words removal, stemming, topic modelling, and lemmatization. Tokenization is the process of segmenting texts into small units called tokens. As part of the pre-processing phase, the text needs to be segmented into linguistic units such as words, numbers, punctuation, alpha-numeric, etc. Stemming and lemmatization are both text normalization techniques and intend to reduce morphological variation of text.

While stemming reduces word to stems by chopping off prefix or suffix, lemmatization tries to find a linguistically appropriate word-form called lemma. This difference is quite crucial in languages having complex morphology.
For example: stemming handles matching “car” to “cars” whereas Lemmatization would handle matching “car” to “cars” along with matching “car” to “vehicle”.

2.5 Machine Translation (MT) approaches

Machine translation (MT), often misinterpreted as computer-aided translation, machine-aided human translation (MAHT), is the process of machine translation by which a developed translation model/software can translate a text from one natural language (such as English) to another (such as French). (Wiki MT, n.d.)

To process any language translation, automated or human, the in-depth knowledge to appreciate the meaning as well as essence is required which makes language translation a complex task. Translation of languages go beyond simple substitution of a word in one language to literal word in another language. This requires substantial knowledge of semantics, morphological structure, grammar, sentence structure, etc., in both source and target languages in a detailed and analytical way. In almost all languages, speakers use narratives and metaphors (Polkinghorne 2005). These metaphors are specific to culture and language (Lakoff and Johnson 1980).

For example, there is a common saying, in Dutch, to give a proposal ‘hands and feet’ (handen en voeten geven in Dutch) to express the physical work that is required to make the proposal concrete. This saying is hard to understand for native English speakers (Otis 2008). Similarly, machine translation presents its own challenges. How machine translation can produce “good” translations remains to be the greatest challenge.

Machine translation, in its rich history of multiple decades, has been classified and categorised into several systems. To completely understand the differences between these translation systems, it is important to understand the classifying categories, the different scenarios when these translation systems are used, the intended usage of these systems, and linguistic techniques used by these MT systems to handle translation tasks. There are three broad categories of machine translation system: Human translation with machine support, machine translation with human support, fully automated machine translation.

As part of this research study, various machine translation approaches have been explored. There are primarily four paradigms of machine translation: Rule-based systems (RBMT), statistical systems (SMT) and neural machine translation (NMT), and Hybrid machine translation (HMT), but other approaches have also been analysed.

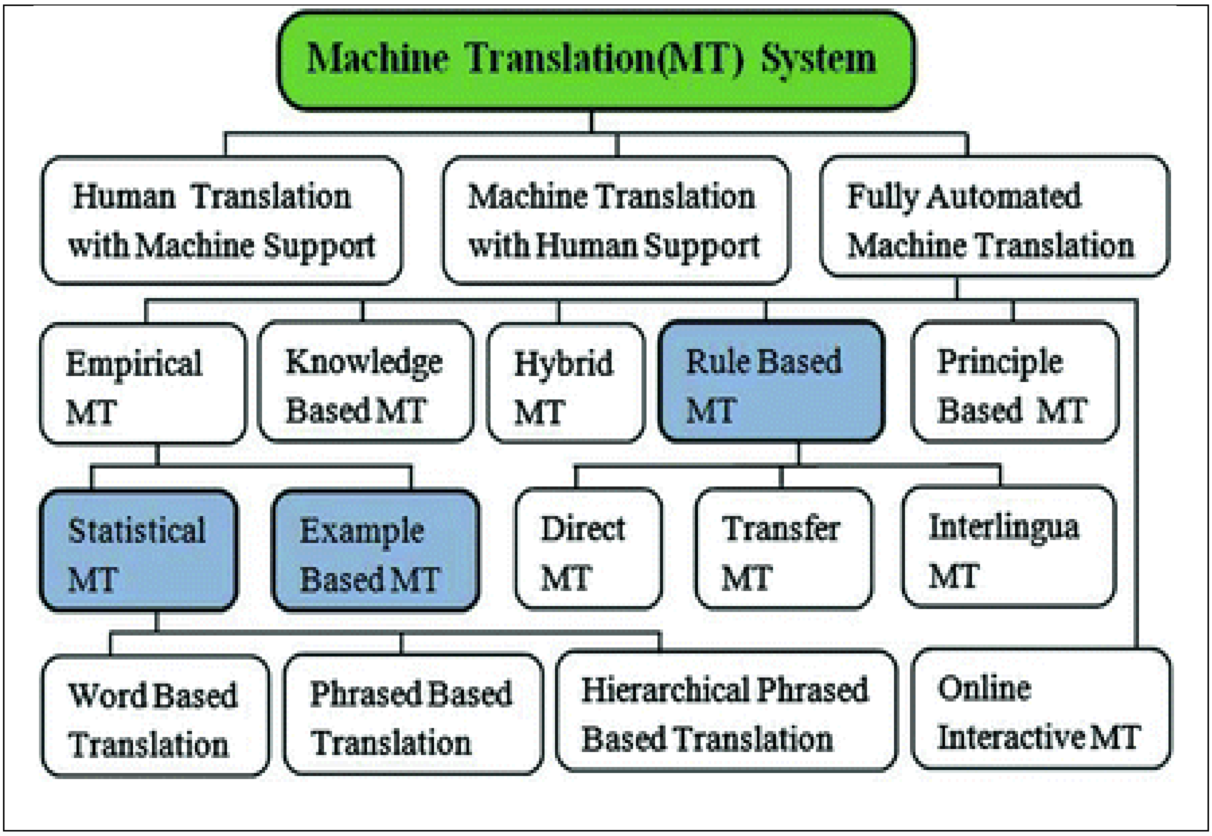

Machine Translation(MT) System
Human Translation with Machine Support
Machine Translation with Human Support
Fully Automated Machine Translation
Empirical MT
Knowledge Based MT
Hybrid MT
Rule Based MT
Principle Based MT
Statistical MT
Example Based MT
Direct MT
Transfer MT
Interlingua MT
Word Based Translation
Phrased Based Translation
Hierarchical Phrased Based Translation
Online Interactive MT

Figure 2.1: Machine translation Md. Saiful Islam, Bipul Syam Purkayastha (2007)

Figure 2.1 gives an overview of the classification of machine learning translation systems. As part of background research, few systems have been explored before selecting the translation system for this project

- **2.5.1 Rule-Based Machine Translation (RBMT)**

Rule-based machine translation, also known as knowledge-based machine translation, was developed in the early 1970s. RBMT is a subsystem of “machine translation with human support” category translation system. It a machine translation approached based on specification of rules for morphology, syntax, semantic regulations for both source and target languages. These specifications are derived from all kinds of dictionaries available covering vocabulary of either or both languages. Rule-based machine translation analyses the input text of source language based on linguistic rules and generates texts in the target language. Usually, there are two steps:

a) In most cases, an initial investment that considerably improves the quality.
b) Ongoing investment to increase quality systematically.

While rule-based machine translation brings companies to a significant quality threshold, the process involved is time-taking and expensive. This process requires extensive lexicons with morphological, syntactic, and semantic information, and large sets of rules.

RBMT uses a complex rule set and then transfers the grammatical structure of the source language into the target language.

Rule-Based Machine Translation is then further classified into Interlingual Machine Translation(IMT) and Transfer-based Machine Translation (TMT)

- **2.5.1.1 Interlingual Machine Translation (IMT)**
  Interlingual machine translation is one of the of rule-based machine-translation approaches. IMT follows the approach that the text of source language is transformed into an interlingual language. Interlingual language is the intermediate represented which is independent of source or target language. The target language is then generated using interlingua. The limitation with interlingual Lachine translation approach is to capture meaning from texts in the source languages to create the interlingua.

- **2.5.1.2 Transfer-based Machine Translation (TMT)**
  Transfer-based Approach addressed the limitations of the Interlingua approach. Transfer-based machine translation approach is similar to interlingual machine translation since the translation is based on intermediate representation which captures the meaning of the original sentence. However, transfer-based machine translation isn’t completely independent of languages involved in the approach. TMT considers differences in structure between the source and target language. This approach can produce translations with accuracy close to 90% (Okpor, 2014).

RBMT isn’t considered to be an efficient model for machine translation tasks due to the drawbacks such as Insufficient number of reliable dictionaries, manual intervention required to establish linguistic, dealing with rule interactions. Example of RBMT are Apertium, GramTrans, Japanese MT systems, Systran, etc.

- **2.5.2 Corpus-Based Machine Translation (CBMT)**

  Corpus-based machine translation (CBMT), also known as data-driven machine translation), is an alternative approach for machine translation. CBMT was introduced in the late 1980s. There are three kinds of corpus used in machine translation: parallel corpus, comparable corpus, and multi-language corpus.
  The idea behind CBMT is to address the issue of knowledge extraction in case of rule-based machine translation. This approach uses a vast amount of data, having text and their translated versions, present in bilingual parallel corpus to capture valuable information. they allow for fast prototyping.

  Corpus-based machine translation approach is broadly categorised into sub approaches as follows:

- **2.5.2.1 Statistical Machine Translation (SMT)**

  Statistical machine translation(SMT) was discovered by researchers at IBM's Thomas J. Watson Research Center (SMT, n.d.). Translations in SMT are generated base on, as the name suggests, statistical models. Parameter of these statistical models are based on the analysis of bilingual text corpora used in this approach. SMT works on word frequency and word combination. However, SMT can be based on different subgroups: word-based, syntax-based, phrase-based, and hierarchical phrase-based. Accuracy of SMT is inversely affected in case the corpora contain idioms, slangs, or casual styling. The corpora should be customized for a certain style to avoid the drastic falling of accuracy. In most of the cases, even customization is not enough to enable SMT to translate idioms.

- **2.5.2.2 Example-based Machine Translation (EBMT)**

  Most recent machine translation research have been around statistical machine translation and example-based machine translation. The underlying idea behind example-based machine translation is that existing large volume of parallel bilingual texts (translated texts) can be used to perform machine translation in a more efficient manner than other approaches. It is proposed that statistical machine translation is also an example-based approach (1998; 2000a).

Some minor languages (Forcada, 2006) may not easily benefit from corpus-based machine translation approaches since the available parallel corpora are not large enough.

- **2.5.3 Hybrid Machine Translation (HMT)**

Hybrid machine translation (HMT) approach, as the name suggests, is combination of two or more machine translation approaches, where at least one of them is rule-based and another is statistical-based approach. The prime idea behind the implementation of this approach is to take advantages of benefits of both, rule-based and statistical approaches while diluting their drawbacks.
Recently, we have witnessed rules being included in statistical machine learning and statistical knowledge being included in rule-based machine translation model, which is the perfect example of hybrid machine translation.

- **2.5.4 Neural Machine Translation (NMT)**

Neural Machine Translation (NMT) has been proven to be the most powerful mechanism to perform language translation when combined with the capabilities of deep learning. In 2016, Google introduced Google Neural Machine Translation (GNMT) which is essentially neural machine translation (NMT). Though a lot of work has been done on NMT and multiple variations of NMT are being investigated by multiple companies.

However, GNMT has consistently been the leading application of NMT. NMT uses huge datasets of translated sentences to train a model that can conveniently translate text of one language to another. One of the most established architecture that NMT follows is Encoder-Decoder structure. This architecture is comprised of two recurrent neural networks (RNNs).

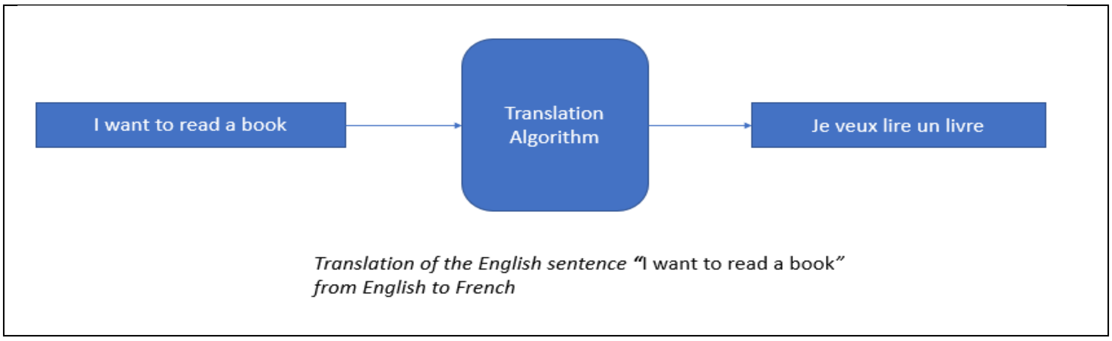

I want to read a book
Translation Algorithm
Je veux lire un livre
Translation of the English sentence "I want to read a book" *from English to French*

Figure 2.2: Machine translation English to Spanish of the English sentence

Figure 2.2 is a naive representation an NMT algorithm (For e.g. GNMT) translating from English to French for a simple English sentence “I want to read a book”.

2.6 Word Embedding Models

The Notion of embedding refers to representing discrete objects to vector to achieve numerical representation. Word embedding, in particular, is learned representation of text data where words or phrases having the same meaning have a similar representation in form of a vector of numerical values. In case of non-text systems, like speech or image recognition systems, the information is available in feature vector format. However, in the case of raw text data, individual words hold their own identifiers and no semantic relationship exists amongst words.

In natural language processing (NLP), text corpora is huge and text embedding helps to reduce the dimensionality of text data which is one of the key breakthroughs considering the challenges NLP has to face in process large datasets.

Text embedding ensures If two words or documents have the same embedding, they are same semantically. This notion of word embedding is summarized by an often-repeated remark by John Firth:

> *“You shall know a word by the company it keeps!”*

There are many techniques to create Word Embeddings. Some of the popular ones are Binary Encoding, TF (term frequency), Encoding, TF-IDF (term frequency-inverse document frequency) Encoding, Latent Semantic Analysis Encoding, Word2Vec Embedding.

2.6.1 Word2vec

Word2Vec is a two-layer neural network for learning word embeddings from raw text. Word2vec, itself, is not a deep neural network, it turns text into a numerical form that deep nets can understand. Word2vec is used to categorize the vectors of words having the same/similar meaning together in vector space. It finds out the similarities in mathematical terms. Vectors created by Word2vec are essentially distributed numeric representation of features (For e.g. context) contained in words.

Word2vec comes in two flavours, the Continuous Bag-of-Words model (CBOW) and the Skip-Gram model.

2.6.1.1 Continuous Bag of Words (CBOW) Model

CBOW learns to predict target words (e.g. dog) from source context words ('the quick brown fox jumps over the lazy") which serves as the context. CBOW strides the entire observation as to make sense of the entire context. CBOW has proven to be a powerful model for smaller datasets.

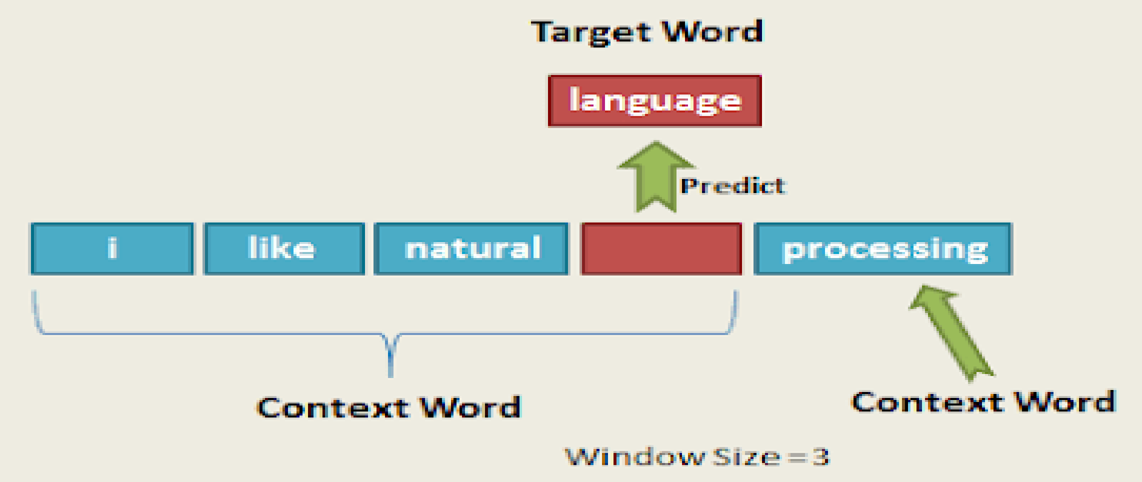


Figure 2.3: CBOW model (Image source: https://www.thinkinfi.com/2019/06/single-word-cbow.html)

Window size mentioned in figure 2.3, indicates the number of words before a given word ("language" in this case) would be included as context words of the given word.

2.6.1.2 Skip-Gram model

Skip-Gram learns to predict the context by the words. It can be seen as inverse of continuous bag of words (CBOW) model. This model takes the word as input and provides the context/surrounding words as output. Each context-target pair is treated as a new observation in Skip-Gram model, and this precisely makes Skip-Gram a better model when dealing with larger datasets. Also, Skip-gram model, being an unsupervised model, can work on any unstructured and raw text. Skip-gram consumes less memory as compared to other word embedding models. However, the time taken for training skip-gram algorithm is on the higher side.

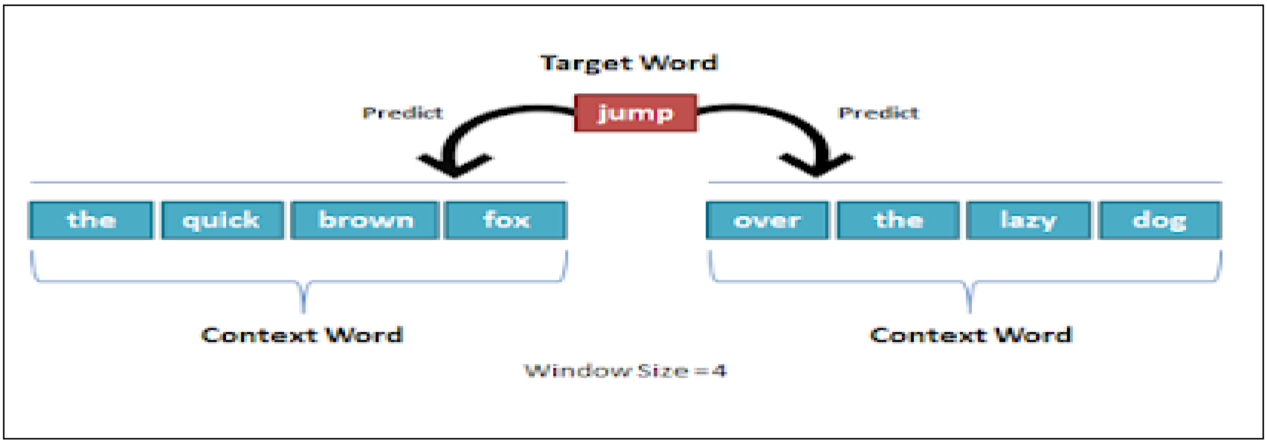


Figure 2.4 Skip-gram (source: https://www.thinkinfi.com/2019/08/skipgram-explained.html)

For example, explained in the above figure, skip-gram model predicts the word “jump” surrounding word with window size 4.

2.6.1.3 FastText

FastText is considered to be an extension of word2vec model. It was developed by Facebook AI research lab in the year 2015. It is extremely fast, reliable, and compatible with most of the existing systems. FastText is used for both text classification and word embedding. FastText handles each word made of the sum of character ngrams. For example, the word vector “stage” is a sum of the vectors of the n-grams “<st”, “sta”, ”stag”, ”stage”, ”stage>”, “tag”, ”tage>”, “age”, ”age>”, ”ge>” (Assuming smallest ngram[minn] is 3 and largest ngram[maxn] is 6). This enables FastText to be able to form a vector for a word from character n-grams. That explains the ability of FastText to generate word embedding for words that aren’t even present in the corpus.

2.6.1.4 GloVe

GloVe is an unsupervised learning algorithm for representing words in form of vector. First, global word to word co-occurrences statistics from a corpus is obtained and then the training is performed on global statistics. The primary difference between word2vec and GloVe is that word2vec is a prediction model whereas GloVe is a count-based model. Glove has been used as the embedding framework for the systems designed to detect psychological distress in patient interviews. (wiki-GloVe, n.d.). Pennington et al. (2014) introduced us to the Global vector (GloVe) model.

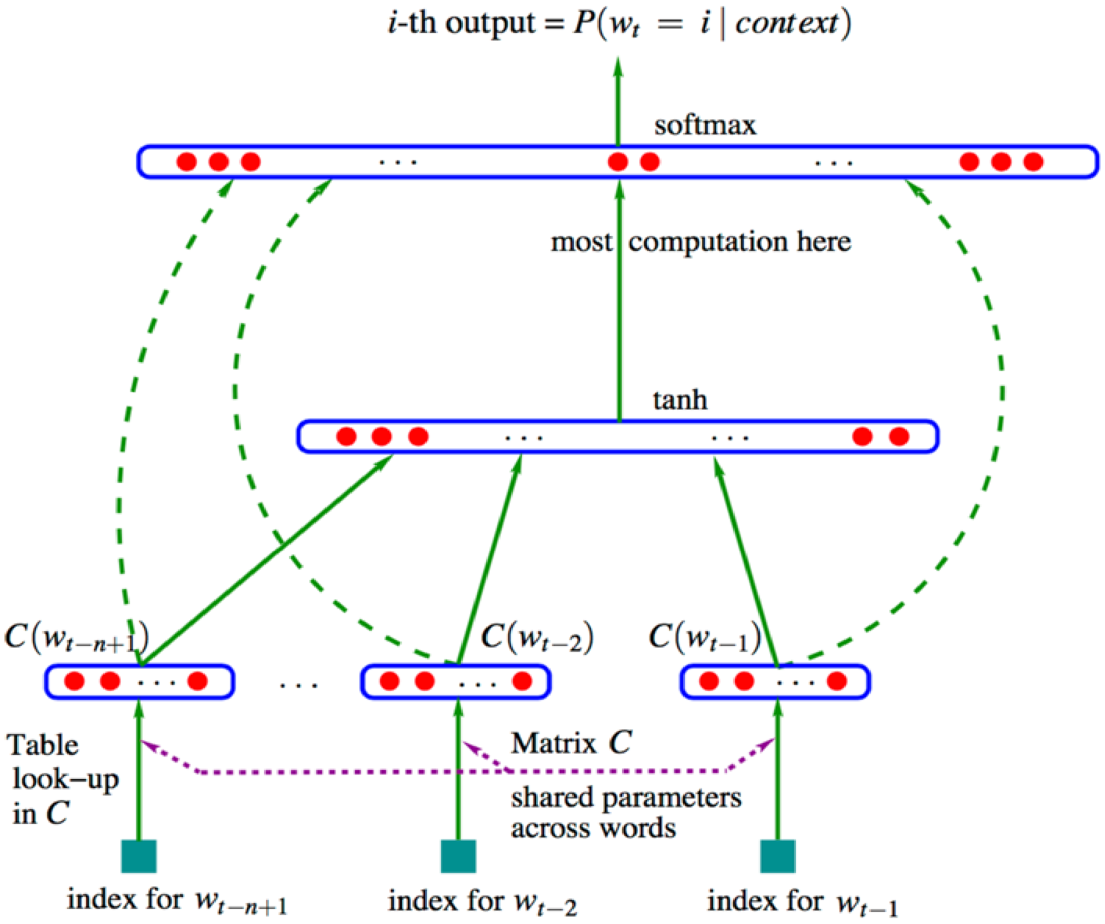


Figure 2.5: Classic neural language architecture (Bengio et al. 2003)

2.6.1.5 Gensim

Gensim, introduced in 2009, is an open-source library designed for unsupervised modelling for unsupervised learning and natural language processing. Since the modelling algorithms are unsupervised in Gensim, no human intervention is required. It only needs a corpus of text. Gensim contains effective implementation for the tasks of topic modelling such as latent semantic analysis (LSA), TD-INF, latent Dirichlet allocation (LDA) etc. Gensim even has implemented word2vec within itself with additional functionality which means it has all merits of word2vec as well as its own benefits.

2.7 Machine translation tools

This section provides an overview of some existing software solutions commonly used in machine translation. We also discuss other software platforms that provide an integrated environment for building translation models in data science projects. In addition, since analysis and evaluation of translation models hugely reply on empirical investigations, these tools can be used by anyone working on a similar project.

Table 2.2 presents some of the machine translation tools that have been used for this project after considering various relevant tools.

Tool Name	**Purpose**	**Language**
NLTK (Bird 2006)	Computational linguistics	Python
Anaconda	Modelling, analysis, and visualization	Python
Tableau SDK	Visualization	Java
Weka	Data mining and analysis	Java

Table 2.2: Machine translation tools

Chapter 3

Machine Learning language model for UR languages

3.1 Overview

This chapter describes the use of machine learning language models, along with text mining, for under-resourced languages following a sequence to sequence (Encoder Decoder) model. Different NLP techniques such as cleaning, formatting, tokenization has been used to pre-process unstructured natural language data into appropriate format for deep learning algorithm. This algorithm is trying to build a sequence to sequence model using LSTM. The seq2seq models are the best choice for solving complex problem related to NLP. To use the seq2seq model it is needed to build an encoder-decoder architecture which both of them are also LSTM models. The encoder reads the input sequence and summarize it into a hidden layer of LSTM having vector representation. The decoder is able to receive this data and generate a sequence based on it.

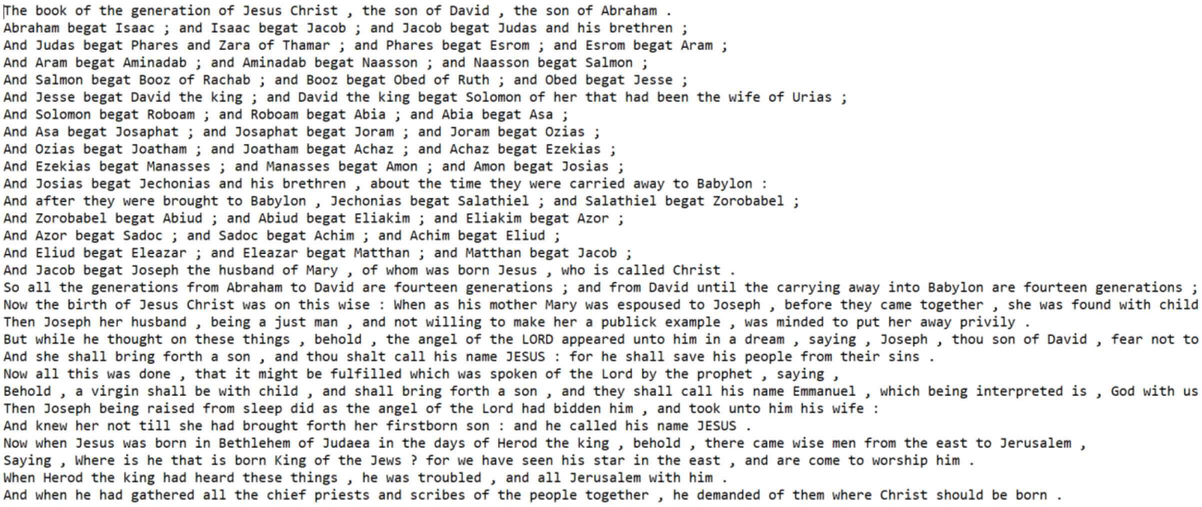
3.2 Software tools

Anaconda is a free and open-source distribution of the Python and R programming languages. Anaconda facilitates simple package management and deployment process. It has its own package management system called “conda” to handle package versions. All input data were present in the text file format. Jupyter notebook, which comes with complete Anaconda installation, was used to apply deep learning algorithm. The Jupyter Notebook is an open-source web application used for data cleaning and transformation, numerical simulation, statistical modelling, data visualization, machine learning. (Jupyter, n.d.)
For visualization purpose, Tableau and Microsoft PowerPoint has been used throughout this project.

3.3 Data Understanding

Data understanding is the second stage of CRISP-DM. During this stage, the dataset that are going to be used for modelling purpose are analysed closely. For this project, we have three text datasets. First is English text data taken from the holy book “Bible”. Second is Sanskrit text data, again, taken from the holy book “Bible”. This text file is essentially translated version of “Bible” (English to Sanskrit). Both files have phrases and sentences of various lengths. The motive of taking these text files for this project is to implement the idea that “small” (under-resourced) language, like Sanskrit, can be learnt from “big” (Well-resourced) language like English (Sharoff 2018, Adams et al 2017).

Please find below snippet of overall dataset:



```
The book of the generation of Jesus Christ , the son of David , the son of Abraham .
Abraham begat Isaac ; and Isaac begat Jacob ; and Jacob begat Judas and his brethren ;
And Judas begat Phares and Zara of Thamar ; and Phares begat Esrom ; and Esrom begat Aram ;
And Aram begat Aminadab ; and Aminadab begat Naasson ; and Naasson begat Salmon ;
And Salmon begat Booz of Rachab ; and Booz begat Obed of Ruth ; and Obed begat Jesse ;
And Jesse begat David the king ; and David the king begat Solomon of her that had been the wife of Urias ;
And Solomon begat Roboam ; and Roboam begat Abia ; and Abia begat Asa ;
And Asa begat Josaphat ; and Josaphat begat Joram ; and Joram begat Ozias ;
And Ozias begat Joatham ; and Joatham begat Achaz ; and Achaz begat Ezekias ;
And Ezekias begat Manasses ; and Manasses begat Amon ; and Amon begat Josias ;
And Josias begat Jechonias and his brethren , about the time they were carried away to Babylon :
And after they were brought to Babylon , Jechonias begat Salathiel ; and Salathiel begat Zorobabel ;
And Zorobabel begat Abiud ; and Abiud begat Eliakim ; and Eliakim begat Azor ;
And Azor begat Sadoc ; and Sadoc begat Achim ; and Achim begat Eliud ;
And Eliud begat Eleazar ; and Eleazar begat Matthan ; and Matthan begat Jacob ;
And Jacob begat Joseph the husband of Mary , of whom was born Jesus , who is called Christ .
So all the generations from Abraham to David are fourteen generations ; and from David until the carrying away into Babylon are fourteen generations ;
Now the birth of Jesus Christ was on this wise : When as his mother Mary was espoused to Joseph , before they came together , she was found with child
Then Joseph her husband , being a just man , and not willing to make her a publick example , was minded to put her away privily .
But while he thought on these things , behold , the angel of the LORD appeared unto him in a dream , saying , Joseph , thou son of David , fear not to
And she shall bring forth a son , and thou shalt call his name JESUS : for he shall save his people from their sins .
Now all this was done , that it might be fulfilled which was spoken of the Lord by the prophet , saying ,
Behold , a virgin shall be with child , and shall bring forth a son , and they shall call his name Emmanuel , which being interpreted is , God with us
Then Joseph being raised from sleep did as the angel of the Lord had bidden him , and took unto him his wife :
And knew her not till she had brought forth her firstborn son : and he called his name JESUS .
Now when Jesus was born in Bethlehem of Judaea in the days of Herod the king , behold , there came wise men from the east to Jerusalem ,
Saying , Where is he that is born King of the Jews ? for we have seen his star in the east , and are come to worship him .
When Herod the king had heard these things , he was troubled , and all Jerusalem with him .
And when he had gathered all the chief priests and scribes of the people together , he demanded of them where Christ should be born .
```

Figure 3.1: English dataset (verses from the Bible)

Third and final dataset of this project is having translated version of Marathi to English. We have a collection of small words to big sentences. This file was added later in the project to improve the learning of model since we don’t have enough data available for Sanskrit (being under-resourced language) online. Inclusion of this file was inspired by the idea that under-resourced language can be learned better from closely-related languages (Rios and Sharoff 2016).

Marathi	Sanskrit
राज्य	राज्य
आवश्यक	आवश्यक
आरोग्य	आरोग्य
आनंद	आनन्द
प्रथम	प्रथम
क्रीडा	क्रीड़ा
संगीत	सङ्गीत
पुस्तक	पुस्तक
स्थान	स्थानं
इतिहास	इतिहास
पत्नी	पत्नी
वस्तू	वस्तु
संगणक	संगणक
संदेश	संदेश
विवाह	विवाह
युद्ध	युद्ध
संस्कृत	संस्कृतम्
सूर्य	सूर्य
राजधानी	राजधानी

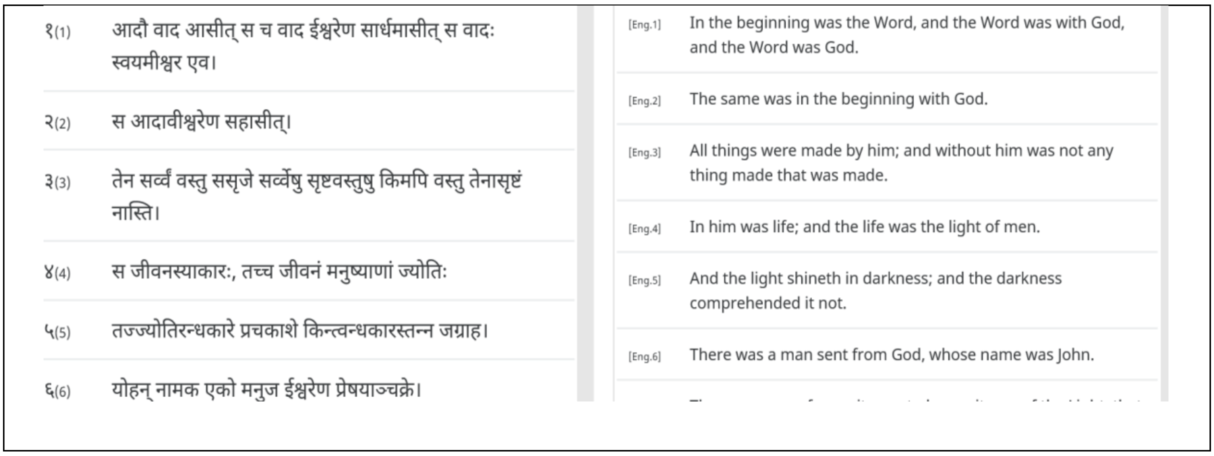
Table 3.2: Common words between Sanskrit and Marathi

3.3.1 Data Sources

With the advancement of technology and presence of numerous platforms hosted on the internet, finding a relevant dataset for any text mining project can be achieved with ease. We still need to make sure that ethical issues are addressed, and the data sources have reliable and updated content. In case of this project, the challenge started with being able to find datasets having enough data so that natural language processing (NLP) can be applied and model can then be developed. Datasets used for this study above have been taken from the sources mentioned below:

- http://sanskritbible.in

  SanskritBible has the Holy Bible available in Sanskrit and more than 20 other languages. The New Testament portion of Sanskrit Bible is freely available for download. Side by side Bible translations in various other languages is also given. Json objects containing a pair of text can be obtained by querying.

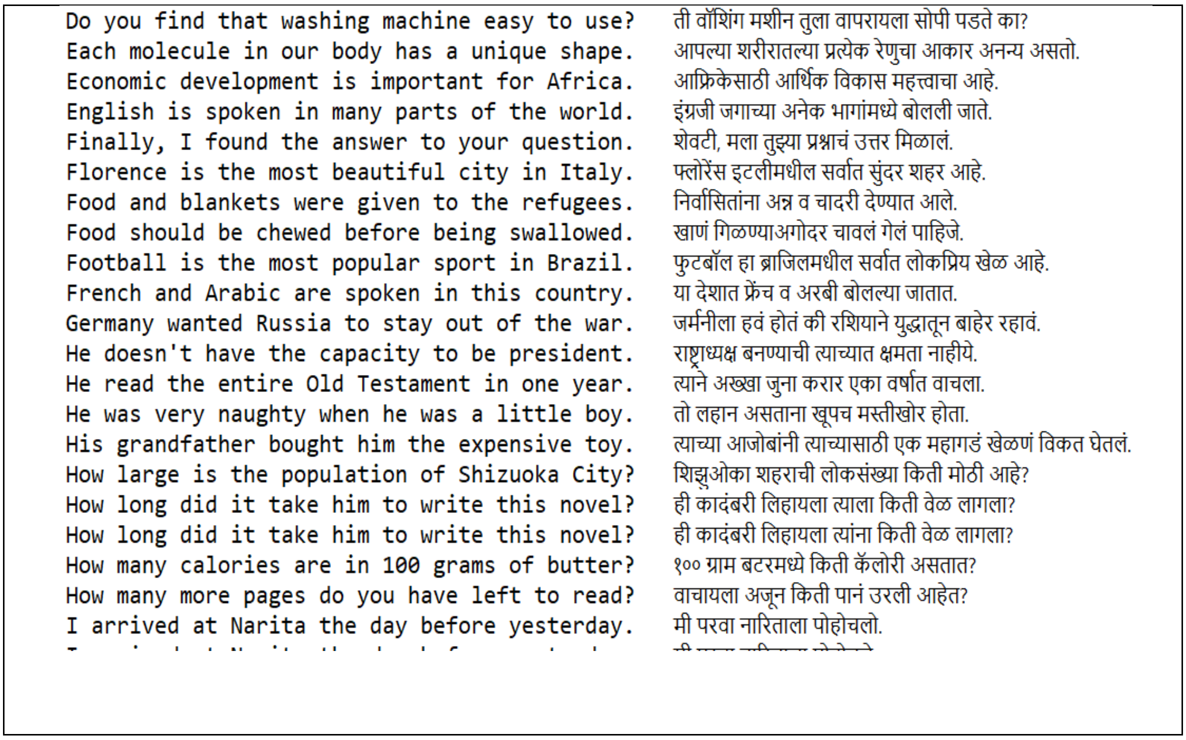


१(1)	आदौ वाद आसीत् स च वाद ईश्वरेण सार्धमासीत् स वादः स्वयमीश्वर एव।	[Eng.1]	In the beginning was the Word, and the Word was with God, and the Word was God.
२(2)	स आदावीश्वरेण सहासीत्।	[Eng.2]	The same was in the beginning with God.
३(3)	तेन सर्व्वं वस्तु ससृजे सर्व्वेषु सृष्टवस्तुषु किमपि वस्तु तेनासृष्टं नास्ति।	[Eng.3]	All things were made by him; and without him was not any thing made that was made.
४(4)	स जीवनस्याकारः, तच्च जीवनं मनुष्याणां ज्योतिः	[Eng.4]	In him was life; and the life was the light of men.
५(5)	तज्ज्योतिरन्धकारे प्रचकाशे किन्त्वन्धकारस्तन्न जग्राह।	[Eng.5]	And the light shineth in darkness; and the darkness comprehended it not.
६(6)	योहन् नामक एको मनुज ईश्वरेण प्रेषयाञ्चक्रे।	[Eng.6]	There was a man sent from God, whose name was John.

Figure 3.3 Parallel translation of verses of the Bible in Sanskrit and English

- http://www.manythings.org/anki/

  This web site is non-commercial and freely available for educational purpose Marathi-to-English translated (side by side) dataset was downloaded from this website.



```
Do you find that washing machine easy to use?	ती वॉशिंग मशीन तुला वापरायला सोपी पडते का?
Each molecule in our body has a unique shape.	आपल्या शरीरातल्या प्रत्येक रेणुचा आकार अनन्य असतो.
Economic development is important for Africa.	आफ्रिकेसाठी आर्थिक विकास महत्त्वाचा आहे.
English is spoken in many parts of the world.	इंग्रजी जगाच्या अनेक भागांमध्ये बोलली जाते.
Finally, I found the answer to your question.	शेवटी, मला तुझ्या प्रश्नाचं उत्तर मिळालं.
Florence is the most beautiful city in Italy.	फ्लोरेंस इटलीमधील सर्वात सुंदर शहर आहे.
Food and blankets were given to the refugees.	निर्वासितांना अन्न व चादरी देण्यात आले.
Food should be chewed before being swallowed.	खाणं गिळण्याअगोदर चावलं गेलं पाहिजे.
Football is the most popular sport in Brazil.	फुटबॉल हा ब्राजिलमधील सर्वात लोकप्रिय खेळ आहे.
French and Arabic are spoken in this country.	या देशात फ्रेंच व अरबी बोलल्या जातात.
Germany wanted Russia to stay out of the war.	जर्मनीला हवं होतं की रशियाने युद्धातून बाहेर रहावं.
He doesn't have the capacity to be president.	राष्ट्राध्यक्ष बनण्याची त्याच्यात क्षमता नाहीये.
He read the entire Old Testament in one year.	त्याने अख्खा जुना करार एका वर्षात वाचला.
He was very naughty when he was a little boy.	तो लहान असताना खूपच मस्तीखोर होता.
His grandfather bought him the expensive toy.	त्याच्या आजोबांनी त्याच्यासाठी एक महागडं खेळणं विकत घेतलं.
How large is the population of Shizuoka City?	शिझुओका शहराची लोकसंख्या किती मोठी आहे?
How long did it take him to write this novel?	ही कादंबरी लिहायला त्याला किती वेळ लागला?
How long did it take him to write this novel?	ही कादंबरी लिहायला त्यांना किती वेळ लागला?
How many calories are in 100 grams of butter?	१०० ग्राम बटरमध्ये किती कॅलोरी असतात?
How many more pages do you have left to read?	वाचायला अजून किती पानं उरली आहेत?
I arrived at Narita the day before yesterday.	मी परवा नारिताला पोहोचलो.
```

Figure 3.4: Parallel translated texts in English and Marathi

3.4 Data Pre-processing

The third stage of the CRISP methodology is data pre-processing(Also called data preparation). It is known to be the most time-consuming stage for any project implementing CRISP-DM. Since most of the data we deal with in real-time, as well as almost all text mining projects, are raw and lack certain structure, data pre-processing needs to be performed to format the data into an appropriate form which can be accepted as an input in learning algorithms.

Standard data pre-processing involves a set of operations include cleaning, vectorization, formatting and many more.

The following steps were applied to all three datasets collected during the data understanding phase to format data for language processing model.

- **Data cleaning**

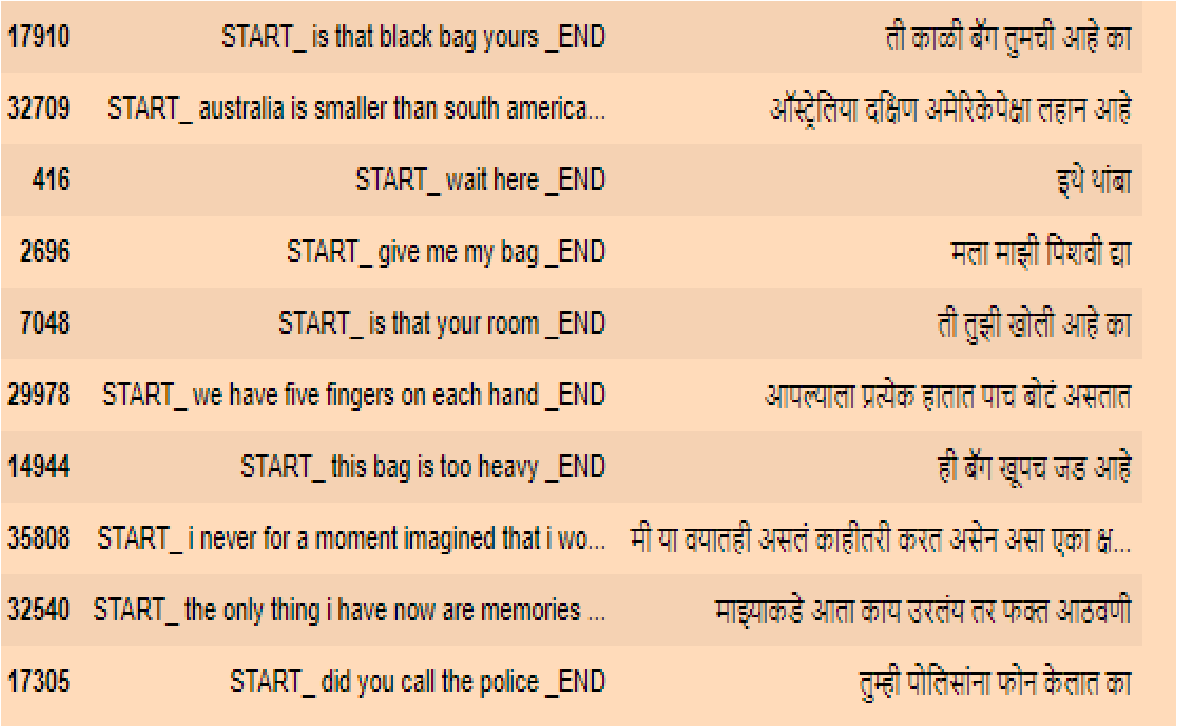
  Natural language processing (NLP) is a time and memory consuming processing which makes cleaning even more important.

  - Lowercasing all characters
  - All characters are converted to lowercase. Having both “friend” and “Friend” in our dataset doesn’t help the model and increases redundancy which in turn increases memory consumption.
  - Removing quotes:
  - Removing the set of all special characters
  - Remove all numbers from text
  - Remove extra white spaces

  Cleaning ensures that we dismiss irrelevant information and optimise memory consumption during the training of language model.

- **Tokenization:**

  As part of tokenization process, the following tasks have been performed:
  - Sentences and phrases present in data has been split into words.
  - Start and end tokens to target sequences was added.



17910	START_ is that black bag yours _END	ती काळी बॅग तुमची आहे का
32709	START_ australia is smaller than south america...	ऑस्ट्रेलिया दक्षिण अमेरिकेपेक्षा लहान आहे
416	START_ wait here _END	इथे थांबा
2696	START_ give me my bag _END	मला माझी पिशवी द्या
7048	START_ is that your room _END	ती तुझी खोली आहे का
29978	START_ we have five fingers on each hand _END	आपल्याला प्रत्येक हातात पाच बोटं असतात
14944	START_ this bag is too heavy _END	ही बॅग खूपच जड आहे
35808	START_ i never for a moment imagined that i wo...	मी या वर्षातही असलं काहीतरी करत असेन असा एका क्ष...
32540	START_ the only thing i have now are memories ...	माझ्याकडे आता काय उरलंय तर फक्त आठवणी
17305	START_ did you call the police _END	तुम्ही पोलिसांना फोन केलात का

Figure 3.5: Pre-processed data sample for English and Marathi

3.5 Data Modelling

Data modelling is the fourth phase of the CRISP-DM methodology. This phase consists of exploring different possible modelling techniques, deciding evaluation metrics, and most importantly creating the models. Data modelling involves three tasks: training an algorithm to predict the labels from the features, tuning the trained model to the objective of the project, and validating it on test data. For this project, the seq2seq model was selected for training the dataset.

3.6 Deep learning and Recurring neural network (RNN)

Deep learning a subset of machine learning in artificial intelligence that make networks capable of learning from unstructured data due to the stacked-layer architecture of neural networks. Learning can be supervised, semi-supervised or unsupervised.
Humans don't start interpreting from scratch every time. We make use of already learned languages, information, and experiences to contemplate anything new we are presented with. Traditional neural networks lack that ability which is a serious drawback. Recurring neural networks address this problem.

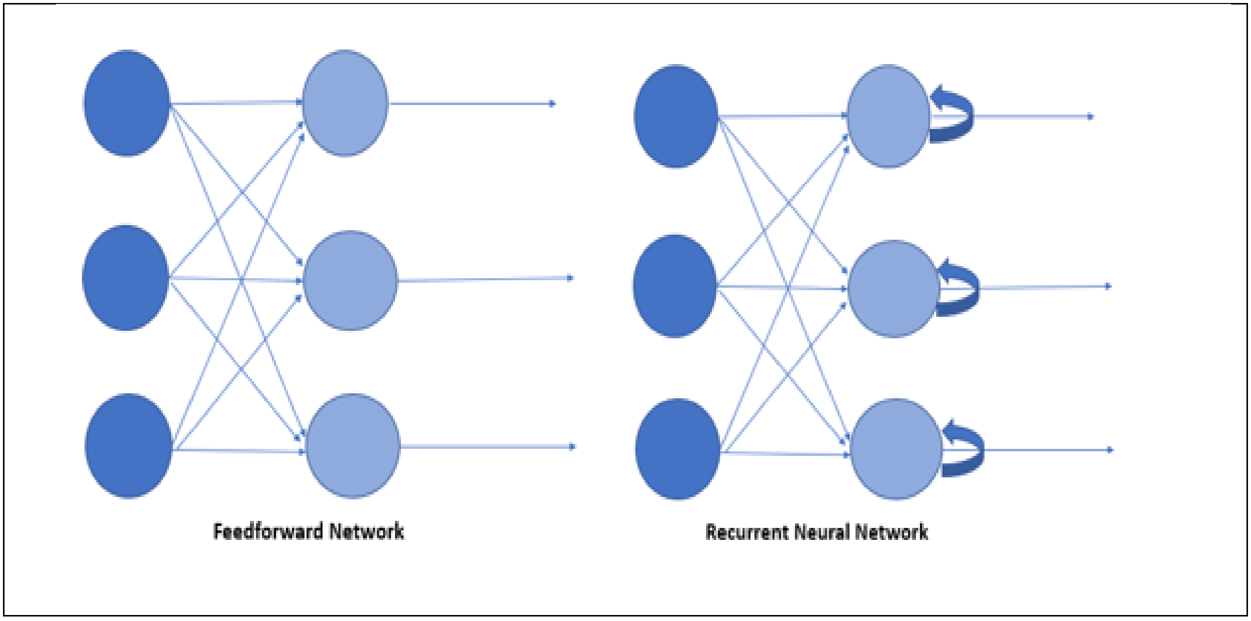


Figure 3.6: Feedforward and recurrent neural network

Underlying structure of Feed Forward Neural Network and recurrent neural network(RNN) is quite similar except feedback between nodes present in RNN. This is the primary difference between feedforward and RNN. These feedbacks, whether from output to input or self-neuron make RNN an excellent choice. Still, RNN has its own challenges and capturing long-term dependencies has been one of them for long (Hochreiter, 1991; Bengio et al., 1994; Hochreiter, 1998)

3.7 Long short-term memory (LTSM)

LSTM was proposed in 1997 by Sepp Hochreiter and Jürgen Schmidhuber. LSTM networks are a type of recurrent neural network as a solution to the two major drawbacks of RNN: vanishing gradient and exploding gradient. This memory cell can maintain information in memory hence enabling LSTM networks to learn long-term dependencies.
LSTM has underlying structural units called gates. These gates (input, forget, and output) regulate the ability of LSTM to protect and control cell state and provide options to limit the flow of information. The input gate is responsible for updating the current state using the input. Forget gate manages the information obtained from the previous state. The output gate decides whether the information should be passed to the next state or not.
These gates, implemented with element-wise multiplication by sigmoid layer, are in the range zero to one. A value of zero indicates “let nothing through,” while a value of one indicates “let everything through!”. Output of sigmoid layer in LSTM block or pass information depending on the strength. The LSTM unit has separate input and forget gates, while the GRU performs both operations together via its reset gate.

3.8 Gated recurrent unit (GRU)

Gated recurrent unit (GRU) was introduced in 2014 by Kyunghyun Cho et al. to serve as a gating mechanism in recurrent neural networks, The GRU is similar to long short-term memory (LSTM) in terms of underlying architecture and objective except that GRU lacks an output gate and has fewer parameters in forget gate.

GRUs have been shown to outperform LSTM on certain smaller datasets. However, as shown by Gail Weiss & Yoav Goldberg & Eran Yahav, the LSTM is "strictly stronger" than the GRU due to its ability to perform unbounded counting, while is not achievable in case of GRU. That explains the capability of LSTM to learn simple languages, unlike GRU. As established by Denny Britz, Anna Goldie, Minh-Thang Luong, Quoc Le of Google Brain, LSTM cells perform significantly better than GRU cells in the first large-scale analysis of architecture variations for NMT.

3.9 Encoder-Decoder Architecture

Encoder-Decoder Architecture is primarily partitioned into two parts, the encoder, and the decoder. The encoder is responsible to encode the inputs into a sequence which comprised of several layers. Then the sequence generated by the encoder layer is passed to the decoder. The decoder then generates the output sequence which, again, contain multiple layers.
Encoder and decoder are essentially recurrent neural networks. In machine translation, the encoder transforms a source sentence, e.g. “I want to read a book”, into a sequence, e.g. a vectorised representation to capture semantic information. The decoder then uses this state to generate the translated target sentence, e.g. “Je veux lire un livre” In French.

The encoder-decoder based neural machine translation (NMT) models (Sutskever et al., 2014; Cho et al., 2014) have seen tremendous success and development in recent time primarily due to the fact that NMT doesn’t require extensive domain knowledge and is simple in its fundamental concepts. Sutskever et al. (2014) proposed to encode the source language text as a fixed-length vector and then the decoder provides the target text based on this fixed-length vector. Encoder and decoder are recurrent neural networks (Sutskever et al., 2014) or their variants (Chung et al, Bahdanau et al., 2014). The fixed-length vector in this architecture plays as a mediator for source and target to work seamlessly with each other. NMT model works on a simple objective that the model should take a sentence in one language in the form of input and provide a translated version of input text as output.

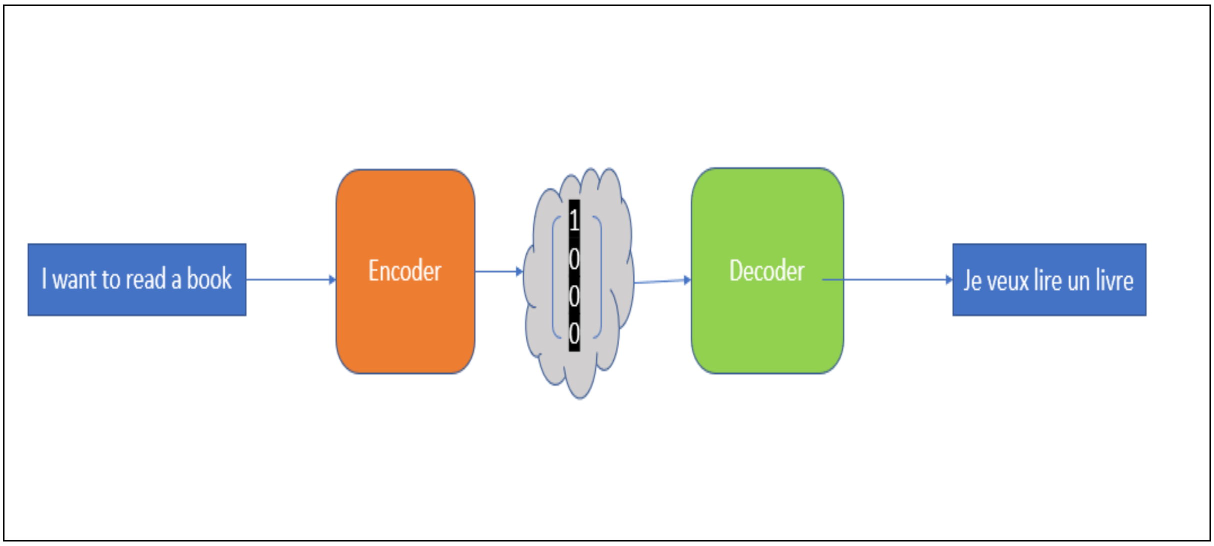


Figure 3.7: Encoder-decoder architecture

3.10 Developing the seq2seq model

Sequence to Sequence (Seq2seq) model was introduced by Google in the year 2014 to map a fixed-length input with a fixed-length output where the length of the input and output are not constant. There are multiple real-time applications of sequence to sequence model. For instance, seq2seq model is implemented in applications like Google Translate, online chatbots, and voice-recognition systems.

Seq2seq primarily has two components i.e. encoder and decoder, and hence it is also known as Encoder-Decoder network. Seq2seq takes as input a sequence of phrases/text and generates an output sequences pf phrases/text. Seq2seq uses the recurrent neural network (RNN) as a fundamental structure to be able to do so. Due to significant benefits of GRU or LSTM over RNN, vanilla version of RNN isn’t used in most of the cases. In this case, both encoder and decoder are LSTM models.

Before discussing the Encoder-Decoder structure, which is the most commonly used algorithm in NMT, it is important to understand how and why it is related to the biggest challenge in machine translation of languages. Since we are dealing with unstructured text in machine translation of languages, unstructured data is required to be transformed to a certain format which is appropriate as input for machine learning models.
Technically, we need to convert textual data to numeric format. Numeric format is achieved by converting each word into a vector. Vector representation of words, in Seq2seq, is facilitated by using Context vector. Context vector, also known as thought vector, has a pre-decided fixed-length. It contains information about sentence embedding. This information is the final hidden state of the encoder. Encoder processes the input sequence and transforms the sequence into a context vector. This representation summarizes the semantic understanding of the whole input text. Output of context vector serves as the input for the decoder which then generates the output.

Sequence to sequence learning (seq2seq) models the conditional probability p(y|x) of mapping an input sequence into an output sequence. The idea behind the concept of conditional probability is that input sequence influences output sequence in a statistical way. Encode, which is basically a recurrent neural network (RNN), generates a vector representation of the input sequence.

Decoder, which is also a recurrent neural network (RNN), generates an output sequence taking into account one unit at once. Usually, conditional probability is calculated as follows:

$$\log p(y|x) = \sum_{j=1}^{m} \log p\left(y_j | y_{<j}, x, \boldsymbol{s}\right)$$

Where the input sequence is x1,. . . , xn , output sequence is y1,. . . , ym

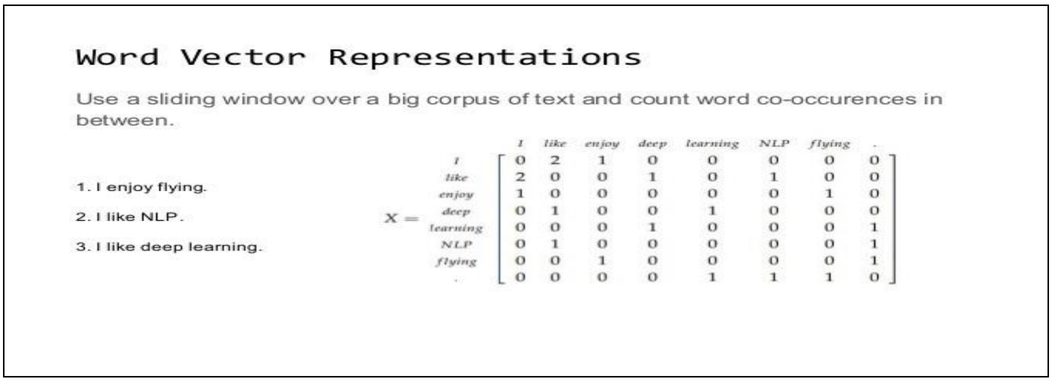
The model consists of 3 parts: encoder, intermediate (encoder) vector and decoder.

Encoder

Encoder picks the provided input data (Natural text) and passes it to the last state (intermediate vector) after processing. Input for encoder in this scenario is plain English sentences. We add appropriate weights (Separate file has been provided for that in code) to the previously hidden state *h_(t-1)* and the input vector *x_t.*
The encoder is made up of the following layers:

- **Input Layer:** Input layer, being the first layer of the encoder takes the English sentence as input and passes it to the embedding layer after processing.

- **Embedding Layer:** Embedding layer takes the output of the input layer as input and converts each word to fixed-size vector.



Word Vector Representations

Use a sliding window over a big corpus of text and count word co-occurences in between.

1. I enjoy flying.
2. I like NLP.
3. I like deep learning.

$$X = \begin{array}{c} \\ I \\ like \\ enjoy \\ deep \\ learning \\ NLP \\ flying \\ . \end{array} \begin{array}{cccccccc} I & like & enjoy & deep & learning & NLP & flying & . \\ 0 & 2 & 1 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{array}$$

Figure 3.8: Word vector representation (source: https://bracketsmackdown.com/word-vector.html

- **First LSTM Layer:** This layer takes a vector that represents a word and passes the output to the next layer at every time step.

- **Second LSTM Layer:** This layer works like the previous layer, but rather than passing its output, it passes its states to the decoder.

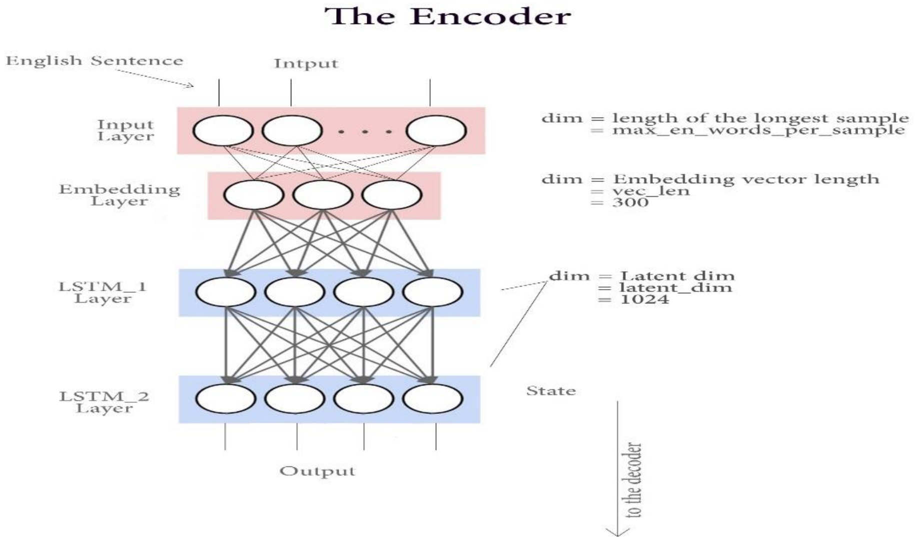

The Encoder
English Sentence
Intput
Input Layer
dim = length of the longest sample = max_en_words_per_sample
Embedding Layer
dim = Embedding vector length = vec_len = 300
LSTM_1 Layer
dim = Latent dim = latent_dim = 1024
LSTM_2 Layer
State
Output
to the decoder

Figure 3.9: Encoder architecture (Source https://towardsdatascience.com/)

For the seq2seq model developed as part of this project, latent_dim has value of 50**, the** length of the longest input layer sequence token is 35 for both Marathi and Sanskrit text.

Encoder Vector

- This is the last and only hidden state from the encoder end of the seq2seq model.
- This vector aims to have the summarized information for all input sequences so that the decoder can make correct predictions.
- This vector works as the initial hidden state of the decoder of the model.

Decoder

Decoder generates an output which can't be read directly by humans. To predict the next word in the sequence, we set the initial states to the states from the previous time step and populate the first character of the target sequence with the start character.

$$h_t = f(W^{(hh)} h_{t-1})$$

Where y_t is output of decoder, t is the time step t, W(S) is the weights assigned, and h_i is the hidden state.

The output *y_t* at time step *t* is computed using the formula:

$$y_t = softmax(W^S h_t)$$

Softmax, in seq2seq model, is used to generate a probability vector which is crucial to decide the final output. The potential of this model lies in the fact that sequences of different lengths can be mapped to each other without any hassle.

Before discussing all layers present in the decoder, It's important to understand that seq2seq can't convert each English sentence into Sanskrit/Marathi in one-time step. The translated sentence is achieved in multiple time steps which equals the number of words present in the longest English sentence. For e.g., if there are 15 words in the longest English sentence, 15-time steps would be required to achieve its Sanskrit/Marathi translation.

Like the encoder layer, the decoder is also made up of multiple layers:

- **Input Layer:** Input layer of the decoder takes the Sanskrit/Marathi sentence and pass it to the embedding layer.
- **Embedding Layer:** Embedding Layer takes the Sanskrit/Marathi sentence as input and convert each word of the sentence to the fixed-size vector.
- **First LSTM Layer:** This layer takes a vector generated by the embedding layer that represents a word and passes its output to the second LSTM Layer.
- **Second LSTM Layer:** This layer processes the output by first LSTM layer and passes its output to the last layer of decoder i.e. dense layer.

- **Dense Layer:** Takes the output from the second LSTM layer and outputs a one-hot vector representing the target Sanskrit/Marathi word.

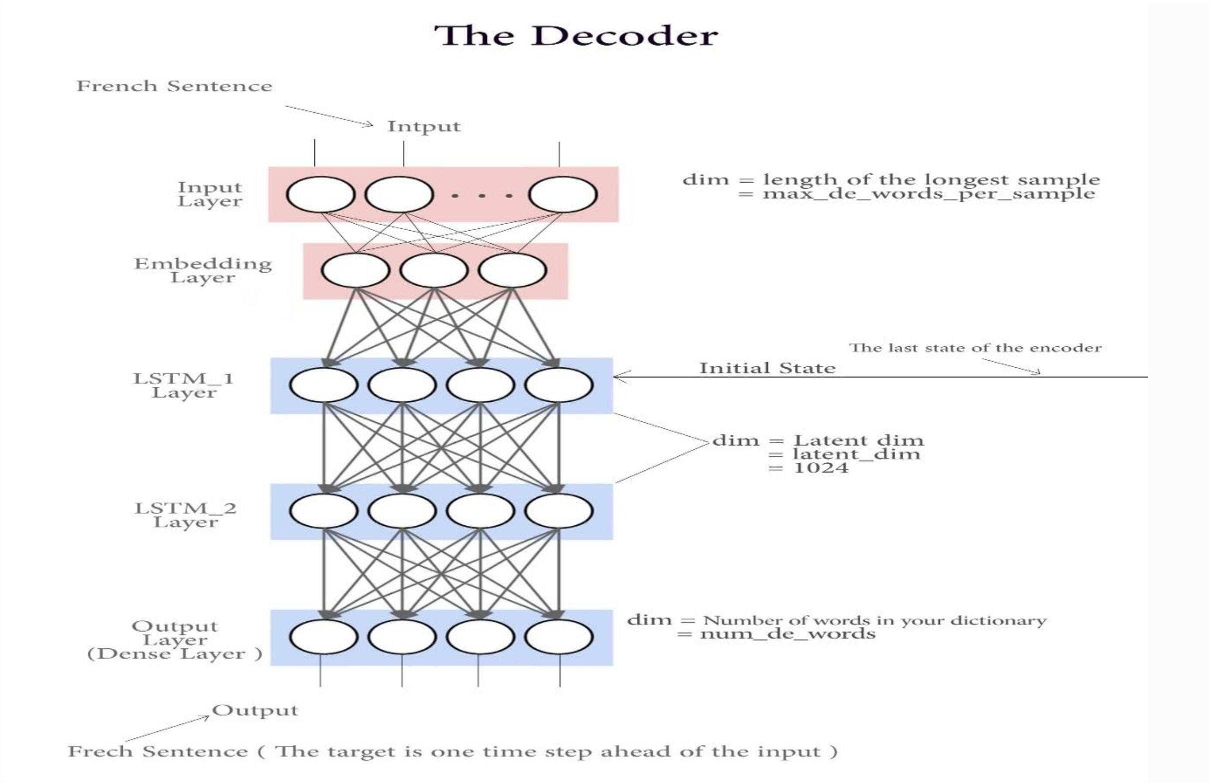

The Decoder
French Sentence
Intput
Input Layer
dim = length of the longest sample
= max_de_words_per_sample
Embedding Layer
The last state of the encoder
Initial State
LSTM_1 Layer
dim = Latent dim
= latent_dim
= 1024
LSTM_2 Layer
Output Layer (Dense Layer)
dim = Number of words in your dictionary
= num_de_words
Output
Frech Sentence (The target is one time step ahead of the input)

Figure 3.9: Decoder architecture

3.11 Parameterization

Seq2Seq accepts several parameters that affect training and evaluation, which are as follows:

▪ **Max length:** Max Length of source sequence for our model is 35 whereas max length of target sequence is 38. Marathi. For Sanskrit 35, 36

• **Latent_dim:** Embedding layer accepts latent_dim as a parameter which has the value of 50 for model developed as part of this project. Latent_dim define the Latent dimensionality of the encoding space of se2seq model.

•**Optimizer**: An optimizer is one of the arguments taken to compile the model. Keras library provides multiple options for optimizers like Stochastic gradient descent (SGD), Adagrad, Adadelta, Adam, rmsprop etc. We have selected “rmsprop “optimizer” to compile our model since it is the recommended optimizer for recurrent neural networks (RNN). The purpose of using optimizer in RNN is generate faster and better outcome by updating the model parameters such as weights.

• **Loss:** Loss function, also called as objective function or optimization score function, helps us understand the difference between expected and actual output in neural networks. Loss, like optimizer, is one of the parameters required to compile a model. Keras provides multiple loss function such as: mean_squared_error, mean_absolute_error, mean_absolute_percentage_error, mean_squared_logarithmic_error, hinge, logcosh, categorical_crossentropy etc. we have compiled our model using the categorical_crossentropy loss function.

• **Metrics**: Metric is a function that is used to calculate the performance of your model. Metrics, like loss and optimizer, is also one of the parameters we provide to model while compilation. We have used accuracy to understand the performance of the model.

• **Batch Size:** Batch size is a number of samples processed before the model is updated. The size of a batch has to be at least of the same size of number of samples in the training dataset. Size of batch is our model is 128.

• **Epochs**: The number of epochs is the number of times training dataset is processed on the model. Usually, setting a higher number of epochs improves the quality of the model. However, having too many epochs not only slows down the process of training model but also inversely affects the vectors representation of training data. Also, excessive number of epochs can get the model to get overfit. Different number of epochs has been used in different experiments.

Chapter 4
Evaluation of Machine Translation

4.1 Evaluation overview

Evaluation is the fifth phase of the CRISP-DM methodology. This is a critical step in which the results of machine translation model were evaluated. To analyse the results, the evaluation metrics that were used should be described in more detail.

4.2 Evaluation metrics

Manual evaluations of machine translation tasks are extensive but also very expensive. Human evaluations can significant amount of take and labour to get the result and it cannot be reused. There are existing automatic machine translation evaluation techniques that are independent of language, faster, and less expensive.
The most commonly used evaluation metric is accuracy, which is the summary of all instances that are correctly predicted. Code snippets used to calculate all evaluation metrics is presented in Appendix B.

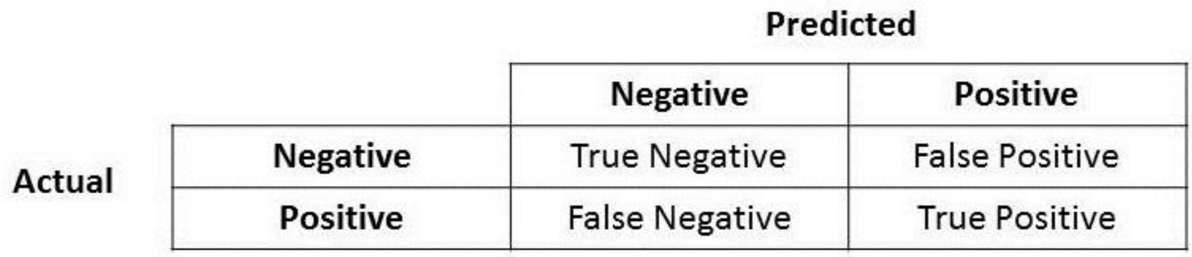
4.2.1 Accuracy:

Accuracy is the simplest and most commonly-used metric in any data science project. It is defined on the basis of understanding and measurement of relevance. Accuracy is usually used along with precision and recall. Precision, also called positive predictive value, is the fraction of relevant outcomes over the retrieved outcomes, while recall, also known as sensitivity, is the fraction of retrieved relevant outcomes over the total number relevant outcomes.

Accuracy, precision, and recall are defined as follows:

Accuracy = (TP+TN)/(TP+FP+FN+TN)
Precision = TP/(TP+FP)
Recall = TP/(TP+FN)

where T is true, F is false, P is positive, and N is negative.



		Predicted	
		Negative	Positive
Actual	Negative	True Negative	False Positive
	Positive	False Negative	True Positive

Figure 4.1: Accuracy

Along with calculating accuracy, we also explore other evaluation metrics available.

4.2.2 Bilingual evaluation understudy (BLUE)

Bilingual evaluation understudy (BLUE) is one the most common evaluation metrics used for machine translation models. It was developed by Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu at IBM in 2002. It's a quite popular metric in natural language processing. It is used for model that generates a text string (Translated text of a natural language) rather than a classification class. BLUE is used as a solution to find a solution to assign a numerical score to a translation that explains the quality of translation.
The BLEU metric ranges from 0 to 1.

However, BLEU is not a very reliable evaluation metric for machine translation systems because of its incapability to consider meaning. It simply doesn't align with the idea of machine translation which is to accurately understand the context of a text in the original language and provide the output target in target language accordingly. Some syntactic or grammatical anomaly is usually acceptable as long as the actual meaning of text in original language remains intact in the translated text. BLEU works on n-grams based system where it rewards the system for having exact match. That implies that a difference in a function word (for e.g. "an" or "on") is as inaccurate as the difference in words having much more importance to overall context of the sentence. It also means that BLEU doesn't have the ability to handle synonyms even when used in perfectly valid context and BLEU penalizes the system. BLEU could be a metric in case evaluation need to be performed across an entire corpus and we understand and accept the limitations of BLEU.

To address these existing issues, two popular variations of BLEU were designed and introduced.

- **NIST :** NIST (US National Institute of Standards and Technology), like BLEU, works based on n-grams but weights for n-grams are based on their rareness which means that when a rare n-gram is matched in reference system, score is improved better than matching a common n-gram.

- **Recall-Oriented Understudy for Gisting Evaluation (ROUGE):** ROUGE is a variation of BLEU which is more focussed on recall rather than precision. Rather than finding the number of n-grams not appearing in the reference translation, it looks for the n-grams that are present in the output.

There are a large number of other methods that can be used to evaluate sequence-to-sequence models. Some of them are based on approaches adopted from other areas of NLP.

- **Perplexity:** The concept of Perplexity has been adapted from information theory. It is a measure to learn how efficiently learned probability distribution matches with the words on the input text in the original language. more often used for language modelling text. Low perplexity score indicates that translation is more accurate and reliable.
- **Word error rate (WER):** Word error rate(WER) is a commonly-used evaluation metric in machine translation models. It calculates the number of insertions, deletions, and substitutions ("an" for "the") in the output sequence for a given reference.

The notion behind 'insertion' and 'deletion' ' is all about the context of references in the used hypothesis. For e.g., for reference "It is raining" and hypothesis "It _ raining", we say that deletion is being performed in WER.

Word error rate is computed as:

$$WER = (I+D+S)/ N$$

Where I is the number of insertions, D is the number of deletions, S is the number of substitutions, N is the total number of words in the reference (N=S+D+C), and C is the number of correct words.

There are other evaluations metrics which were developed specifically for sequence to sequence modelling tasks. Table summarizes such evaluation metrics as mentioned below:

Evaluation Metric	**Functionality**
STM (subtree metric)	STM compares the sentences/phrases for the reference and output translations. Output with different syntactic structures is penalized.
METEOR	Meteor is similar to BLEU but includes few more functionalities, like comparing the stems of words (“going” and “goes” would be considered as matches) and considering synonyms.
TER	TER considers stemming, paraphrases, and synonyms for evaluation.
Error rate	Error rate works based on the number of changes required to transform the original output translation into a natural and readable translation.
LEPOR	Lepor was specifically developed to provide more information for Asian languages like Chinese and Japanese.
MEWR	This is the most recent metric on the list. MEWR doesn’t require reference translations which makes it a good choice for under-resourced languages. MEWR uses a combination of word and sentence embeddings and perplexity.

Table 4.1 Evaluation Metrics

Any natural-language is complex for models to process, which means that evaluation translation of language is a complex task. It is often stated that developing and selecting the right evaluation metrics for natural language generation might be the toughest task in NLP. Earlier, the human evaluation used to be the standard in machine translation and though we

want to achieve complete automated evaluation, there is still a place for manual intervention in translation tasks.
However, human evaluation is expensive, time-consuming, and can’t be reused. Apparently, a combination of human evaluation and at least one automatic evaluation metric might be a good evaluation metric especially for system going in production.

.

4.3 Experiment results

This section presents the results of the evaluation of the machine translation model in this chapter. The experiments were conducted on two training and two testing datasets.
For training data: **Q1** means Marathi-to-English translation dataset having both training and test data whereas **Q2** means Sanskrit-to-English translation dataset having both training and test data. For both, **Q1** and **Q2**, the dataset has been divided into training and test data in 90:10 ratio. (26012, 4850 for Sanskrit and 32248, 3584 for Marathi)

4.3.1 Experiment results for Q1 dataset

Translation Text Language	Epochs	Accuracy	Loss
Marathi	200	0.90	0.67
Marathi	420	0.92	0.55

Table 4.2 Experiment results for Q1 dataset

Figure X shows the graph of loss vs epochs for training of Q1 dataset.

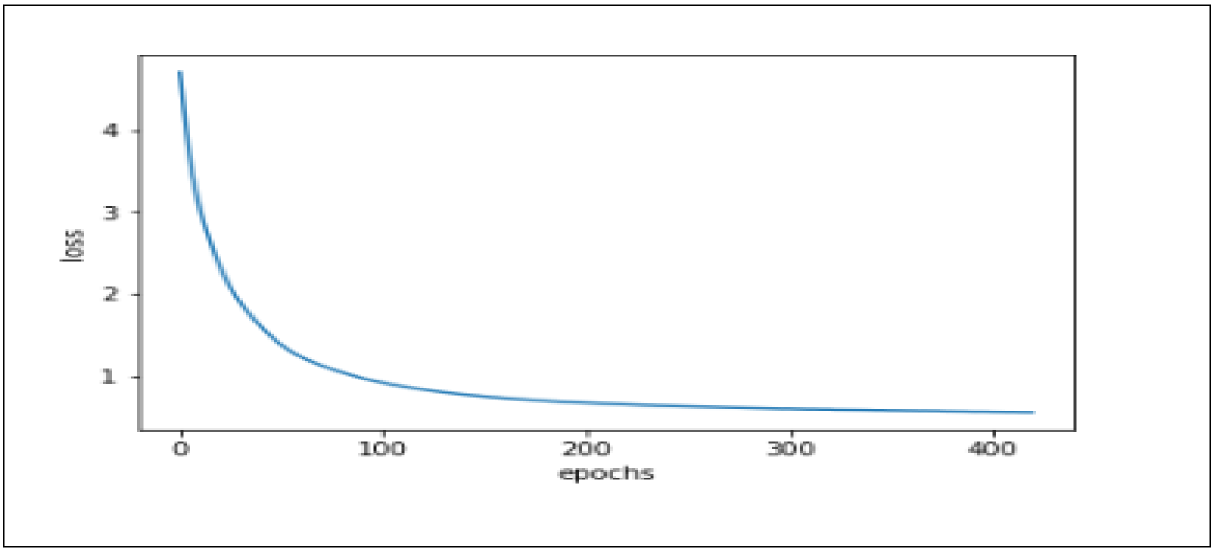


Figure 4.2: Loss vs. Epochs for Q1 dataset

As the figure X suggests, loss is decreasing exponentially as number of epochs go higher and model is getting better at understanding its task of translating Marathi text to English text.

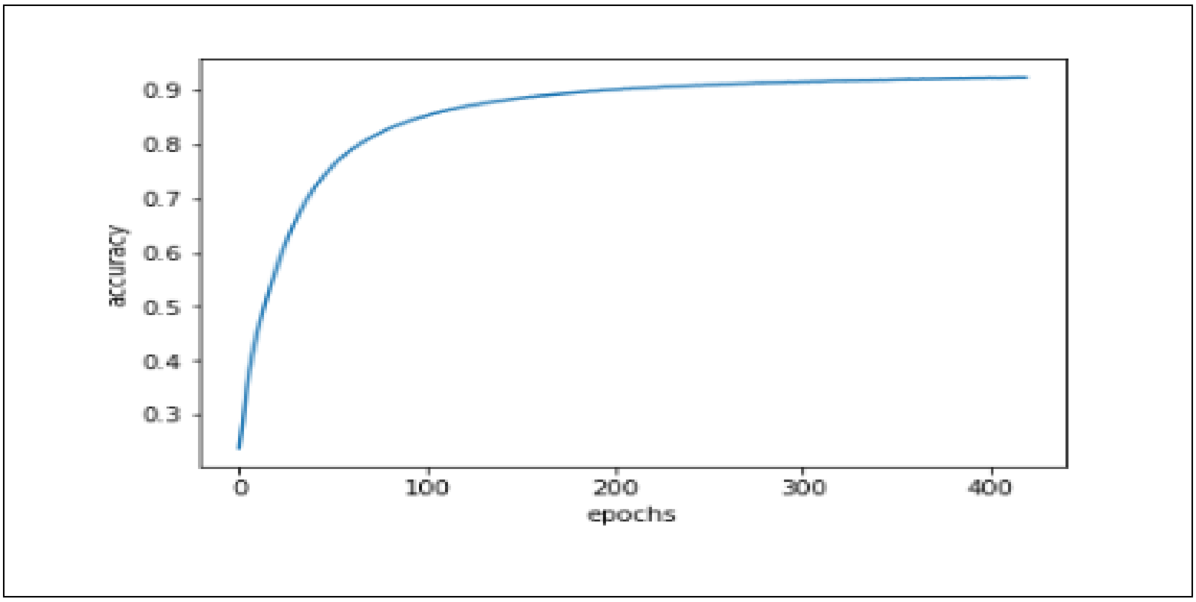


Figure 4.3: Accuracy vs. Epochs for Q1 dataset

As the figure X suggests, accuracy is increasing exponentially as the number of epochs go higher and model is getting better at understanding its task of translating Marathi text to English text.

4.3.2 Experiment results for Q2 dataset

Figure X shows the accuracy and loss data for the model developed for Sanskrit to English translation of text data.

Translation Text Language	Epochs	Accuracy	Loss
Sanskrit	**200**	**0.19**	**4.87**
Sanskrit	**420**	**0.37**	**3.79**

Table 4.3: Experiment results for Q2 dataset

Figure 4.4 shows the graph of loss vs epochs for training of Q1 dataset.

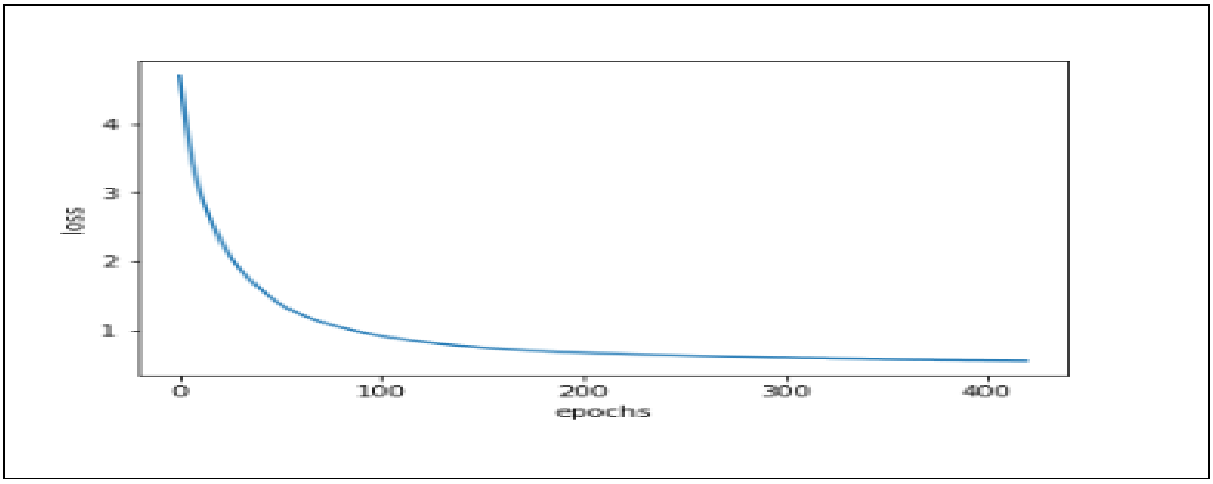


Figure 4.4: Loss vs. Epochs for Q2 dataset

As the figure 4.4 suggests, loss is decreasing exponentially as number of epochs go higher and model is getting better at understanding its task of translating Marathi text to English text.

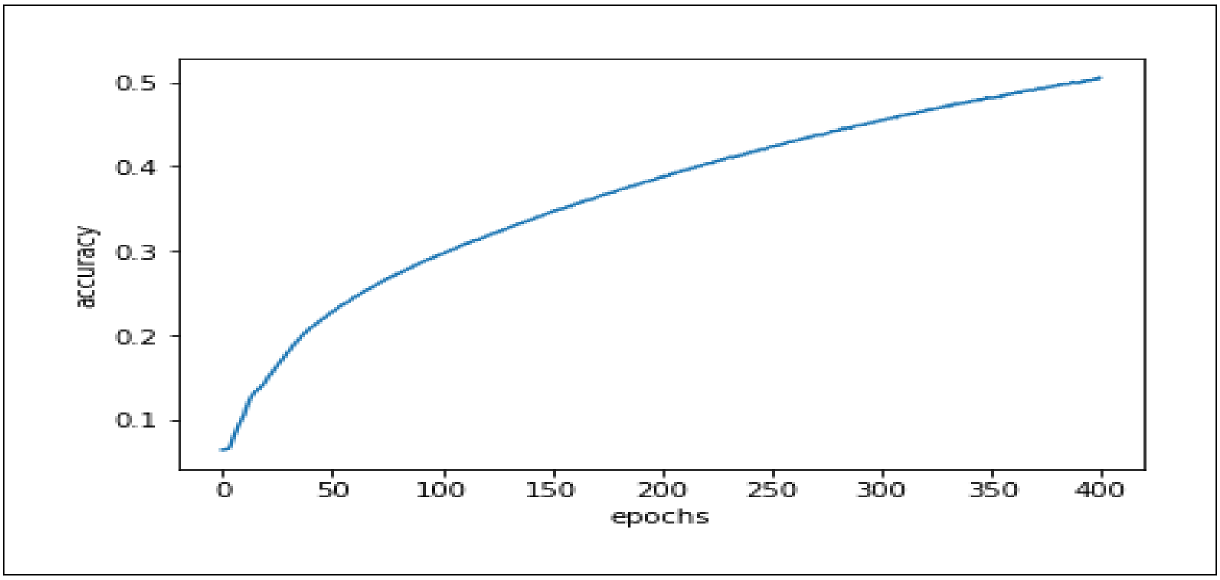


Figure 4.5: Accuracy vs. Epochs for Q2 dataset

As the figure 4.5 suggests, accuracy is increasing exponentially as number of epochs go higher and model is getting better at understanding its task of translating Marathi text to English text.

4.3.3 Additional experiment for Q1 dataset

Translation Text Language	Epochs	Mean_Absolute_Error	Loss
Marathi	100	3.6530e-04	1.8265e-04
Marathi	250	3.6530e-04	1.8265e-04

Table 4.3: Additional experiment for Q1 dataset

Figure 4.6 shows the graph of loss vs epochs for training of Q1 dataset.

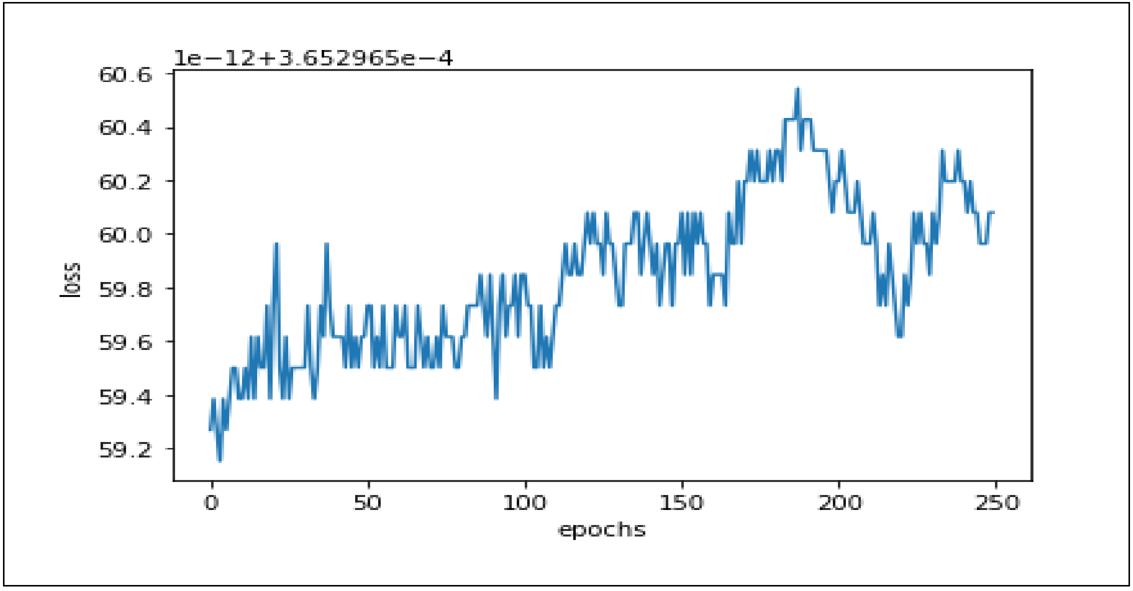


Figure 4.6: Loss vs Epochs for training of Q1 dataset (MSE)

As figure 4.6 suggests, loss is decreasing exponentially as the number of epochs go higher and the model is getting better at understanding its task of translating Marathi text to English text.

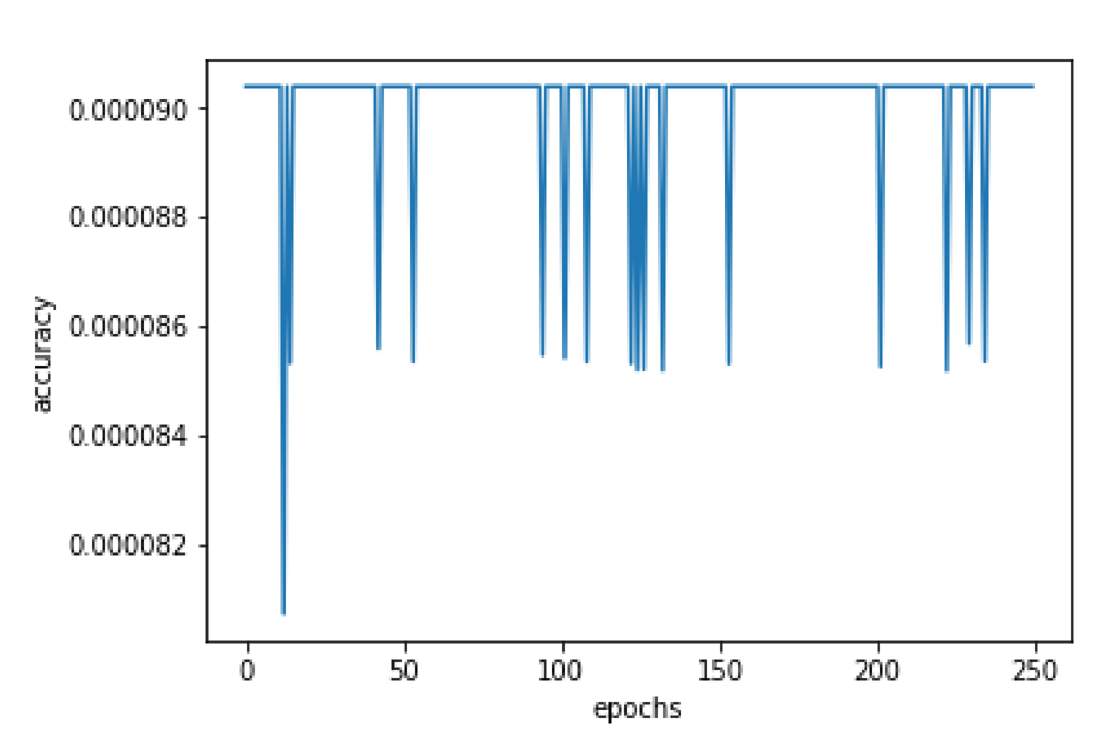


Figure 4.7: Accuracy vs Epochs for training of Q1 dataset(MSE)

As figure 4.7 suggests, accuracy is increasing exponentially as the number of epochs go higher and the model is getting better at understanding its task of translating Marathi text to English text. During the experiments performed for Q1 and Q2 datasets, it is evident that Same model does not perform well for Sanskrit dataset which is smaller than Marathi dataset. There can be two ways to improve the performance of model for Sanskrit dataset: collect bigger datasets and increasing the number of epochs. Though the accuracy of the model for Sanskrit needs improvement, we observe that accuracy grows exponentially with increasing number of epochs.

Chapter 5

Exploring translation beyond NMT

5.1 Overview

Machine translation is one of the most important as well as challenging tasks in the field of natural language processing. As discussed, we have multiple approaches available to build and evaluate a model for machine translation.
In the previous two chapters, the translation model was measured based on the accuracy of model following neural machine translation and word Embedding approach. This chapter describes the attempt to assess the output of seq2seq model manually.

5.2 NMT model evaluation

Generating output sequence in seq2seq is simple and inexpensive. However, any model is not completely accurate. In the case of under-resourced languages, it's even more challenging and manual evaluation is required to assess how "good" translated text is. It means that not only literal translation is reliable, but context and essence of input text should also be captured in output text of target language.
Natural languages are rich in terms of context, facts, information, belief, and emotions but some of the information doesn't hold any significant value (For e.g. stop words). In addition, it is extremely time-consuming and laborious to go through entire text data of output. However, we have tried to compare a few output texts to understand how well the model has been developed.

To get the overall idea of the developed model, the steps involved has been listed as follows:

- Installing and loading the required packages:
  First, the sklearn, keras, matplotlib, pandas, numpy packages was installed in Anaconda. Anaconda, along with these libraries provides an environment to build a NMT based on seq2seq model.

- Loading the text:
  The input data used were text files (Input text in natural languages i.e., Sanskrit, Marathi and English), so the data were loaded using pandas package.

- Cleaning the text:
  Tasks such as converting the text to lower case, removing quotes, excluding punctuation, numbers, and whitespaces were performed using lambda functions.

- Pre-processing of data:
  Splitting of every sentence in text data was split into words and then tokenization was performed. Training and test data ration (90:10) were set. We have 32248 samples of training data and 3584 samples of test data for Marathi, whereas 26012 samples of training data and 4850 samples of test data for Sanskrit.

- Generating batch of data, setting up encoder and decoder with LTSM.
- Setting model parameters:
  Model parameter like epochs, latent_dim, optimizer, loss, metrics, and batch size were set appropriately.

Accuracy indicates that the model doesn’t perform very well particularly for Sanskrit. For Marathi, results appear to be better in terms of accuracy and loss still we investigate output for both languages.

Table X shows that translation is capturing the meaning as well as synonym
(For e.g. “mad” and “angry”) for words present in input text.

Marathi Text	“ते माझ्यावर रागावले”
Expected English Translation	"He is mad at me"
Actual English Translation	"He got angry with me"
Sanskrit Text	"हे प्रियतमाः अस्मासु यदीश्वरेणैतादृशं प्रेम कृतं तर्हि परस्परं प्रेम कर्त्तुम् अस्माकमप्युचितं।"
Expected English Translation	"Beloved if God so loved us we ought also to love one another"
Actual English Translation	"Beloved if God so loved us we ought to be reserved unto you and to remember the word of god"

Table 5.1: Comparison and evaluation for output text

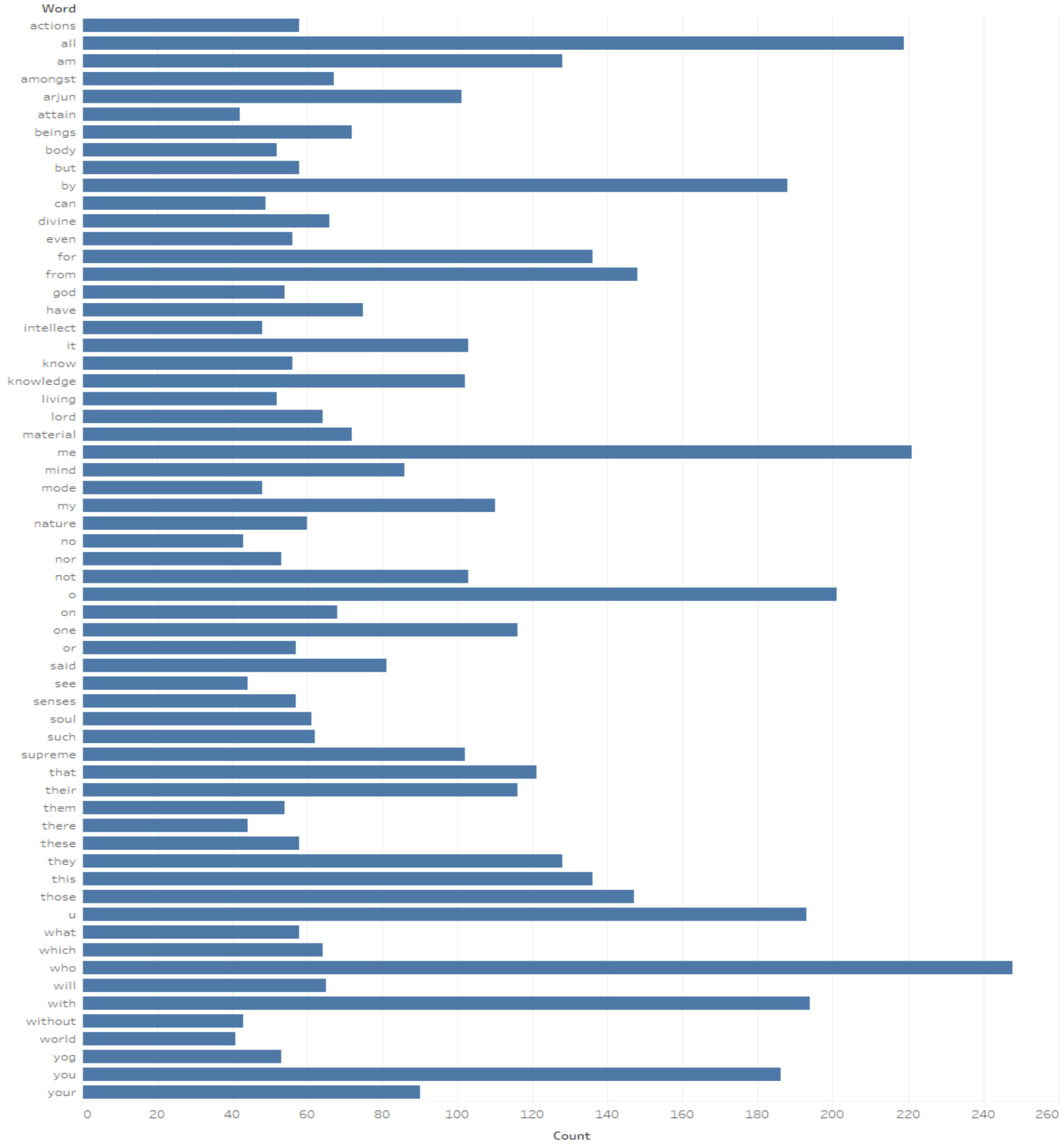
On the other hand, we have tried to find output texts that aren’t correctly translated by seq2seq model.

Marathi Text	“मी माझ्या आईला तिच्या वाढदिवसाला नेहमीच फोन करते”
Expected English Translation	" I always call my mother on her birthday"
Actual English Translation	“ I always call my mother on her house”
Sanskrit Text	"युष्माकं यथा वाधा न जायते तदर्थं युष्मान् एतानि सर्व्ववाक्यानि व्याहरं।।"
Expected English Translation	"These things have I spoken unto you that ye may be saved"
Actual English Translation	"These things have I spoken unto you that ye should not be offended"

Table 5.2: Comparison and evaluation for output text(2)

5.3 Visualisation

This section provides a visual representation of the most frequently used Sanskrit words(Taken from Bible translation) and Marathi dataset.
Word cloud has been used to present the frequent terms in both, Sanskrit and Marathi, which provides better visualization of words. Word cloud visualizations are engaging and simpler to be interpreted. They are also easy to share and engaging. In addition,
they help in understanding the context of a text by listing the most significant words found in the text.



Sum of Count for each Word. The view is filtered on Word, which excludes a, as, be, do and e.

Figure 5.1 Word frequency visualization

Figure 5.3 and Figure 5.4 show the word cloud for Marathi and Sanskrit respectively depicting the most frequent words appearing in dataset for both languages.
The significance of a word (i.e. Number of times a word is present in dataset) is directly proportional to the font size of the word in word cloud. The most frequently used words are displayed in bigger fonts.

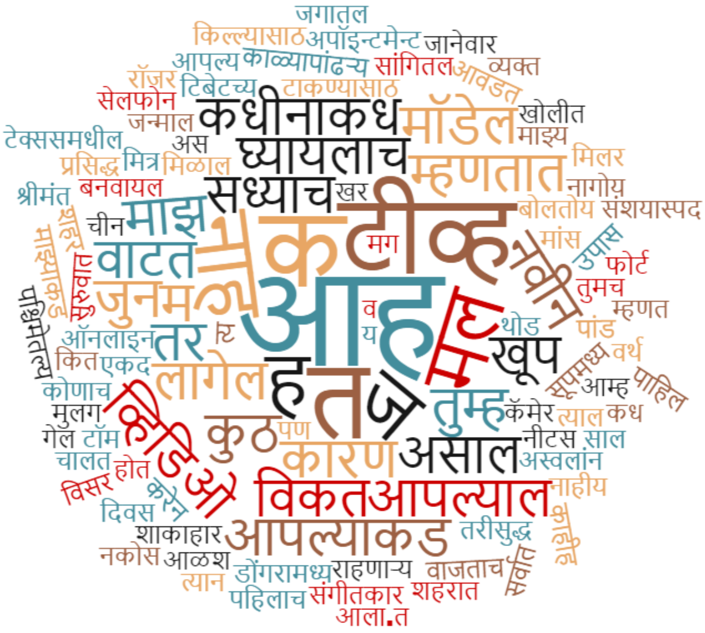


Figure 5.2 Marathi word cloud

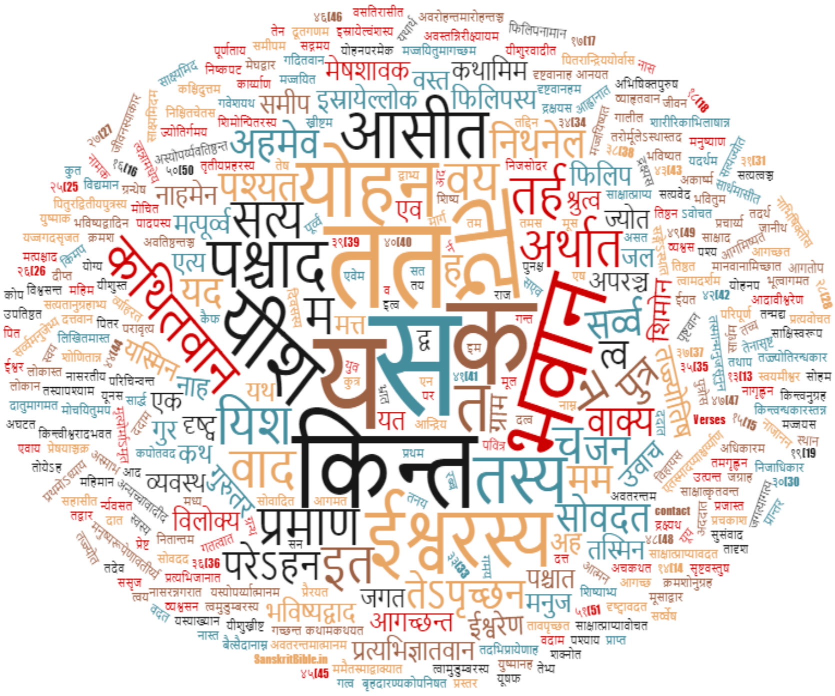


Figure 5.3: Sanskrit word cloud

Figure 5.2, 5.3, 5.4 present the pictorial representation of our dataset. Since the datasets used in this project consist of words from natural language, creating a frequency chart helps us understand how frequently a certain word is being used and if it is a dominant word in the dataset. Wordcloud is a standard way to visualize text dataset because it is aesthetic and easy for the viewer to understand.

Chapter 6

Conclusion

6.1 Overview

The purpose of this chapter is to discuss the project with regard to researcher's experience. First, we discuss the process involved in achieving aim for this aims and objectives discussed in chapter 1. Second, potential research and aspects of this study has been discussed and then this chapter has been concluded with personal reflection.

6.2 Challenges of under-resourced languages for MT

This research has been focussed on tasks like morphology, natural language translation, and machine translation, however, there is still a lot of work to be done while resources are scarce. Some of the biggest challenges that must be taken into account are small datasets, high dialectal variation, lack of technologies to support orthographic normalizations, lack of linguistic pre-processing tools.

This research for under-resourced languages try to achieve understanding of human language structures (For Sanskrit and Marathi specifically) and explore ways to build general computational model that can further be used or explored better. Through this work, we presented a review of approaches available for machine translation in natural language processing. However, these changes are amplified when dealing with under-resourced languages. Available digital resources and their related tools for Sanskrit and Marathi has been discussed as well. We live in a world with a rich linguistic diversity. Roughly, more than 6500 languages exist but many languages have not crossed the "digital divide", even in the age of technologies like smart keyboards with predictive text available in only 10% of overall languages. Meanwhile, only 23 languages account for more than half of the world's population. Google has been working hard to bring language technology to more languages and approximately 500 languages have been part of the process until now. This alone explains why under-resourced languages need attention and why it is quite challenging to work with such languages when approximately 40 % of languages have been considered endangered already.

Recently, NMT has seen tremendous progress in machine translation tasks and it looks much more promising than other machine translation models (For e.g. Phrase-based MT) with increasing data due to enormous amount of data being generated every single day on various platforms across the world. However, under low data situations (In case of low-resourced languages in particular), NMT is not as efficient.

Our study shows that NMT produces output unrelated to input in case of low data situation for Sanskrit. While NMT is commonly considered to be the most efficient model for machine translation tasks, it does poorly for rare or unseen test) data. Nature languages, in general, are complex and every word usually has multiple translations or interpretations. NMT systems are still extremely accurate when enough data is presented to train the model and improving ways or sources to collect dataset for under-resourced languages would certainly drastically improve the translation task.

6.3 Machine Translation Competitions and programs

The last couple of decades have witnessed some exciting improvements in the area of machine translation which has drawn a lot of attention of organisation and many machine translation competitions have taken place. In the year 2006, Tenjinno machine translation competition was held in association with the 8th International Colloquium on Grammatical Inference (ICGI 2006). The competition was organised to improve the current state-of-the-art in grammatical inference. Tenjinno was the successor to the earlier Abbadingo [1], Gowachin and Omphalos [2, 3] competitions. At the ICGI 2004, it was decided competition tasks should closely relate to a real-world application compared to earlier competitions; hence the task of machine translation was chosen. In October 2011, DARPA (Defence Advanced Research Projects Agency) initiated the Broad Operational Language Translation (BOLT) program so that new techniques for machine translation can be explored to be further used in text and speech common in online and in-person conversations.

6.4 Project Evaluation

The Bible, being the sacred book followed by billions of people across the world, is considered to be the respected source for value system, guidance, and wisdom. Representing the information and values from the sacred texts needs a lot of expertise and understanding even being translated by an expert who has the depth and understanding to appreciate the language as well as religious beliefs. Extracting same information by building learning algorithm is, understandably, way harder.
The results of this research project on text mining and machine translation will benefit a wide range of researchers in text mining, Artificial Intelligence, and natural languages. Machine translation has vast potential, and this research will contribute towards the growing field of the corpus linguistics of natural language texts.

Following five criteria have been considered for project evaluation:

a) **Research Focus:** Focusing the research on the translation version of the Bible is valuable since there are billions of people following the same. Also, the same text that has been used in this study is available in more than 25 languages which makes it easier for anyone to perform manual analysis and evaluation (section) for native speaker of other languages in future. This research area is quite crucial considering the number of languages in the world and the limited amount of labelled and parallel data available for them.

b) **Research Methodology:** This study has following CRISP-DM methodology throughout to have a structured way of progressing with understanding and implementing the work. CRISP-DM is a simple and powerful methodology independent of domain and technology. Under section 1.8, a chart describing the project plan has been included which suggests how the underlying idea and approach of CRISP-DM has collaborated with the work involved in this project.

c) **Originality:** There have a lot of studies done in the field of NMT and under-resourced languages, which explain the use of natural language processing and Seq2Seq model. However, finding the approach followed in this research specifically for our choice of languages (Sanskrit and Marathi), along with the translation model (Seq2Seq or encoder-decoder architecture model) is an unexplored area in general.

d) **Literature Review Study**: As part of background research, as well in other phases of this study, a lot of existing literature was studied to understand the idea of unified architecture similar to the one proposed in 'One model to learn them all'. Finally, with NMT we are witnessing unsupervised methods being applied to learn machine translation systems from only one language. Additional examination of literature was examined to get complete idea of all kinds of researches done in the field of natural language processing which concern data other than text (For e.g. image and speech processing). Close to 30 citations have been included in the references in this report.

e) **Learning And Development**: This project required a lot of background research concerning natural languages, under-resourced languages, in particular, detailed understanding of concepts text mining, working knowledge of tools and techniques used to build NMT model. I had a primary level of understanding of pre-requisites before starting with the project work. Collecting reliable dataset for any under-resourced language is significantly hard. As part of deliverables for this project, datasets for both, Sanskrit and Marathi, have been provided. These resources can be reused for machine translation work in future.

6.5 Achievement

The primary aim of this project was to apply relevant programming and statistical tools to analyse unstructured data and build a universal machine translation task which can learn from any other under-resourced language in future. The model developed as part of this project fulfils the aim. The model is ready to be used for any language without any additional modification required specific to the language. All we need to do is to provide a text file containing texts from a language.

The objective of this project was to perform an empirical investigation to provide a machine translation technique while following all the required steps in a systematic manner which would serve as a foundation for future researchers studying languages or machine translation. Throughout the realization of this project, entire process has been discussed in this dissertation. The outcomes of this project are as follows:

a) A wide range of literature was studied to establish the meaning and understanding of the term "under-resourced" languages. Different approaches, right from the beginning, of machine translation task, has been explored and discussed.

b) Three training and testing datasets have been provided for researchers in the field of natural languages or computational linguistic language texts. All datasets were made available on GitHub (Appendix B).

c) Applying natural language processing techniques like tokenisation, stemming, filtering was applied to get appropriate format for collected datasets which can be used in learning algorithms.

d) Implementing NMT and specifically Sequence-to-Sequence model to perform machine translation.

e) Conducting experiments to evaluate the model developed as part of this project.

f) Implementation of the objectives of this project has been explained in detail by detailed explanation, tables, and visualizations. The performance of the experiments was explored by evaluating the results as well as manually. All commonly-used machine translation models were explored before concluding that NMT is the most appropriate model for tasks involving translation of natural languages.

g) Source code of the machine translation model has been provided on GitHub.

6.6 Future Work

As machine translation applications are continuously evolving to achieve significantly high accuracy levels, they are being increasingly used in more areas of business, and new applications using machine translation are being introduced. That opens many doors for a lot of text mining as well as language enthusiasts. This study would serve as a foundation for several future research opportunities. A considerable amount of time and effort was invested to explore under-resourced languages, collecting dataset after investigating available datasets online, preparing the training and testing data and then coding the seq2seq model for machine translation, and then evaluating and analysing the results. Using the developed model and overall analysis, other text mining applications and machine translation tasks can be conducted for further improvisation.

The research and evaluation done for this project suggest that though NMT is the most appropriate machine translation model, the quality of the model can be improved significantly with bigger datasets. Also, a lot of evaluation metrics have been introduced specifically for machine translation model involving sequences which can be explored and a better understanding of the quality of translation can be obtained. There are several areas of technical application that use a variety of languages to reach out to a bigger audience and this project can be considered as an extension for their researches.

Furthermore, multiple studies and researches are being conducted at the University of Leeds in the School of Languages and/or School of English and lack of resources and automated translators is one of the primary challenges in studying under-resourced languages. This project could make a significant contribution to the school of languages in their endeavours.

6.7 Personal Reflection

This section summarizes the researcher’s experience in accomplishing the challenging task of building a universal machine translation model for under-resourced language. Also, the challenges faced during the entire course of this project has been discussed along with recommendations for future researchers attempting similar projects. Entire project experience has enabled me to understand a number of different areas of interests such as linguistic computing, natural language processing, text mining, machine learning, and specifically machine translation. During the early phases of this project, the critical challenges were to establish a clear understanding of the aims and objectives by exploring various approaches and then performing scoping and planning.

Prior to working on this project, I had a quite basic understanding of text mining and machine translation. These topics are essential to be understood in depth before attempting to design a solution for the given task. I started by investigating various research papers published on the study of languages, morphology, semantics of languages, text mining, natural language processing, and machine translation.

Machine translation has a rich history, starting from the 1930s, and It was important to understand the evolution to be able to decide the most appropriate machine translation model. The time spent in an extensive study of research papers helped me greatly in having overall clarity of my objectives before getting started with actual work required for this project.

Scoping and planning of this project turned out to be challenging due to lack of prior experience with similar tasks which impacted the overall implementation of this project. For example, I had started to work on this project keeping only Sanskrit in my mind. Finding a quality resource for Sanskrit is hard and developed model didn't perform very well against multiple evaluation metrics. The idea of closely-related language was then implemented and Marathi datasets were explored in detail and later introduced to the same model. This change helped me in being certain that the developed model is language-independent and no change at source code level was required to accommodate a new language. However, additional effort was required to collect Marathi dataset and then data understanding and the pre-processing process were implemented again.

This project is an empirical investigation project which means along with the understanding of multi-disciplinary areas like text mining and machine learning, different approaches, algorithms, evaluation techniques were required to be understood in detail before performing different experiments to reach a conclusion. Several useful sources, including the IEEE database, Google Scholar, CORE, DBLP Computer Science Bibliography, were decisive to this research and are highly recommended to anyone working on a similar project. Exploring these sources have helped me to locate relevant and reliable information.

To achieve the aims and objectives of any project of similar nature, it is important to allocate sufficient time for background research and have a sharp understanding of business problems, already carried out researches, and tools and technologies required in the course. CRISP-DM is an excellent methodology and following the same anyway helps us align tasks with phases which gives a proper distinction among task. However, further dividing tasks into subtasks and then setting flexible milestones for individual subtasks to accommodate any modification or unseen problem at hand. Individual deadlines and prioritization of subtasks can make project management easier. Other than sharpening my technical skills, this project has helped me develop my soft skills such as time management, diving confidently into a problem domain with no prior expertise, finding accurate and reliable resources, self-learning, etc.

Machine translation is an actively evolving field of research and the outcome of this project has a lot of scope of improvisation but this dissertation would serve as a foundation for anyone carrying out a similar project in the future. My overall experience of working on this paper was extremely satisfying and I am grateful that I have had this opportunity to learn and contribute to the research area of machine translation.

List of References

Adams, O. et al. (2017). Cross-Lingual Word Embeddings for Low-Resource Language Modelling.

Alosaimy, A. and Atwell, E. (2017). Tagging Classical Arabic Text using Available Morphological Analysers and Part of Speech Taggers.

Alturayeif, N. (2017). Text Mining and Similarity Measures of Quran and Bible.

Babych, B. (2017). Unsupervised induction of morphological lexicon for Ukrainian to appear in Proc CAMRL'2017.

BERMENT V. (2004), Méthodes pour informatiser des langues et des groups de langues peu dotées, PhD thesis, Université Joseph Fourier.

Bojanowski, P. et al. (2017). Enriching Word Vectors with Subword Information.

Bordes, A. et al. (2012). Joint Learning of Words and Meaning Representations for Open-Text Semantic Parsing.

Dickins, J. (2010). Basic Sentence Structure in Sudanese Arabic.

Douglas, F. (2009). Scottish Newspapers, Language and Identity.

Hofmann, M. and Klingenberg, R. 2013. RapidMiner: Data Mining Use Cases and Business
Analytics Applications. CRC Press. Available at:
http://dl.acm.org/citation.cfm?id=2543538.

KRAUWER S. (2003), “The Basic Language Resource Kit (BLARK) as the First Milestone for the Language Resources Roadmap”, in Proceedings of the International Workshop “Speech and Computer”, SPECOM 2003,Moscow, Russia.

Rios, M. and Sharoff, S. (2016). Language adaptation for extending post-editing estimates for closely related languages.

Ruder, S. et al. (2017). A Survey of Cross-lingual Word Embedding Models.

Sharoff, S. (2018). Language adaptation experiments via cross-lingual embeddings for related languages.

SketchEngine. (2017). Embedding Viewer.

Soricut, R. and Och, F. (2015). Unsupervised morphology induction using word embeddings.

Tulkens, S. et al. (2016). Evaluating Unsupervised Dutch Word Embeddings as a Linguistic Resource.

Wang, P. et al. (2015). Semantic expansion using word embedding clustering and convolutional neural network for improving short text classification.
Vine, B. (2017). Archive of New Zealand English.
Watson, J. (2012). The structure of Mehri.

Yang, Z. et al. (2016). Multi-Task Cross-Lingual Sequence Tagging from Scratch.

https://en.wikipedia.org/wiki/Marathi_language

Nicolas Boulanger-Lewandowski, Yoshua Bengio, and Pascal Vincent. 2013. Audio chord recognition with recurrent neural networks.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In Proceedings of the Empirical Methods in Natural Language Processing (EMNLP 2014), October. to appear.

Alex Graves. 2012. Sequence transduction with recurrent neural networks. In Proceedings of the 29th International Conference on Machine Learning (ICML 2012).

A. Graves. 2013. Generating sequences with recurrent neural networks. arXiv:1308.0850 [cs.NE], August.

S. Hochreiter and J. Schmidhuber. 1997. Long shortterm memory. Neural Computation, 9(8):1735– 1780.

Nal Kalchbrenner and Phil Blunsom. 2013. Two recurrent continuous translation models. In Proceedings of the ACL Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 1700–1709. Association for Computational Linguistics.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1, NAACL ’03, pages 48–54, Stroudsburg, PA, USA. Association for Computational Linguistics.

Appendix A
External Materials

The following corpora were used in the project:

- Translation corpus of the Bible (obtained from sanskritbible.in).
- Parallel translated text for English and Marathi (Obtained from http://www.manythings.org/anki/)

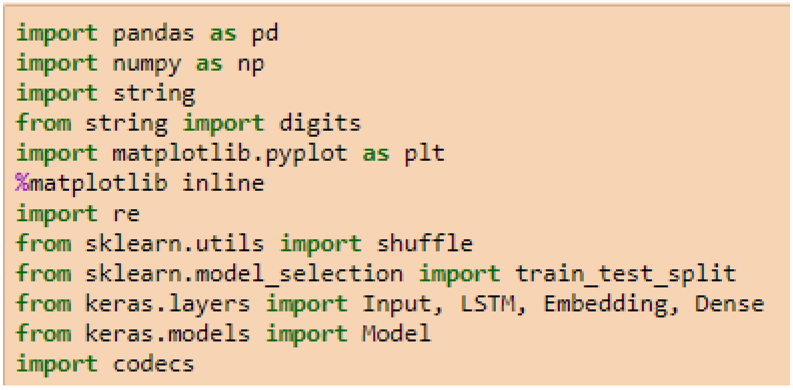
Appendix B

Programming in Python

As discussed in chapter 3, seq2seq model has been developed for performing machine translation and programming for this model has been done in Python. Complete source code is available at: *https://github.com/vatsagarima11/NLP*

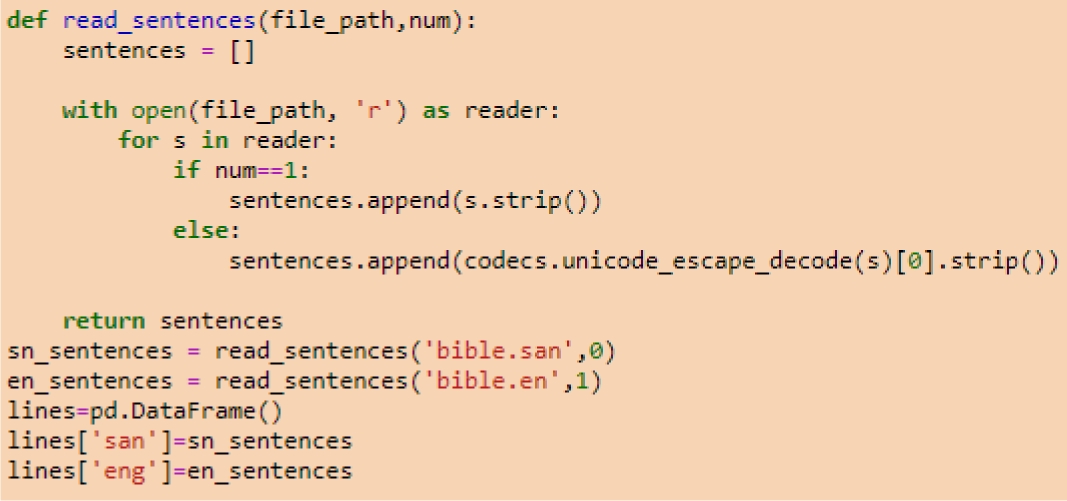
Below are the code snippets for tasks performed in the model.

1) Importing required libraries:



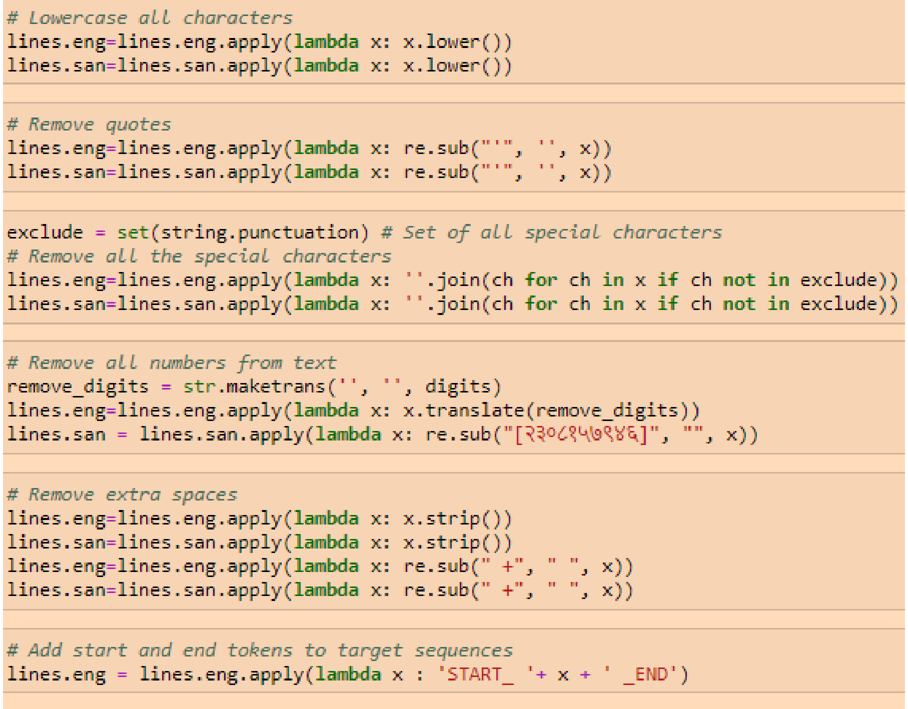
```
import pandas as pd
import numpy as np
import string
from string import digits
import matplotlib.pyplot as plt
%matplotlib inline
import re
from sklearn.utils import shuffle
from sklearn.model_selection import train_test_split
from keras.layers import Input, LSTM, Embedding, Dense
from keras.models import Model
import codecs
```

2) Reading text data from input files:



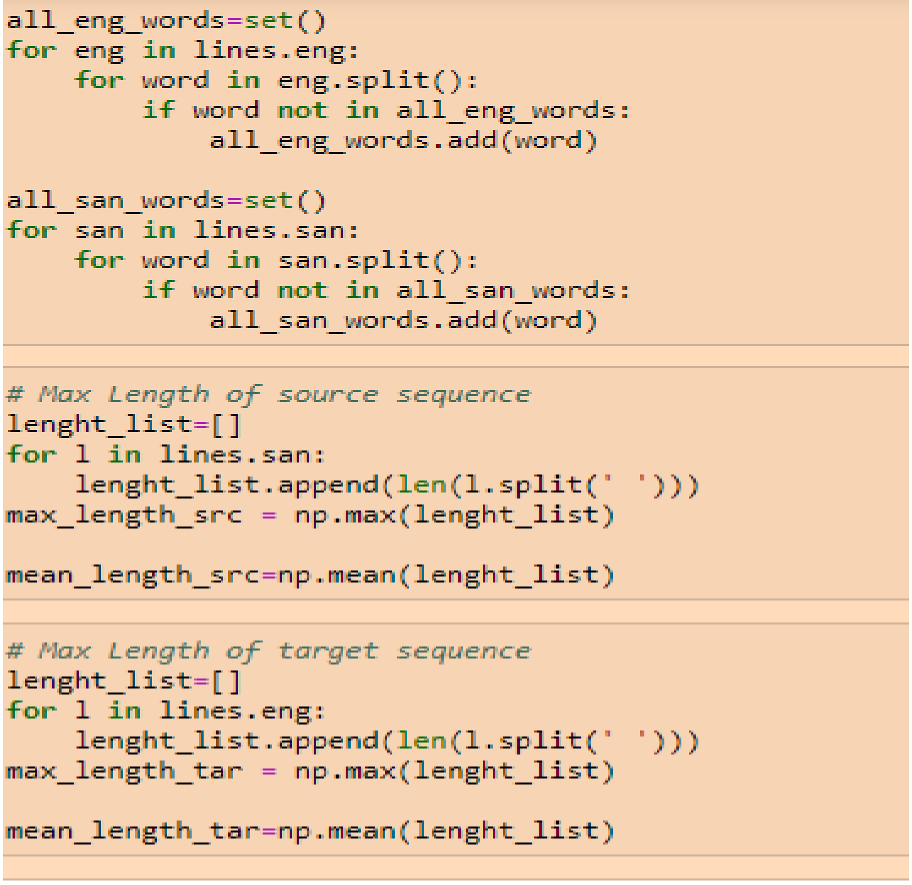
```
def read_sentences(file_path,num):
    sentences = []

    with open(file_path, 'r') as reader:
        for s in reader:
            if num==1:
                sentences.append(s.strip())
            else:
                sentences.append(codecs.unicode_escape_decode(s)[0].strip())

    return sentences
sn_sentences = read_sentences('bible.san',0)
en_sentences = read_sentences('bible.en',1)
lines=pd.DataFrame()
lines['san']=sn_sentences
lines['eng']=en_sentences
```

3) Pre-processing of text data to remove all special characters, whitespaces and lower casing all characters.



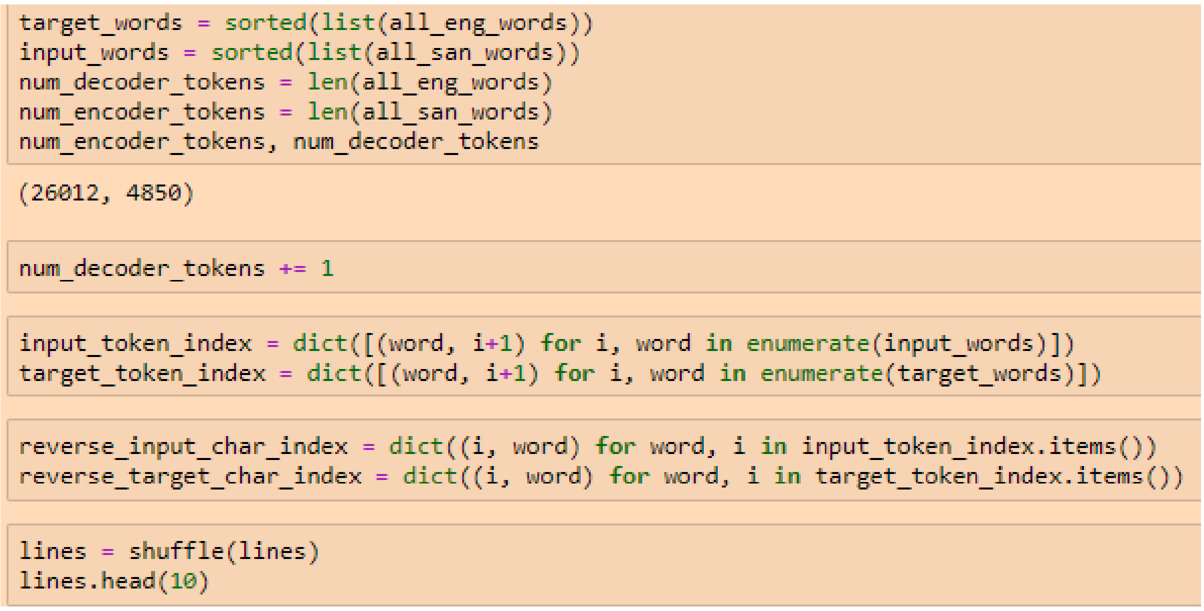
```
# Lowercase all characters
lines.eng=lines.eng.apply(lambda x: x.lower())
lines.san=lines.san.apply(lambda x: x.lower())

# Remove quotes
lines.eng=lines.eng.apply(lambda x: re.sub("'", '', x))
lines.san=lines.san.apply(lambda x: re.sub("'", '', x))

exclude = set(string.punctuation) # Set of all special characters
# Remove all the special characters
lines.eng=lines.eng.apply(lambda x: ''.join(ch for ch in x if ch not in exclude))
lines.san=lines.san.apply(lambda x: ''.join(ch for ch in x if ch not in exclude))

# Remove all numbers from text
remove_digits = str.maketrans('', '', digits)
lines.eng=lines.eng.apply(lambda x: x.translate(remove_digits))
lines.san = lines.san.apply(lambda x: re.sub("[२३०८१५७९४६]", "", x))

# Remove extra spaces
lines.eng=lines.eng.apply(lambda x: x.strip())
lines.san=lines.san.apply(lambda x: x.strip())
lines.eng=lines.eng.apply(lambda x: re.sub(" +", " ", x))
lines.san=lines.san.apply(lambda x: re.sub(" +", " ", x))

# Add start and end tokens to target sequences
lines.eng = lines.eng.apply(lambda x : 'START_ '+ x + ' _END')
```

4) Setting max length for source and target sequence



```
all_eng_words=set()
for eng in lines.eng:
    for word in eng.split():
        if word not in all_eng_words:
            all_eng_words.add(word)

all_san_words=set()
for san in lines.san:
    for word in san.split():
        if word not in all_san_words:
            all_san_words.add(word)

# Max Length of source sequence
lenght_list=[]
for l in lines.san:
    lenght_list.append(len(l.split(' ')))
max_length_src = np.max(lenght_list)

mean_length_src=np.mean(lenght_list)

# Max Length of target sequence
lenght_list=[]
for l in lines.eng:
    lenght_list.append(len(l.split(' ')))
max_length_tar = np.max(lenght_list)

mean_length_tar=np.mean(lenght_list)
```

5) Tokenization for cleaned dataset



```
target_words = sorted(list(all_eng_words))
input_words = sorted(list(all_san_words))
num_decoder_tokens = len(all_eng_words)
num_encoder_tokens = len(all_san_words)
num_encoder_tokens, num_decoder_tokens
```

```
(26012, 4850)
```

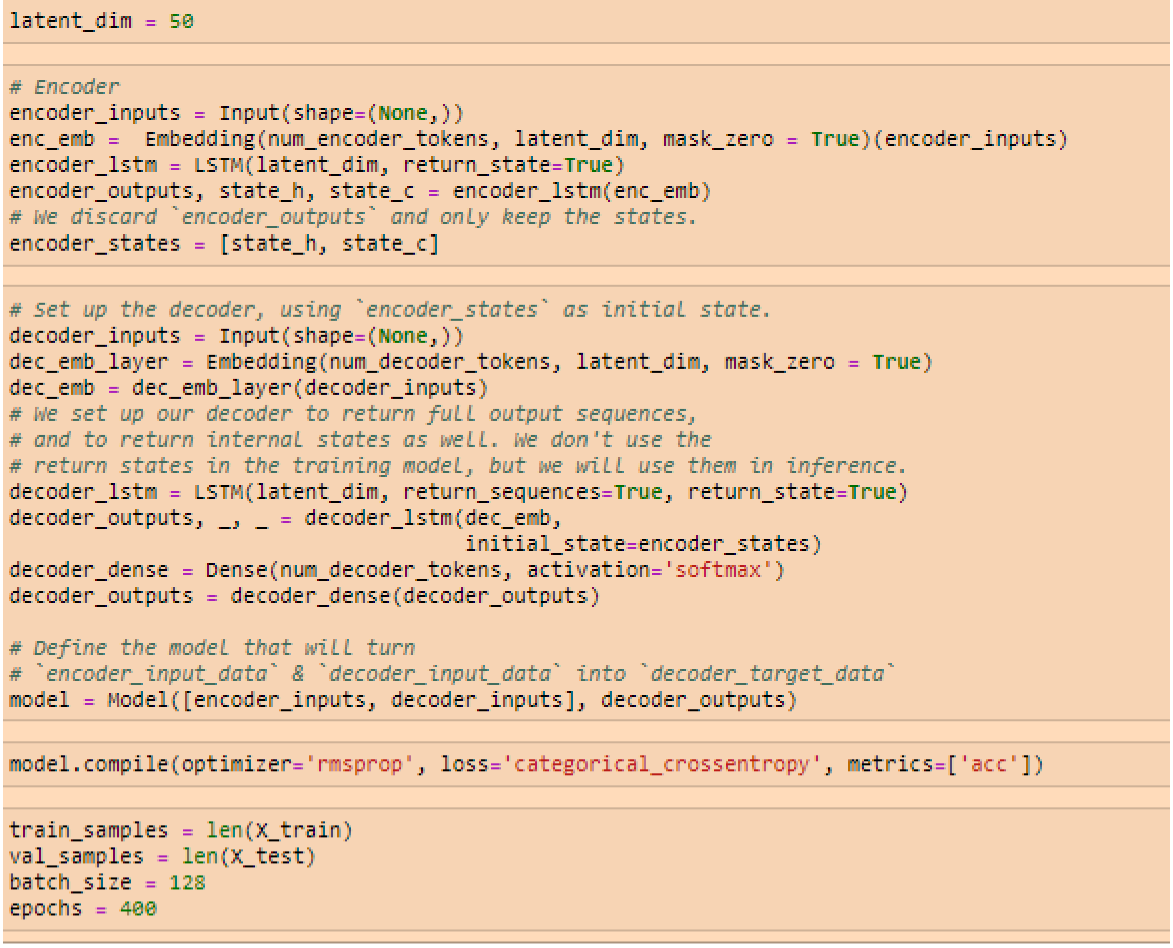
```
num_decoder_tokens += 1
```

```
input_token_index = dict([(word, i+1) for i, word in enumerate(input_words)])
target_token_index = dict([(word, i+1) for i, word in enumerate(target_words)])
```

```
reverse_input_char_index = dict((i, word) for word, i in input_token_index.items())
reverse_target_char_index = dict((i, word) for word, i in target_token_index.items())
```

```
lines = shuffle(lines)
lines.head(10)
```

6) Setting states from encoder to decoder and then compiling the model.



```
latent_dim = 50
```

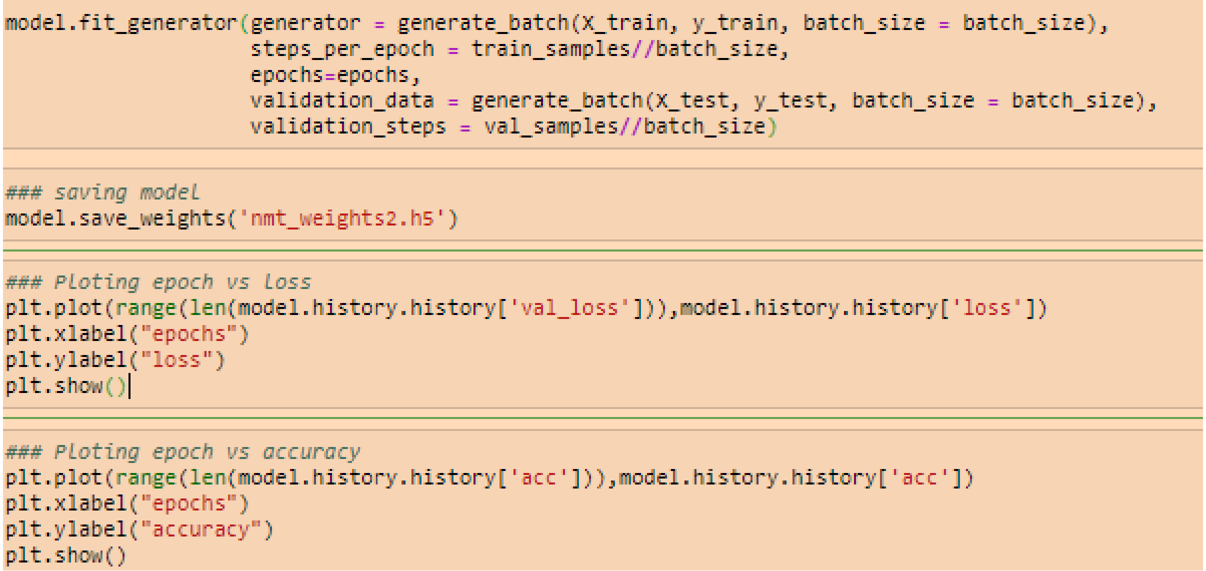
```
# Encoder
encoder_inputs = Input(shape=(None,))
enc_emb =  Embedding(num_encoder_tokens, latent_dim, mask_zero = True)(encoder_inputs)
encoder_lstm = LSTM(latent_dim, return_state=True)
encoder_outputs, state_h, state_c = encoder_lstm(enc_emb)
# We discard `encoder_outputs` and only keep the states.
encoder_states = [state_h, state_c]
```

```
# Set up the decoder, using `encoder_states` as initial state.
decoder_inputs = Input(shape=(None,))
dec_emb_layer = Embedding(num_decoder_tokens, latent_dim, mask_zero = True)
dec_emb = dec_emb_layer(decoder_inputs)
# We set up our decoder to return full output sequences,
# and to return internal states as well. We don't use the
# return states in the training model, but we will use them in inference.
decoder_lstm = LSTM(latent_dim, return_sequences=True, return_state=True)
decoder_outputs, _, _ = decoder_lstm(dec_emb,
                                     initial_state=encoder_states)
decoder_dense = Dense(num_decoder_tokens, activation='softmax')
decoder_outputs = decoder_dense(decoder_outputs)

# Define the model that will turn
# `encoder_input_data` & `decoder_input_data` into `decoder_target_data`
model = Model([encoder_inputs, decoder_inputs], decoder_outputs)
```

```
model.compile(optimizer='rmsprop', loss='categorical_crossentropy', metrics=['acc'])
```

```
train_samples = len(X_train)
val_samples = len(X_test)
batch_size = 128
epochs = 400
```

7) Training the model based on the epochs, batch size decided in the previous step



```
model.fit_generator(generator = generate_batch(X_train, y_train, batch_size = batch_size),
                    steps_per_epoch = train_samples//batch_size,
                    epochs=epochs,
                    validation_data = generate_batch(X_test, y_test, batch_size = batch_size),
                    validation_steps = val_samples//batch_size)
```

```
### saving model
model.save_weights('nmt_weights2.h5')
```

```
### Ploting epoch vs loss
plt.plot(range(len(model.history.history['val_loss'])),model.history.history['loss'])
plt.xlabel("epochs")
plt.ylabel("loss")
plt.show()
```

```
### Ploting epoch vs accuracy
plt.plot(range(len(model.history.history['acc'])),model.history.history['acc'])
plt.xlabel("epochs")
plt.ylabel("accuracy")
plt.show()
```

Ethical Issues Addressed

The project does not contain any private or personal data and no ethical issues had to be addressed.