To Study different Lexical Automatic Machine Translation Evaluation Metric for Indic languages

A Project Work Report

Submitted in the partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE WITH SPECIALIZATION IN ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

Submitted by:

Ankit Ghosal	20BCS6621
Ayush Raj	20BCS6612
Divya Prem	20BCS6618
Lalit Bisht	20BCS6609

Under the Supervision of:

Ms. Shweta Chauhan

CHANDIGARH UNIVERSITY, GHARUAN, MOHALI - 140413, PUNJAB MARCH - JUNE 2023

ACKNOWLEDGEMENT

In completing this project report on the project titled To Study different Lexical Automatic Machine Translation Evaluation Metric for Indic Languages, we would like to express our gratitude and appreciation to all those who gave us the possibility to complete this project.

Special thanks to our supervisor Mrs. Shweta mam who not only guided us but also with their suggestions and encouragement helped us during all times of the fabrication process and doing the project. We also sincerely thank them for their time spent proofreading and correcting our mistakes and errors during the project.

We would also like to acknowledge with much appreciation the crucial role of the university management and many thanks go to all lecturers and supervisors who have given their full effort in guiding the team in achieving the goal as well as their encouragement to maintain our progress intrack. Our profound thanks go to all classmates, especially to our friends for spending their time helping and giving support whenever we need it in fabricating our project.

The author

BONAFIDE CERTIFICATE

Certified that this project report
"Automatic Machine Translation Evaluation Metrics for Indic.Languages"
is the bonafide work of "Lalit Bisht", "Divyaprem" , "Ankit Ghosal" and "Ayush Raj" who carried out the project work under my/our supervision.

<<Signature of the head of the department>>	<<Signature of the supervisor>>
SIGNATURE	**SIGNATURE**
SHWETA CHAUHAN	<<NAME>>
SUPERVISOR	**HEAD OF THE DEPARTMENT**

Abstract

Nowadays, in the globalised context in which we find ourselves, language barriers can still be an obstacle to accessing information. On occasions, it is impossible to satisfy the demand for translation by relying only on human translators, therefore, tools such as Machine Translation (MT) are gaining popularity due to their potential to overcome this problem. Consequently, research in this field is constantly growing and new MT paradigms are emerging. In this paper, a systematic literature review has been carried out in order to identify what MT systems are currently most employed, their architecture, the quality assessment procedures applied to determine how they work, and which of these systems offer the best results. The study is focused on the specialised literature produced by translation experts, linguists, and specialists in related fields that include the English–Spanish language combination. Research findings show that neural MT is the predominant paradigm in the current MT scenario, being Google Translator the most used system. Moreover, most of the analysed works used one type of evaluation—either automatic or human—to assess machine translation and only 22% of the works combined these two types of evaluation. However, more than a half of the works included error classification and analysis, an essential aspect for identifying flaws and improving the performance of MT systems.

Table of Contents

1. INTRODUCTION

1.1 Problem Definition

The problem to be addressed is to study different lexical automatic machine translation evaluation metrics for Indic languages. The Indic languages are a group of languages spoken in the Indian subcontinent, including Hindi, Bengali, Tamil, Telugu, and others. Machine translation is the process of translating text from one language to another using computers. Automatic evaluation metrics are used to measure the quality of machine translation output.

There are several metrics available for evaluating machine translation quality, but not all of them are suitable for Indic languages. Therefore, the goal of this study is to explore and compare different lexical metrics that can effectively evaluate machine translation quality for Indic languages. This study will also consider the unique characteristics of Indic languages, such as their complex grammar and syntax, which can affect the accuracy of machine translation.

By studying and comparing different evaluation metrics for Indic languages, this project aims to improve the accuracy and reliability of machine translation for these languages, which can have a significant impact on communication and collaboration between people who speak different languages.

1.2 Problem Overview

The problem of studying different lexical automatic machine translation evaluation metrics for Indic languages is important because it can improve the accuracy and reliability of machine translation for these languages. Indic languages are spoken by a large number of people in the Indian subcontinent, and machine translation can be a useful tool for communication and collaboration between people who speak different languages. However, existing machine translation evaluation metrics may not be suitable for Indic languages, which have unique characteristics that can affect the accuracy of machine translation.

The goal of this study is to explore and compare different lexical metrics for evaluating machine translation quality in Indic languages. This will involve analyzing the strengths and weaknesses of various metrics and determining which ones are most effective for evaluating the accuracy of machine translation output. The study will also take into account the complex grammar and syntax of Indic languages and how these factors can impact the accuracy of machine translation.

The outcome of this study can have a significant impact on the development and improvement of machine translation systems for Indic languages, which can in turn facilitate communication and collaboration across language barriers. By identifying effective metrics for evaluating machine translation quality in Indic languages, this project can contribute to the advancement of natural language processing and help bridge the gap between different languages and cultures.

Indic Languages

Indic languages are a group of languages that belong to the Indo-European language family and are mainly spoken in South Asia. The most widely spoken branch of Indic languages is the Indo-Aryan languages, which have more than 800 million speakers in India, Pakistan, Bangladesh, Nepal, Sri Lanka, and Maldives . Some of the major

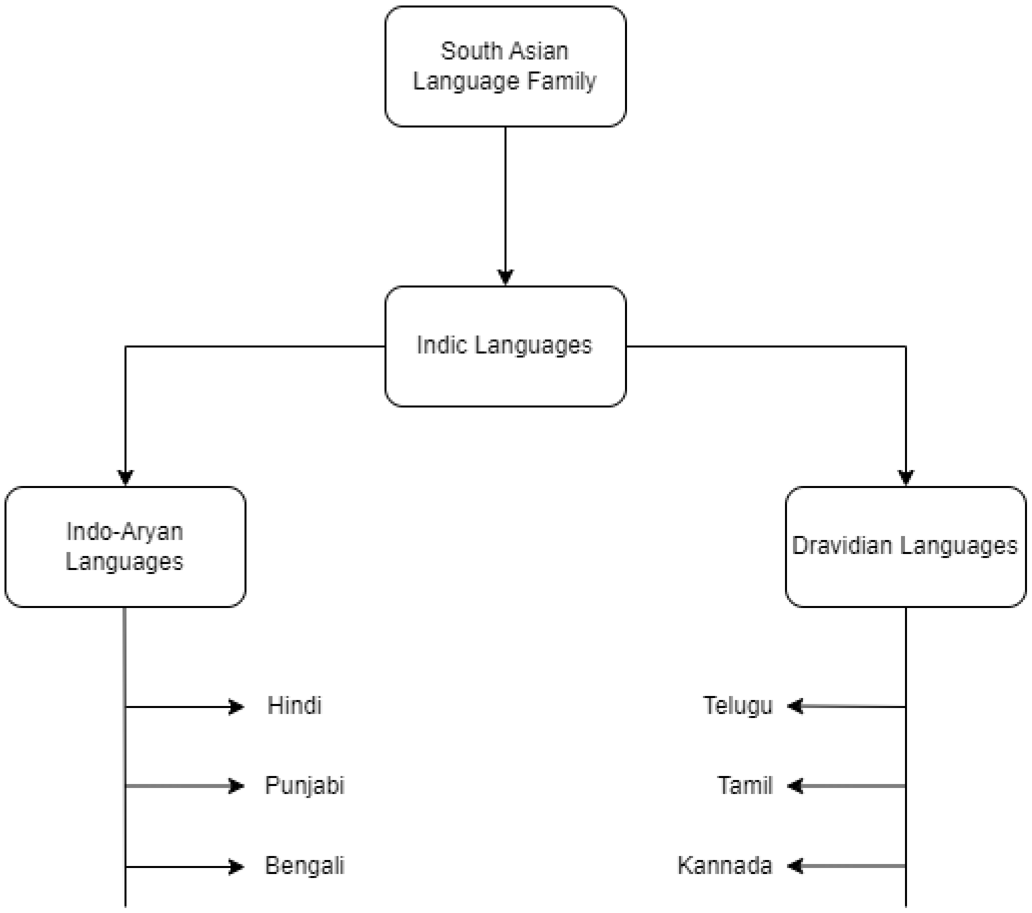
Indo-Aryan languages are Hindi, Bengali, Urdu, Punjabi, Marathi, Gujarati, Sindhi, Nepali, and Sinhala.

Another branch of Indic languages is the Dravidian languages, which are spoken by about 20% of Indians. The Dravidian languages are not related to the Indo-Aryan languages, but have influenced each other through contact and borrowing. Some of the major Dravidian languages are Tamil, Telugu, Kannada, Malayalam, and Odia.

The Indian Constitution recognizes 22 languages as official languages of India. These include 15 Indo-Aryan languages and 6 Dravidian languages. One of these languages is English, which is used as an associate official language along with Hindi. The Indian government also grants the status of classical language to six languages that have a long and rich literary tradition. These are Sanskrit, Tamil, Telugu, Kannada, Malayalam, and Odia.

Indic languages have a diverse and complex history and culture. They have developed various writing systems, such as Devanagari, Bengali-Assamese script, Gurmukhi script, Gujarati script, Oriya script, Sinhala script, Tamil script, Telugu script, Kannada script and Malayalam script. They have also produced many literary works of poetry, drama, epics, philosophy and religion. Some of the famous examples are the Vedas, the Ramayana, the Mahabharata, the Bhagavad Gita and the works of Kalidasa.

We have used Hindi, Bengali and Telugu for the purpose of this research paper. Hindi is the most used Indic language in the country with more than 500 million people calling it their native language. Bengali is the second most spoken language in India with a speaker base of more than 95 million and to add more variety we have also used Telugu which has a user base of more than 80 million. Using these languages, we aim to provide a comprehensive idea of how different evaluation metrics will perform when used to evaluate indic languages.



1.3 Hardware Specification

Recommended Operating Systems: Windows 8+ Minimum
RAM: 8 GB Hard Disk: 128 GB
Processor:-Intel CORE i5(1.50 GHZ) or above
Ethernet connection: (LAN) OR a wireless adapter (Wi-Fi)

1.4 Software Specification

Editor: Pycharm, Jupyter, Spyder
Language: Python
Libraries: NLTK library, JIWER, -e(Pyter), Spacy etc.

2. LITERATURE SURVEY

2.1 Existing System

Initial research has been done to translate Indian languages, mostly focusing Hindi and Bengali. However, most of the focus is still rule-based because of the unavailability of parallel data to build SMT systems for these languages. (Dasgupta et al., 2004) proposed an approach for English to Bangla MT that uses syntactic transfer of English sentences to Bangla with optimal time complexity. In generation stage of the phrases they used a dictionary to identify subject, object and also other entities like person, number and generate target sentences. (Naskar et al., 2006) presented an example-based machine translation system for English to Bangla. Their work identifies the phrases in the input through a shallow analysis, retrieves the target phrases using the example-based approach and finally combines the target phrases using some heuristics based on the phrase reordering rules from Bangla. They also discussed some syntactic issues between English and Bangla. (Anwar et al., 2009) proposed a method to analyze syntactically Bangla sentence using context sensitive grammar rules which accepts almost all types of Bangla sentences including simple, complex and compound sentences and then interpret input Bangla sentence to English using a NLP conversion unit. The grammar rules employed in the system allow parsing five categories of sentences according to Bangla intonation. The system is based on analyzing an input sentence and converting into a structural representation (SR). Once an SR is created for a particular sentence it is then converted to corresponding English sentence by NLP conversion unit. For conversion, the NLP conversion utilizes the corpus. (Islam et al., 2010) proposed a phrase-based Statistical Machine Translation (SMT) system that translates English sentences to Bengali. They added a transliteration module to handle OOV words. A preposition handling module is also incorporated to deal with systematic grammatical differences between English and Bangla. To measure the performance of their system, they used BLEU, NIST and TER scores. (Durrani et al., 2010) also made use of transliteration to aid translation between Hindi and Urdu which are closely related languages. (Roy & Popowich, 2009) applied three reordering techniques namely lexicalized, manual and automatic reordering to the source and language in a Bangla English SMT system. (Singh et al., 2012) presented a Phrase based model approach to English-Hindi translation. In their work they discussed the simple implementation of default phrase-based model for SMT for English to Hindi and also give an overview of different Machine translation applications that are in use nowadays. (Sharma et.al. 2011) presented English to Hindi SMT system using phrase-based model approach. They used human evaluation metrics as their evaluation measures. These evaluations cost higher than the already available automatic evaluation metrics. (Yamada & Knight, 2001) used methods based on tree to string mappings where source language sentences are first parsed and later operations on each node. (Eisner, 2003) presented issues of working with isomorphic trees and presented a new approach of nonisomorphic tree-to-tree mapping translation model using synchronous tree substitution grammar (STSG). (Li et al., 2005) first gave idea of using maximum entropy model based on source language parse trees to get n-best syntactic reordering's of each sentence which was further extended to use of lattices. (Bisazza & Federico 2010) further explored lattice-based reordering techniques for Arabic-English; they used shallow syntax chunking of the source language to move clause-initial verbs up to the maximum of 6 chunks where each verb's placement is encoded as separate path in lattice and each path is associated with a feature weight used by the decoder. (Jawad et al., 2010) presented complete study work for English to Urdu MT that uses factored based MT. In their work they discussed the complete divergence between two languages. Vocabulary difference between Urdu and English has been discussed. In their work they showed the importance of factored based models when we got information about the morphology of both source and targeted language. (Khan, et al., 2013) presented baseline SMT system for English to Urdu translation using Hierarchical Model given by (Chiang, et al., 2005) They also made a comparison of simple default phrase-based model with the hierarchical model and showed the performance of simple phrase-based is much better for such local language like Urdu then the hierarchical phrase-based approach to SMT. (Singh et al., 2008) presented a Punjabi to Hindi Machine Translation System. The proposed system for Punjabi to Hindi translation has been implemented with various research techniques based on Direct MT architecture and language corpus. The output is evaluated in order to get the suitability of the system for the Punjabi Hindi language pair. A lot of work is being carried out using Neural Networks technology in the field of MT which is being a good approach nowadays. Neural Machine Translation is a newly proposed approach in MT. The main drawback using the approach is it requires a relatively large amount of training corpus as compared to SMT. (Khalilov, et al., 2008) estimated a continuous space language model with a neural network in an Italian to English MT system. (Bahdanau,et al., 2014) presented a Neural Machine Translation by jointly learning to align and translate. In our work we used Phrase-based SMT models and evaluated their performance on the morphologically rich Indian languages.

2.2 Proposed System

Evaluation of machine translation systems for Indic languages poses unique challenges due to the complex morphology, syntax, and semantics of these languages. Here is a proposed system for machine translation evaluation for Indic languages:

Corpus Creation: The first step in evaluating machine translation systems for Indic languages is to create a corpus of parallel texts in the source and target languages. This corpus should cover a wide range of domains, genres, and registers to ensure the evaluation is robust.

Expert Evaluation: To capture the nuances of translation quality in Indic languages, expert evaluators who are fluent in both the source and target languages should be employed. The experts should evaluate the translations based on parameters such as fluency, grammaticality, coherence, adequacy, and faithfulness to the source text.

Reference-based Evaluation Metrics: Reference-based evaluation metrics such as BLEU, TER, and METEOR can be used as automated evaluation measures. However, these metrics may need to be modified or adapted to better capture the quality of translations in Indic languages.

Domain-specificity: Machine translation systems may perform differently in different domains, and evaluation should take this into account. For example, a machine translation system trained on news articles may not perform well on legal documents or scientific papers. Thus, evaluation should be carried out across multiple domains.

Open-ended Evaluation: Some translations require creativity and idiomatic expressions, which are difficult for machine translation systems to handle. In such cases, open-ended evaluation methods that allow for more subjective evaluation should be employed.

Human-Aided Evaluation: Human-aided evaluation methods, where a human post-editor checks and corrects the machine translation output, can also be used. This method can capture the quality of the final output and provide feedback on the strengths and weaknesses of the machine translation system.

In summary, a robust evaluation system for machine translation in Indic languages should include a combination of expert evaluation, reference-based evaluation metrics, domain-specificity, open-ended evaluation methods, and human-aided evaluation methods.

2.3 Literature Review Summary

Year and Citation	Article/ Author	Tools/ Software	Technique	Source	Evaluation Parameter
10 April 2021	Irene Rivera-Trigueros	NLP	VERTa, Rouge, METEOR, F-Meausre	Google Scholar	
16 Jan 2017	Nadir Durrani Dr	NLP	Statistical Machine Translation (SMT), Parallel Corpus, Phrase-based Translation	Cornell University	
05 Jan 2015	Simon Corston-Oliver, Michael Gamon and Chris Brocket	NLP	Decision Trees,CMU-Cambridge Statistical Language Modeling Toolkit	Microsoft Research	
2016	Aaron Li-Feng Han, Derek F. Wong , Lidia S. Chao	NLP	BLEU, TER, PER, WER	Faculty of Science University of Amsterdam, Faculty of Science and Technology University of Macau	

01 APRIL 2011	Jesús Giméez, Enriqe Amigo	NLP	BLEU, TER, WER, NIST, PER	TALP Research Center, LSI Department Universitat Politecnica ` de Cataluny	

.

3. PROBLEM FORMULATION

Evaluation of machine translation systems is a challenging problem due to several reasons. Here are some of the main evaluation problems in machine translation:

1. Subjectivity: Machine translation evaluation is often subjective because different people may have different opinions about what constitutes a good translation. For example, a translation that is accurate but not fluent may be acceptable in some situations but not in others.
2. Lack of reference translations: Many languages have limited resources, and there may be a lack of reference translations for a particular language pair or domain. This makes it difficult to evaluate the accuracy and quality of machine translation systems.
3. Domain-specificity: Machine translation systems may perform well in one domain but poorly in another. For example, a machine translation system trained on news articles may not perform well on legal documents or medical reports.
4. Limited coverage: Machine translation systems may have limited coverage of certain language features or structures, which can lead to inaccuracies in translation.
5. Over-reliance on automatic evaluation metrics: Many machine translation evaluation systems rely on automatic evaluation metrics such as BLEU or TER scores. While these metrics provide a quick and objective evaluation of machine translation quality, they may not always align with human judgments of translation quality.
6. Lack of creativity and idiomatic expressions: Some translations require creativity and idiomatic expressions which are difficult for machine translation systems to handle, leading to inaccuracies and lower quality translations.

To address these evaluation problems, researchers are working on developing new evaluation methods that take into account subjective human judgments, domain-specificity, and lack of reference translations. They are also exploring alternative evaluation metrics that can capture the nuances of translation quality, such as fluency, coherence, and adequacy.

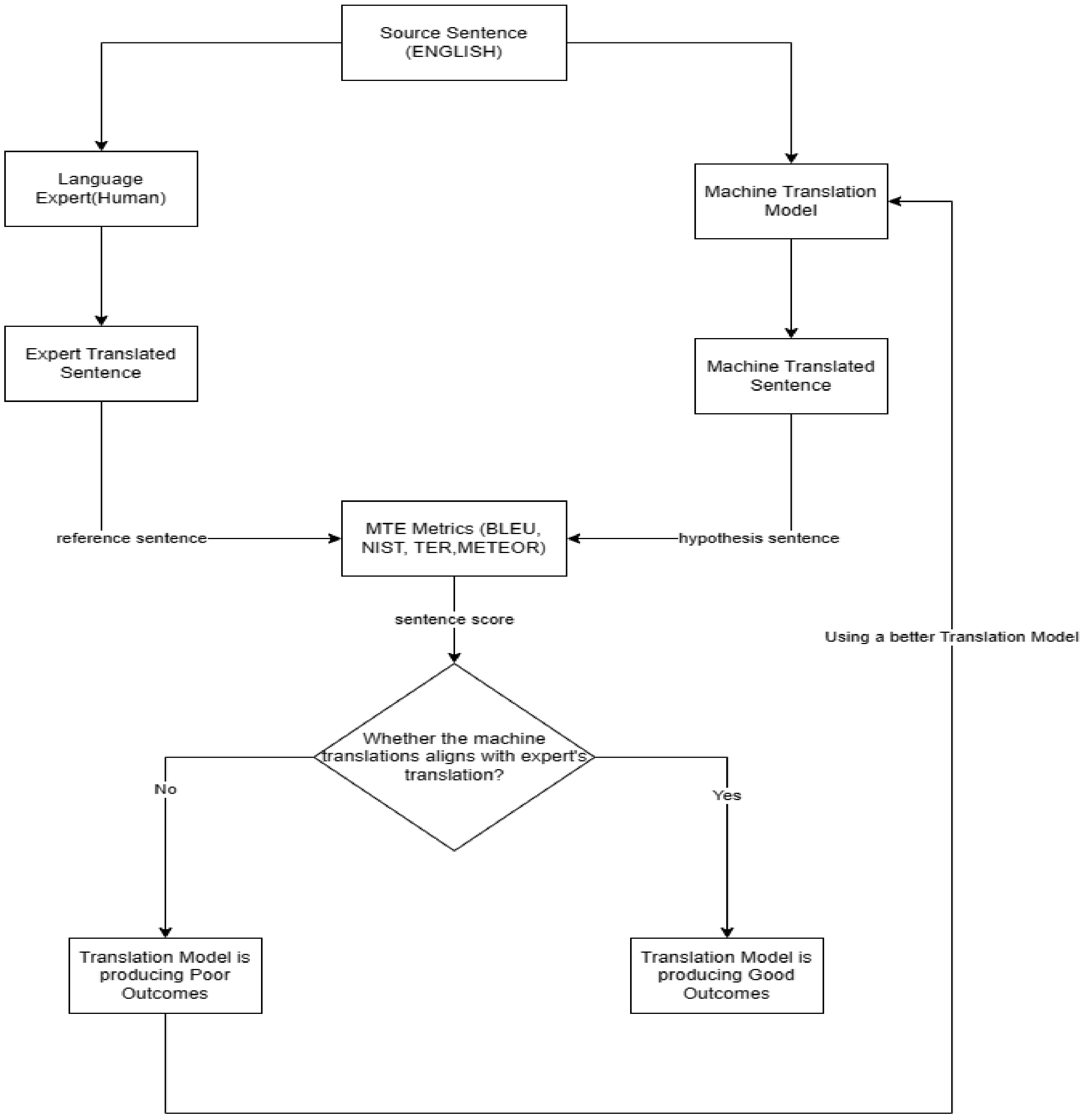
4. OBJECTIVES

The following are some objectives for studying different lexical automatic machine translation evaluation metrics for Indic languages:

1. To identify and analyze the strengths and weaknesses of different automatic machine translation evaluation metrics that are commonly used for other languages.
2. To review the unique linguistic characteristics of Indic languages, such as complex grammar and syntax, and to investigate how these characteristics can impact the accuracy of machine translation.
3. To evaluate the effectiveness of existing metrics for evaluating machine translation quality in Indic languages and to identify which metrics are most suitable for these languages.
4. To propose new metrics or modifications to existing metrics that can more accurately evaluate machine translation quality in Indic languages.
5. To conduct experiments using different evaluation metrics to compare the accuracy of machine translation output for Indic languages, and to draw conclusions about the most effective metrics for this purpose.
6. To create a set of guidelines or recommendations for developers and researchers working on machine translation systems for Indic languages, based on the results of the study.
7. To contribute to the advancement of natural language processing research and the development of more accurate and reliable machine translation systems for Indic languages.

5. METHODOLOGY

Design Flow / Process :



Source Sentence (ENGLISH)
Language Expert(Human)
Machine Translation Model
Expert Translated Sentence
Machine Translated Sentence
reference sentence
MTE Metrics (BLEU, NIST, TER,METEOR)
hypothesis sentence
sentence score
Using a better Translation Model
Whether the machine translations aligns with expert's translation?
No
Yes
Translation Model is producing Poor Outcomes
Translation Model is producing Good Outcomes

5.1 BILINGUAL EVALUATION UNDERSTUDY (BLEU)

Machine Translation (MT) is an essential task in the field of Natural Language Processing (NLP). It involves the use of computer algorithms to translate text from one language to another. One of the primary challenges in MT is evaluating the quality of the translated text. In recent years, the BLEU (Bilingual Evaluation Understudy) metric has become a standard evaluation metric for machine translation.

In this paper, we will explore the BLEU metric in detail, including its history, how it works, and its strengths and weaknesses. We will also compare the BLEU metric to other MT evaluation metrics and discuss how it can be improved in the future.

History of BLEU

BLEU was first introduced by Papineni et al. in 2002 in their paper "BLEU: a Method for Automatic Evaluation of Machine Translation." The metric was developed to address the need for an objective and automated way to evaluate the quality of machine translations. Before the development of BLEU, most MT evaluation was done manually, which was time-consuming and often subjective.

How BLEU Works

The BLEU metric works by comparing the output of a machine translation system to one or more human reference translations. It then calculates a score based on how closely the machine translation matches the reference translations.

The score is calculated based on the precision of the n-grams (contiguous sequences of n words) in the machine translation that match the n-grams in the reference translation. The BLEU score is a weighted geometric mean of the precision of the n-grams, with a penalty for translations that are too short.

The precision for each n-gram is calculated by dividing the number of times the n-gram appears in the machine translation by the total number of n-grams in the machine translation. The geometric mean is then calculated by taking the nth root of the product of the precisions for each n-gram up to a certain length.

BLEU can be calculated for a single translation or for a set of translations. In the latter case, the scores are averaged over all the translations in the set.

Strengths of BLEU

One of the main strengths of BLEU is that it is easy to compute and interpret. The metric only requires the machine translation and one or more reference translations, making it a straightforward and efficient evaluation tool.

BLEU is also language-independent, meaning it can be used to evaluate translations between any pair of languages. This makes it useful for comparing the quality of machine translations across different languages and for evaluating the performance of MT systems in multilingual settings.

Another strength of BLEU is that it correlates well with human judgments of translation quality. Studies have shown that BLEU scores have a high correlation with human judgments of translation quality, particularly for translations that are of low to moderate quality.

Weaknesses of BLEU

Despite its many strengths, BLEU has several weaknesses that can limit its usefulness in certain contexts. One of the primary weaknesses of BLEU is that it is a shallow metric that only looks at the overlap between the machine translation and the reference translations at the n-gram level. This means that BLEU can fail to capture important aspects of translation quality, such as fluency, idiomatic expression, and grammar.

Another weakness of BLEU is that it does not take into account the meaning of the translated text. This can be particularly problematic for languages with complex syntactic structures or for translations that involve idiomatic expressions or cultural references.

BLEU is also sensitive to the length of the machine translation. Translations that are too short are penalized, which can be problematic for languages with complex sentence structures or for translations that are naturally shorter than the reference translations.

5.2 NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY (NIST)

Machine Translation (MT) has become an increasingly important task in Natural Language Processing (NLP), with the ability to facilitate cross-lingual communication and enable access to information across language barriers. As a result, the evaluation of MT systems has become critical to ensure their accuracy and effectiveness. The National Institute of Standards and Technology (NIST) is one organization that has developed an evaluation metric for MT systems. This paper will discuss the NIST evaluation metric, including its history, how it works, and its strengths and weaknesses.

History of NIST

The NIST evaluation metric was developed by the National Institute of Standards and Technology (NIST) in the 1990s. The metric was developed in response to the need for an objective and automated way to evaluate the quality of MT systems. The metric was first used in the DARPA (Defense Advanced Research Projects Agency) evaluation campaign in 1999, and it has since become a standard metric for evaluating MT systems.

How NIST Works

The NIST evaluation metric works by comparing the output of a machine translation system to one or more human reference translations. The metric then calculates a score based on how closely the machine translation matches the reference translations.

The NIST metric uses a modified form of the BLEU metric, which calculates the precision of n-grams in the machine translation that match the n-grams in the reference translation. However, the NIST metric differs from the BLEU metric in several ways.

First, the NIST metric uses a weighted version of the precision calculation, which gives higher weight to longer n-grams. This is because longer n-grams are more informative and indicative of good translation quality than shorter n-grams.

Second, the NIST metric uses a logarithmic transformation of the precision scores, which reduces the impact of outliers and makes the scores more comparable across different translations.

Finally, the NIST metric uses a different approach to penalizing short translations. Instead of a fixed penalty, the NIST metric calculates the expected length of the translation based on the length of the source text, and then calculates the ratio of the actual length to the expected length. Translations that are shorter than the expected length are penalized, while translations that are longer than the expected length are rewarded.

Strengths of NIST

One of the primary strengths of the NIST evaluation metric is its robustness. The metric has been extensively tested and validated over many years of use, and it has been shown to be effective for evaluating the quality of machine translations across a wide range of languages and domains.

Another strength of the NIST metric is its flexibility. The metric can be used to evaluate translations at the sentence or document level, and it can be adapted to different languages and translation tasks. This makes it a useful tool for evaluating the performance of MT systems in a variety of contexts.

Finally, the NIST metric has a high correlation with human judgments of translation quality. Studies have shown that NIST scores have a high correlation with human judgments of translation quality, particularly for translations that are of high quality.

Weaknesses of NIST

Despite its many strengths, the NIST evaluation metric has several weaknesses that can limit its usefulness in certain contexts. One of the primary weaknesses of the NIST metric is that it is a shallow metric that only looks at the overlap between the machine translation and the reference translations at the n-gram level. This means that the metric can fail to capture important aspects of translation quality, such as fluency, idiomatic expression, and grammar.

Another weakness of the NIST metric is that it can be sensitive to the choice of reference translations. If the reference translations are not representative of the range of possible translations, the metric can give inaccurate results.

5.3 THE METRIC FOR EVALUATION OF TRANSLATION WITH EXPLICIT ORDERING (METEOR)

Machine Translation (MT) has become an important task in Natural Language Processing (NLP), with the ability to facilitate cross-lingual communication and enable access to information across language barriers. As a result, the evaluation of MT systems has become critical to ensure their accuracy and effectiveness. The Metric for Evaluation of Translation with Explicit Ordering (METEOR) is one organization that has developed an evaluation metric for MT systems. This paper will discuss the METEOR evaluation metric, including its history, how it works, and its strengths and weaknesses.

History of METEOR

The METEOR evaluation metric was developed by Alon Lavie and Abhaya Agarwal at Carnegie Mellon University in 2005. The metric was developed in response to the need for an evaluation metric that took into account not only the overlap between the machine translation and the reference translation, but also other factors such as synonymy, paraphrasing, and word order. The metric was first used in the Workshop on

Statistical Machine Translation in 2005, and it has since become a popular metric for evaluating MT systems.

How METEOR Works

The METEOR evaluation metric works by comparing the output of a machine translation system to one or more human reference translations. The metric then calculates a score based on how closely the machine translation matches the reference translations.

The METEOR metric uses a combination of unigram and bigram-based measures, which are combined using a harmonic mean. The metric also takes into account a variety of factors such as synonymy, paraphrasing, and word order.

The unigram-based measures compare the machine translation to the reference translations at the word level, taking into account synonyms and paraphrases. The bigram-based measures compare the machine translation to the reference translations at the phrase level, taking into account word order.

The METEOR metric also uses a concept called alignment, which matches the words in the machine translation to the words in the reference translations. This allows the metric to take into account differences in word order and word choice.

Finally, the METEOR metric uses a penalty function to penalize the machine translation for missing words or adding extra words that are not in the reference translation.

Strengths of METEOR

One of the primary strengths of the METEOR evaluation metric is its ability to take into account a variety of factors such as synonymy, paraphrasing, and word order. This makes it a more comprehensive metric than some of the other metrics that only look at the overlap between the machine translation and the reference translation.

Another strength of the METEOR metric is its ability to handle multiple reference translations. This is important because different human translators may produce different translations of the same sentence, and the METEOR metric can take into account these variations.

Finally, the METEOR metric has been shown to have a high correlation with human judgments of translation quality. Studies have shown that METEOR scores have a high correlation with human judgments of translation quality, particularly for translations that are of high quality.

Weaknesses of METEOR

Despite its many strengths, the METEOR evaluation metric has several weaknesses that can limit its usefulness in certain contexts. One of the primary weaknesses of the METEOR metric is that it can be sensitive to the choice of reference translations. If the reference translations are not representative of the range of possible translations, the metric can give inaccurate results.

Another weakness of the METEOR metric is that it can be affected by the quality of the automatic word alignment. If the automatic word alignment is not accurate, the metric can give inaccurate results.

Finally, the METEOR metric can be time-consuming to compute, particularly when multiple reference translations are used. This can limit its usefulness in certain contexts where speed is important.

5.4 TRANSLATION ERROR RATE (TER)

Evaluation of machine translation (MT) systems is crucial to determine their quality and effectiveness in producing translations that are accurate and appropriate for their intended use. Translation Error Rate (TER) is one of the widely used evaluation metrics for MT systems. TER is a simple and effective method for comparing machine translations with their corresponding human reference translations. This paper provides an overview of the TER evaluation metric, its history, how it works, and its strengths and weaknesses.

History of TER

The TER metric was developed by Snover et al. (2006) as part of the National Institute of Standards and Technology (NIST) MT evaluation campaign. The primary objective of the NIST MT evaluation campaign was to provide a standard benchmark for evaluating the quality of MT systems. The TER metric was developed to address the shortcomings of other evaluation metrics such as BLEU, which did not capture the full range of errors made by MT systems.

How TER Works

The TER metric works by comparing the output of a machine translation system with its corresponding human reference translation(s). It is based on the idea of edit distance, which is a measure of the number of edits (insertions, deletions, and substitutions) required to transform one string of text into another.

The TER score is calculated by dividing the edit distance between the machine translation and the reference translation(s) by the total number of words in the reference translation(s). The resulting value is then multiplied by 100 to give a percentage score.

The TER metric is capable of capturing a range of errors made by MT systems, including word order errors, missing or extra words, and errors in morphology, syntax, and semantics. The metric is also capable of handling multiple reference translations.

Strengths of TER

One of the primary strengths of the TER metric is its simplicity. The metric is easy to understand and implement, and it does not require any external resources such as language models or corpora. This makes it an attractive option for evaluating MT systems in low-resource settings.

Another strength of the TER metric is its ability to handle multiple reference translations. This is important because human translators may produce different translations of the same source text, and the TER metric can take into account these variations.

The TER metric has also been shown to have a high correlation with human judgments of translation quality. Studies have shown that TER scores have a high correlation with human judgments of translation quality, particularly for translations that are of high quality.

Weaknesses of TER

Despite its many strengths, the TER evaluation metric has some limitations. One of the primary weaknesses of the TER metric is that it does not take into account the quality of the reference translation(s). If the reference translations are of poor quality, the metric may give inaccurate results.

Another weakness of the TER metric is that it can be affected by the length of the sentences being evaluated. Longer sentences are more likely to contain errors, and the TER metric may penalize longer sentences more severely than shorter sentences.

Finally, the TER metric does not capture certain types of errors such as errors in word choice or ambiguity. This can limit its usefulness in evaluating certain types of MT systems, particularly those that rely heavily on context.

Conclusion

The Translation Error Rate (TER) evaluation metric is a simple and effective method for comparing machine translations with their corresponding human reference translations. It is capable of capturing a range of errors made by MT systems, and it has been shown to have a high correlation with human judgments of translation quality. Despite its limitations, the TER metric remains a popular choice for evaluating MT systems, particularly in low-resource settings.

5.5 GOOGLE TRANSLATE

Introduction

Google Translate is a popular machine translation service developed by Google. It allows users to translate text, documents, and web pages between different languages. The service uses a neural machine translation (NMT) system, which has been trained on vast amounts of data to produce translations that are often accurate and natural-sounding. This paper provides an overview of Google Translate, its history, how it works, and its strengths and weaknesses.

History of Google Translate

Google Translate was launched in April 2006, initially offering translation services for just a few languages. Over the years, Google has added support for more than 100 languages, making it one of the most widely used machine translation services in the world.

The service has undergone significant improvements over the years, particularly with the introduction of neural machine translation technology in 2016. The NMT system has enabled Google Translate to produce translations that are more accurate and natural-sounding than previous machine translation systems.

How Google Translate Works

Google Translate works by using statistical machine translation and neural machine translation technologies to produce translations. The service uses a vast amount of data, including existing translations and user-generated content, to train its machine learning models.

When a user inputs text for translation, the service breaks the text down into smaller segments, such as sentences or phrases. It then analyzes each segment, comparing it to similar segments in its database, to identify the best translation. The NMT system uses a sequence-to-sequence model to translate the input text, taking into account context, syntax, and grammar.

Google Translate also provides various options for users to customize their translations. For example, users can choose between formal or informal language, adjust the level of politeness, and choose between different dialects of a language.

Strengths of Google Translate

One of the primary strengths of Google Translate is its ability to handle a large number of languages. The service supports over 100 languages, including less common languages such as Bengali, Khmer, and Yoruba. This makes it a valuable tool for individuals and organizations working in multilingual environments.

Another strength of Google Translate is its speed and convenience. The service can provide translations instantly, making it a useful tool for individuals who need to quickly translate text on the go. The service is also available on multiple platforms, including web browsers, mobile devices, and desktop applications.

The NMT system used by Google Translate has also been shown to produce translations that are more accurate and natural-sounding than previous machine translation systems. This is particularly true for translations between closely related languages, such as Spanish and Portuguese, where previous machine translation systems often struggled.

Weaknesses of Google Translate

Despite its many strengths, Google Translate has some limitations. One of the primary weaknesses of the service is its reliance on machine learning models that are trained on large amounts of data. The accuracy of the translations can be affected by the quality and quantity of the data used to train the models.

Google Translate also struggles with translations between languages that are very different from each other, such as translations between English and Chinese. These translations often require a deeper understanding of the languages' syntax and grammar, which can be difficult for machine translation systems to replicate.

Finally, the service is not perfect, and there are occasional errors and mistranslations. These errors can be particularly problematic in sensitive contexts, such as legal or medical documents.

Conclusion

Google Translate is a popular machine translation service that provides quick and convenient translations between a large number of languages. The service uses a neural machine translation system that has been shown to produce translations that are often accurate and natural-sounding. Despite its limitations, Google Translate remains a valuable tool for individuals and organizations working in multilingual environments.

5.6 YANDEX TRANSLATE

Introduction

Yandex Translate is a machine translation service developed by Yandex, a Russian technology company. The service provides translations between multiple languages and is available on a variety of platforms, including web browsers, mobile devices, and desktop applications. This paper provides an overview of Yandex Translate, its history, how it works, and its strengths and weaknesses.

History of Yandex Translate

Yandex Translate was launched in 2011, initially offering translation services for just a few languages. Over the years, Yandex has added support for more than 100 languages, making it one of the most widely used machine translation services in Russia.

The service has undergone significant improvements over the years, particularly with the introduction of neural machine translation technology in 2017. The NMT system has enabled Yandex Translate to produce translations that are more accurate and natural-sounding than previous machine translation systems.

How Yandex Translate Works

Yandex Translate uses statistical machine translation and neural machine translation technologies to produce translations. The service uses a vast amount of data, including existing translations and user-generated content, to train its machine learning models.

When a user inputs text for translation, the service breaks the text down into smaller segments, such as sentences or phrases. It then analyzes each segment, comparing it to similar segments in its database, to identify the best translation. The NMT system uses a sequence-to-sequence model to translate the input text, taking into account context, syntax, and grammar.

Yandex Translate also provides various options for users to customize their translations. For example, users can choose between formal or informal language, adjust the level of politeness, and choose between different dialects of a language.

Strengths of Yandex Translate

One of the primary strengths of Yandex Translate is its accuracy. The NMT system used by the service has been shown to produce translations that are more accurate and natural-sounding than previous machine translation systems. This is particularly true for translations between closely related languages, such as Russian and Ukrainian, where previous machine translation systems often struggled.

Another strength of Yandex Translate is its ability to handle a large number of languages. The service supports over 100 languages, making it a valuable tool for individuals and organizations working in multilingual environments.

Yandex Translate also provides contextually appropriate translations, taking into account the context of the input text. This means that the service can produce translations that are more accurate and natural-sounding, particularly for complex or technical texts.

Weaknesses of Yandex Translate

Despite its many strengths, Yandex Translate has some limitations. One of the primary weaknesses of the service is its reliance on machine learning models that are trained on large amounts of data. The accuracy of the translations can be affected by the quality and quantity of the data used to train the models.

Yandex Translate also struggles with translations between languages that are very different from each other, such as translations between English and Chinese. These translations often require a deeper understanding of the languages' syntax and grammar, which can be difficult for machine translation systems to replicate.

Finally, the service is not perfect, and there are occasional errors and mistranslations. These errors can be particularly problematic in sensitive contexts, such as legal or medical documents.

Conclusion

Yandex Translate is a machine translation service that provides accurate translations between a large number of languages. The service uses a neural machine translation system that has been shown to produce translations that are often accurate and natural-sounding. Despite its limitations, Yandex Translate remains a valuable tool for individuals and organizations working in multilingual environments, particularly in Russian-speaking countries.

5.7 AIBHARAT DATASET

Introduction

The Ai Bharat dataset is a large-scale multilingual speech dataset developed by the Indian government's Ministry of Electronics and Information Technology (MeitY) and the Indian Institute of Technology, Madras. The dataset was created to support the development of speech recognition systems in Indian languages, which has been a challenging task due to the high linguistic diversity of the country. This paper provides an overview of the Ai Bharat dataset, its development, and its potential applications.

Background and Development of the Ai Bharat Dataset

India is a country with significant linguistic diversity, with over 22 official languages and hundreds of dialects. Speech recognition technology has traditionally been developed for major languages such as English, Mandarin, and Spanish. However, the lack of speech recognition systems for Indian languages has been a significant barrier to the development of technology and services that can serve the country's diverse population.

In response to this challenge, MeitY launched the "Ai for All" initiative, which aims to develop and promote the use of artificial intelligence (AI) technology in India. As part of this initiative, MeitY and IIT Madras collaborated to create the Ai Bharat dataset, which is the first large-scale multilingual speech dataset for Indian languages.

The dataset was created by recording speech from over 10,000 speakers across 19 Indian languages, including Hindi, Bengali, Tamil, and Telugu. The recordings were collected in a variety of environments, including indoor and outdoor settings, to ensure that the dataset represents the real-world variability of speech. The dataset includes over 1.5 million speech segments, totaling more than 2,000 hours of speech.

Potential Applications of the Ai Bharat Dataset

The Ai Bharat dataset has the potential to support a wide range of applications, particularly in the development of speech recognition systems for Indian languages. Speech recognition is a critical technology that can enable people to interact with technology and services more efficiently, especially for those who are illiterate or have limited access to education.

The dataset can be used to train speech recognition systems for a variety of applications, including voice assistants, call centers, and speech-to-text applications. The dataset's multilingual nature also makes it valuable for research into multilingual speech recognition systems, which can have significant benefits for countries with diverse linguistic populations.

Furthermore, the Ai Bharat dataset can be used to support research in other areas, such as natural language processing and machine learning. The dataset's size and variety of languages and speakers make it a valuable resource for developing and testing new algorithms and models.

Conclusion

The Ai Bharat dataset is a significant contribution to the development of speech recognition technology in India. The dataset's size, multilingual nature, and diversity of speakers make it a valuable resource for researchers and developers working on speech recognition, natural language processing, and machine learning. The dataset has the potential to support a wide range of applications that can benefit India's diverse population, particularly those who have limited access to education and technology. As such, the Ai Bharat dataset is a valuable contribution to the field of AI and has the potential to make a significant impact on the lives of millions of people.

5.8 NORMALISATION

Introduction

Normalization and Pearson correlation are two important statistical concepts that are widely used in research and data analysis. Normalization is the process of transforming data so that it conforms to a common scale, while Pearson correlation is a measure of the linear relationship between two variables. This paper provides an overview of normalization and Pearson correlation, their importance in data analysis, and their practical applications.

Normalization

Normalization is a process of transforming data so that it conforms to a common scale. The main goal of normalization is to make different variables comparable by eliminating the effects of differences in their scales. Normalization is commonly used in data preprocessing and data mining to improve the accuracy and efficiency of statistical analyses.

Normalization techniques are classified into two categories: min-max normalization and z-score normalization. Min-max normalization scales data so that it falls between a minimum and maximum value, typically between 0 and 1. This technique is commonly used in image processing and computer vision applications. Z-score normalization transforms data so that it has a mean of 0 and a standard deviation of 1. This technique is commonly used in data analysis and statistical modeling.

Normalization is important in data analysis because it eliminates the effects of differences in the scales of variables, making it easier to compare and analyze data.

Normalization also reduces the risk of bias in statistical analyses by ensuring that different variables have equal weight in the analysis.

Pearson Correlation

Pearson correlation is a measure of the linear relationship between two variables. It is commonly used in statistical analysis to determine the degree to which two variables are related. The Pearson correlation coefficient is a value between -1 and 1, where -1 indicates a perfect negative correlation, 0 indicates no correlation, and 1 indicates a perfect positive correlation.

Pearson correlation is commonly used in research to determine the degree to which two variables are related. For example, researchers may use Pearson correlation to determine the relationship between smoking and lung cancer or between exercise and heart disease. Pearson correlation can also be used to identify variables that are strongly correlated, which can be useful in identifying potential confounding variables in statistical analyses.

Practical Applications

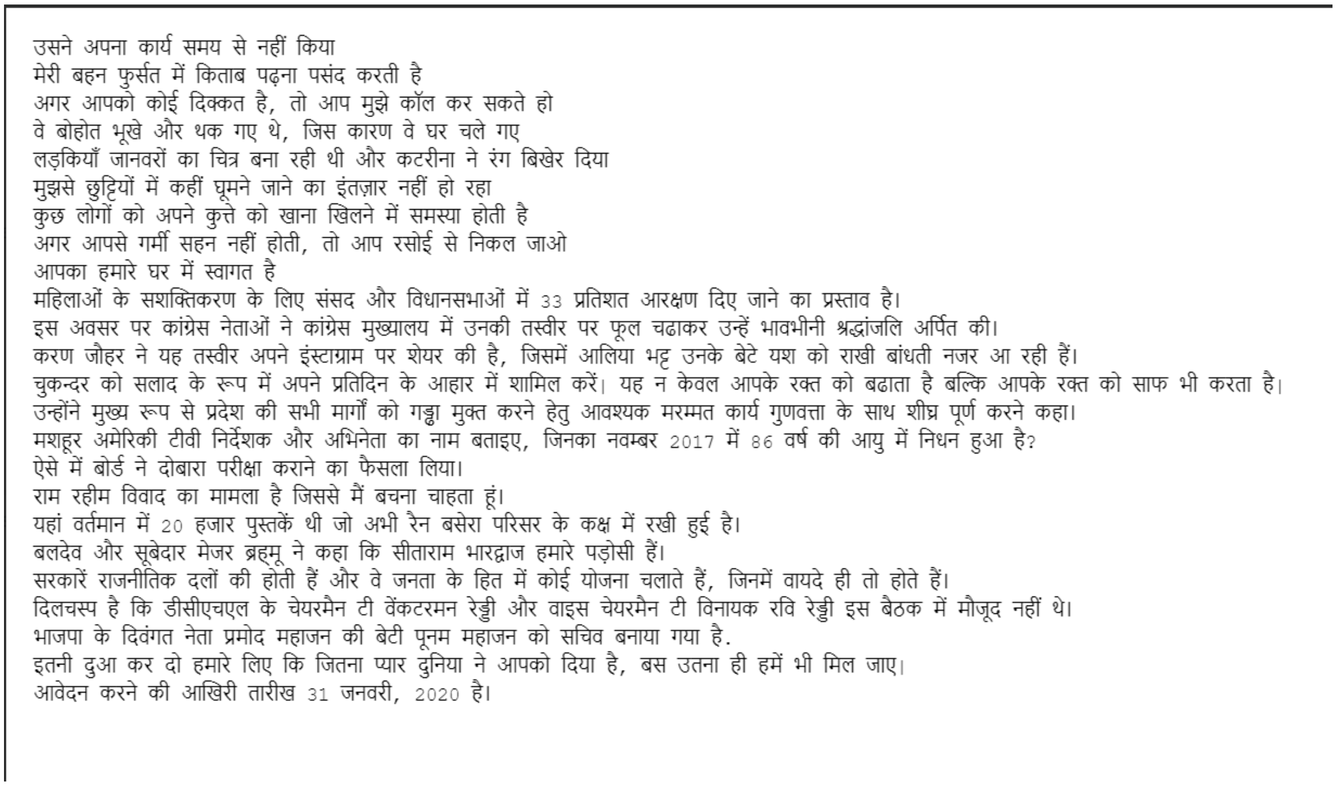
Normalization and Pearson correlation have practical applications in a wide range of fields, including healthcare, finance, and engineering. In healthcare, normalization and Pearson correlation are used to analyze data from clinical trials, identify risk factors for diseases, and develop predictive models for patient outcomes. In finance, normalization and Pearson correlation are used to analyze financial data, identify trends, and develop investment strategies. In engineering, normalization and Pearson correlation are used to analyze data from experiments, identify correlations between variables, and develop models for predicting performance.

Conclusion

Normalization and Pearson correlation are two important statistical concepts that are widely used in research and data analysis. Normalization is the process of transforming data so that it conforms to a common scale, while Pearson correlation is a measure of the linear relationship between two variables. Both concepts are important in data analysis because they enable researchers to compare and analyze data more effectively. The practical applications of normalization and Pearson correlation are widespread and can be found in a wide range of fields. As such, an understanding of these concepts is essential for researchers and data analysts who are working with complex datasets.

5.8 SAMPLE DATASET

HINDI LANGUAGE



उसने अपना कार्य समय से नहीं किया
मेरी बहन फुर्सत में किताब पढ़ना पसंद करती है
अगर आपको कोई दिक्कत है, तो आप मुझे कॉल कर सकते हो
वे बोहोत भूखे और थक गए थे, जिस कारण वे घर चले गए
लड़कियाँ जानवरों का चित्र बना रही थी और कटरीना ने रंग बिखेर दिया
मुझसे छुट्टियों में कहीं घूमने जाने का इंतज़ार नहीं हो रहा
कुछ लोगों को अपने कुत्ते को खाना खिलने में समस्या होती है
अगर आपसे गर्मी सहन नहीं होती, तो आप रसोई से निकल जाओ
आपका हमारे घर में स्वागत है
महिलाओं के सशक्तिकरण के लिए संसद और विधानसभाओं में 33 प्रतिशत आरक्षण दिए जाने का प्रस्ताव है।
इस अवसर पर कांग्रेस नेताओं ने कांग्रेस मुख्यालय में उनकी तस्वीर पर फूल चढाकर उन्हें भावभीनी श्रद्धांजलि अर्पित की।
करण जौहर ने यह तस्वीर अपने इंस्टाग्राम पर शेयर की है, जिसमें आलिया भट्ट उनके बेटे यश को राखी बांधती नजर आ रही हैं।
चुकन्दर को सलाद के रूप में अपने प्रतिदिन के आहार में शामिल करें। यह न केवल आपके रक्त को बढाता है बल्कि आपके रक्त को साफ भी करता है।
उन्होंने मुख्य रूप से प्रदेश की सभी मार्गों को गड्ढा मुक्त करने हेतु आवश्यक मरम्मत कार्य गुणवत्ता के साथ शीघ्र पूर्ण करने कहा।
मशहूर अमेरिकी टीवी निर्देशक और अभिनेता का नाम बताइए, जिनका नवम्बर 2017 में 86 वर्ष की आयु में निधन हुआ है?
ऐसे में बोर्ड ने दोबारा परीक्षा कराने का फैसला लिया।
राम रहीम विवाद का मामला है जिससे मैं बचना चाहता हूं।
यहां वर्तमान में 20 हजार पुस्तकें थी जो अभी रैन बसेरा परिसर के कक्ष में रखी हुई है।
बलदेव और सूबेदार मेजर ब्रह्मू ने कहा कि सीताराम भारद्वाज हमारे पड़ोसी हैं।
सरकारें राजनीतिक दलों की होती हैं और वे जनता के हित में कोई योजना चलाते हैं, जिनमें वायदे ही तो होते हैं।
दिलचस्प है कि डीसीएचएल के चेयरमैन टी वेंकटरमन रेड्डी और वाइस चेयरमैन टी विनायक रवि रेड्डी इस बैठक में मौजूद नहीं थे।
भाजपा के दिवंगत नेता प्रमोद महाजन की बेटी पूनम महाजन को सचिव बनाया गया है.
इतनी दुआ कर दो हमारे लिए कि जितना प्यार दुनिया ने आपको दिया है, बस उतना ही हमें भी मिल जाए।
आवेदन करने की आखिरी तारीख 31 जनवरी, 2020 है।

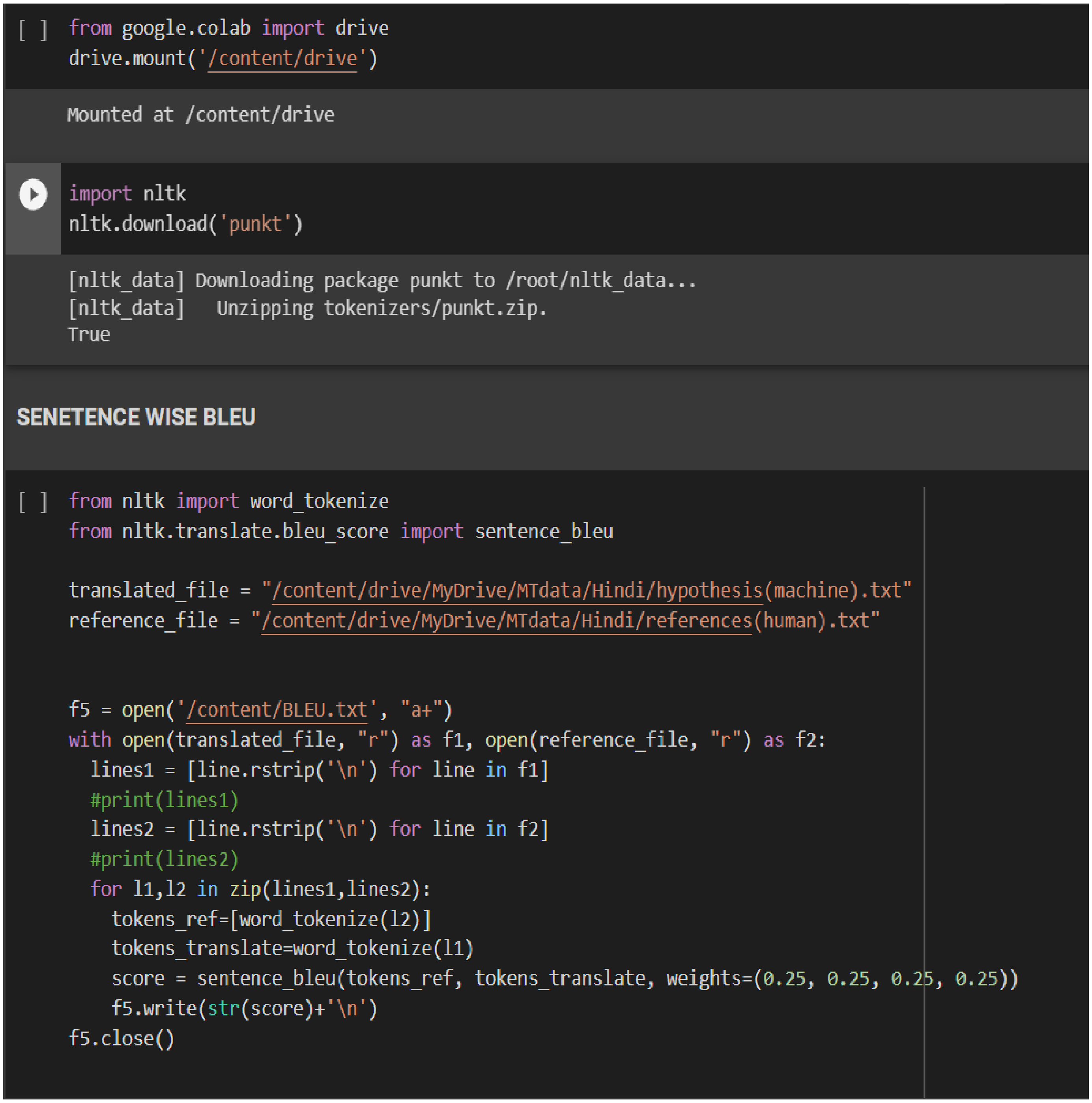
সে শুধু তার দেশেই নয়, এমনকি তার রাজ্যেও বিখ্যাত।
আজ বিকেলের মধ্যে এই সমস্যার সমাধান হবে।
একজন দোকানদার কে ভেজাল মেশানো শস্য বিক্রি করার জন্য রিপোর্ট করা হয়েছিল।
সে গরীব হলেও সৎ।
মন দিয়ে কাজ করো, অন্যথায় তুমি ব্যর্থ হবে।
সে আমাকে ঠান্ডা জল খেতে দেয়নি।
তাকে এখানে থাকতেই হবে।
এই গ্রামেই আমি বড় হয়েছি।
তুমি কি জানো আজ সব দোকান বন্ধ কেন?
পৃথিবীর অনেক আশ্চর্যই ঐতিহাসিক স্মৃতিসৌধের স্বীকৃতি পেয়েছে।
মার্শাল আর্টস শেখার জন্য তোমার ভালো প্রতিবর্ত ক্রিয়া এবং তৎপর শরীর প্রয়োজন।
সোসাইটির বেশির ভাগ মানুষই অনুরোধটি রাখবে।
বই পড়ে দর্জির কাজ শেখা যায়।
আমার বাবা ছুটির দিনে কাজ করেন না।
ছোটবেলা থেকেই বাস্কেট বল আমার প্রিয় খেলা।
তার জুতো টা একদম নতুন কিন্তু এখন হারিয়ে গেছে।
তুমি যদি মন দিয়ে কাজ কর তবে হারওয়ার্ডের মতন ভালো কলেজে পড়ার সুযোগ পাবে।
সামারা অফিসে গেছিল তার কোট টা আনতে এবং তারপর রাতের খাওয়ারের জন্য ফিরে এসেছিল।
ঘোড়া সাধানরত সবাই পছন্দ করে কিন্তু টিকটিকি কেও পছন্দ করে না।
তার সব চেয়ে বড় ভয় হলো ভিড় রাস্তায় গাড়ি চালানো।
দেশের যেকোনো জায়গাই লু বইতে পারে এবং তা শরীর খারাপ এমনকি মৃত্যুরও কারণ হতে পারে।

TELUGU LANGUAGE

ప్రకాష్ రాజ్ పారితోషకాల విషయంలో చాలా పక్కాగా ఉంటాడని పేరు.
ఒక టెస్టు మ్యాచ్‌లో తొలుత శతకం బాది, రెండో ఇన్నింగ్స్‌లో 50కి పైగా పరుగులు చేయడం స్మిత్‌కు ఇది తొమ్మిదోసారి.
తమిళ సినిమా అయినా అక్కడక్కడ తెలుగు బోర్డులు కనిపించేలా చేశారు. ఇంకా సూపర్‌గా రన్ అవుతోంది కనుక ఫుల్ రన్‌తో నూట ఎనబై నుంచి రెండు వందల కోట్ల నెట్ వసూళ్ల
వరకు సాధించే అవకాశముంది.
ఈ పట్టణానికి అతి సమీపంలో మధుర్ మహాగణపతి అనే ఆలయం ఉంది.
బాలాజీ మోహన్ దర్శకత్వంలో వస్తున్న తమిళ్ సినిమాలో ధనుష్ హీరోగా ,సింగర్ విజయ్ ఏసుదాస్ నటించనున్నారు.
బీజేపీ వారు నమ్రతగా అంబేడ్కర్ సిద్ధాంతాలను వెలివేయాలని చూస్తున్నారు.
రాజా, యిక్కడ మనం ఎక్కువ సేపు ముచ్చటలాడడం మంచిదికాదు.
ఇప్పుడే ఇలా ఉంటే ఇక ముందు ముందు ఎండలు ఇంకెంత తీవ్రంగా ఉంటాయో అనిపిస్తుంది కదా.
మరో వివాహం చేసుకునేందుకు వీలుగా భార్య పాయల్ నుంచి విడాకులు ఇప్పించాలి ’ అని ఒమర్ పిటిషన్‌లో విన్నవించారు.
తెలుగు, హిందీతో పాటు ఇతర దక్షిణాది భాషల్లో పాన్‌ఇండియా మూవీగా తెరకెక్కించబోతున్నారు.
నా ప్రయాణం మొత్తం పదిరోజులే కాబట్టి నేను మూడు నాలుగు ప్రదేశాల కన్న ఎన్నుకోలేను, ఒక్కో చోటా రెండు మూడు రోజుల కన్న ఎక్కువ సమయం ఉండలేను.
రవితేజ, నాగార్జునలతో కుదరని ఈ కథను నానితో తెరకెక్కించాలని దిల్ రాజు భావిస్తున్నాడు.
రాయలసీమలోనూ అక్కడక్కడా వర్షాలు కురిసే అవకాశం ఉందని పేర్కొన్నారు.
ఆయన నిన్న గుంటూరులో మీడియాతో మాట్లాడుతూ పవన్ కళ్యాణ్ చేసిన సూచనలను పరిగణనలోకి తీసుకొని సమస్యను సామరస్యంగా పరిష్కరిస్తామని చెపుతున్నారు.
నేటి లావాదేవీల్లో హీరోహోండా, హీరో మోటార్ కార్ షేర్ విలువ అత్యధిక స్థాయిలో 2.31 శాతం నష్టపోయింది.
రాజధానిపై అసెంబ్లీలో చర్చించిన తరువాతే నిర్ణయం ఉంటుందని స్పష్టం చేశారు.
అదే సమయంలో, తెలంగాణలోని అన్ని పార్టీల నేతలూ కలసి. . పోలవరం డామ్ ఎత్తు తగ్గించాలనీ, లేదంటే రాజమండ్రి మునిగిపోతుందనీ, ఇలాంటి అనుమానాలను ఆంధ్రా
నేతలతో వారు మాట్లాడించేవారన్నారు.
టాటా మోటార్స్ తమ పోటీదారులు ఉత్పత్తుల కంటే టియాగో కారు మీద ప్రతి నెలా సేల్స్ పెరగడాన్ని గమనించింది. నిజమే, టాటా టియాగో ఇప్పటి వరకు జరిపిన విక్రయాల్లో
జూన్ నెల సేల్స్ అత్యధికం. దీనికి తోడు దీని ప్రధాన పోటీదారులను కూడా ఇదే నెలలో వెనక్కి నెట్టింది.
దిల్‌రాజు నైజాం రైట్స్‌ను తన వద్ద ఉంచుకుని మిగిలి అన్ని ఏరియాలను అమ్మేసే యోచనలో ఉన్నాడు.
మరో వైపు హైదరాబాద్‌ను కిరోసిన్ రహిత నగరంగా మార్చడానికి వంట గ్యాస్ కనెక్షన్ లేని వారందరికీ కనెక్షన్లు ఇచ్చి అంచలంచెలుగా కిరోసిన్ కోటాకు కోత పెట్టాలని చూస్తున్నారు.
వరద నీట మునిగిన నాలుగు గ్రామాలకు విద్యుత్ సరఫరా నిలిచిపోయింది.
ఈ ప్రాంతంలోకి అడవుల నుంచి విష సర్పాలు నిత్యమూ వస్తుంటాయని, ఏనుగు ఇలా రావడం మాత్రం ఇదే తొలిసారని ఇక్కడి జవాన్లు తెలిపారు.
రుచితోపాటు పరిశుభ్రతను పాటించడం, బయట హోటళ్ల కంటే తక్కువ రేటుకు విక్రయించడం వల్ల వీటి ప్రాబల్యం పెరుగుతోంది.
ఈ సినిమా కోసం మరే సినిమా కూడా చేయడానికి ఒప్పుకోలేదు.
రాష్ట్రంలో సగ భాగం మహిళలు ఉండగా మహిళలకు ఒక్క మంత్రిపదవి కూడా ఇవ్వలేదని, తన కుటుంబంలోని నలుగురే అధికారాన్ని అనుభవిస్తున్నారని ఆమె విమర్శించారు.
అయితే శృతి నటన బాగున్నా. . ఆమె నటించిన సినిమాలు బాక్సాఫీస్ దగ్గర పెద్దగా అలరించలేక పోతున్నాయి.
గిన్నెలో నీళ్లు తీసుకుని మరిగించాలి, దాన్లో జామాకులను వేసి, మరో 20 నిమిషాలు మరిగించి, వడగట్టి చల్లారనివ్వాలి.

6. CODE SNIPPETS

6.1 BLEU



```
[ ] from google.colab import drive
    drive.mount('/content/drive')
```

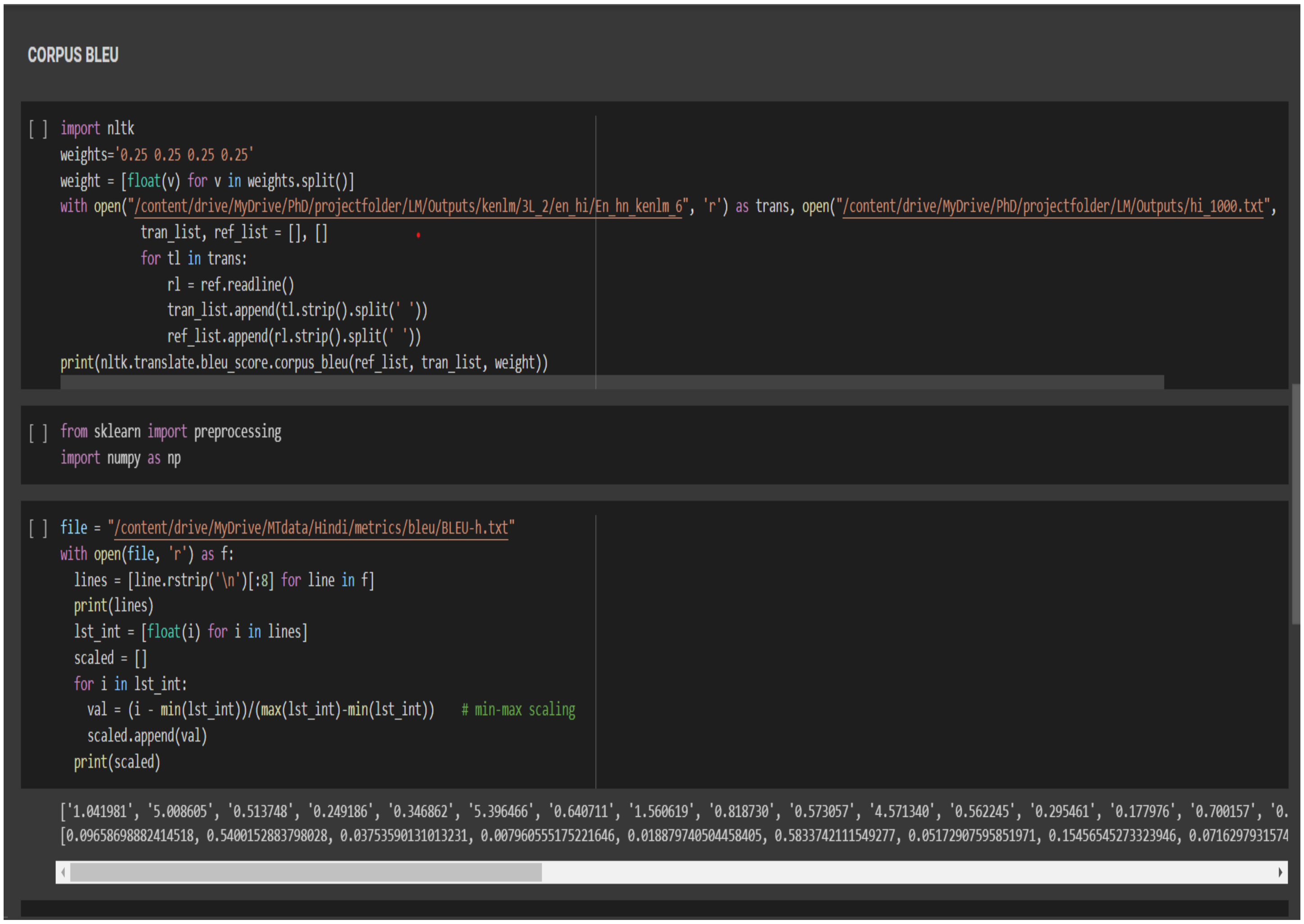
```
Mounted at /content/drive
```

```
import nltk
nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
True
```

SENETENCE WISE BLEU

```
[ ] from nltk import word_tokenize
    from nltk.translate.bleu_score import sentence_bleu

    translated_file = "/content/drive/MyDrive/MTdata/Hindi/hypothesis(machine).txt"
    reference_file = "/content/drive/MyDrive/MTdata/Hindi/references(human).txt"


    f5 = open('/content/BLEU.txt', "a+")
    with open(translated_file, "r") as f1, open(reference_file, "r") as f2:
      lines1 = [line.rstrip('\n') for line in f1]
      #print(lines1)
      lines2 = [line.rstrip('\n') for line in f2]
      #print(lines2)
      for l1,l2 in zip(lines1,lines2):
        tokens_ref=[word_tokenize(l2)]
        tokens_translate=word_tokenize(l1)
        score = sentence_bleu(tokens_ref, tokens_translate, weights=(0.25, 0.25, 0.25, 0.25))
        f5.write(str(score)+'\n')
    f5.close()
```



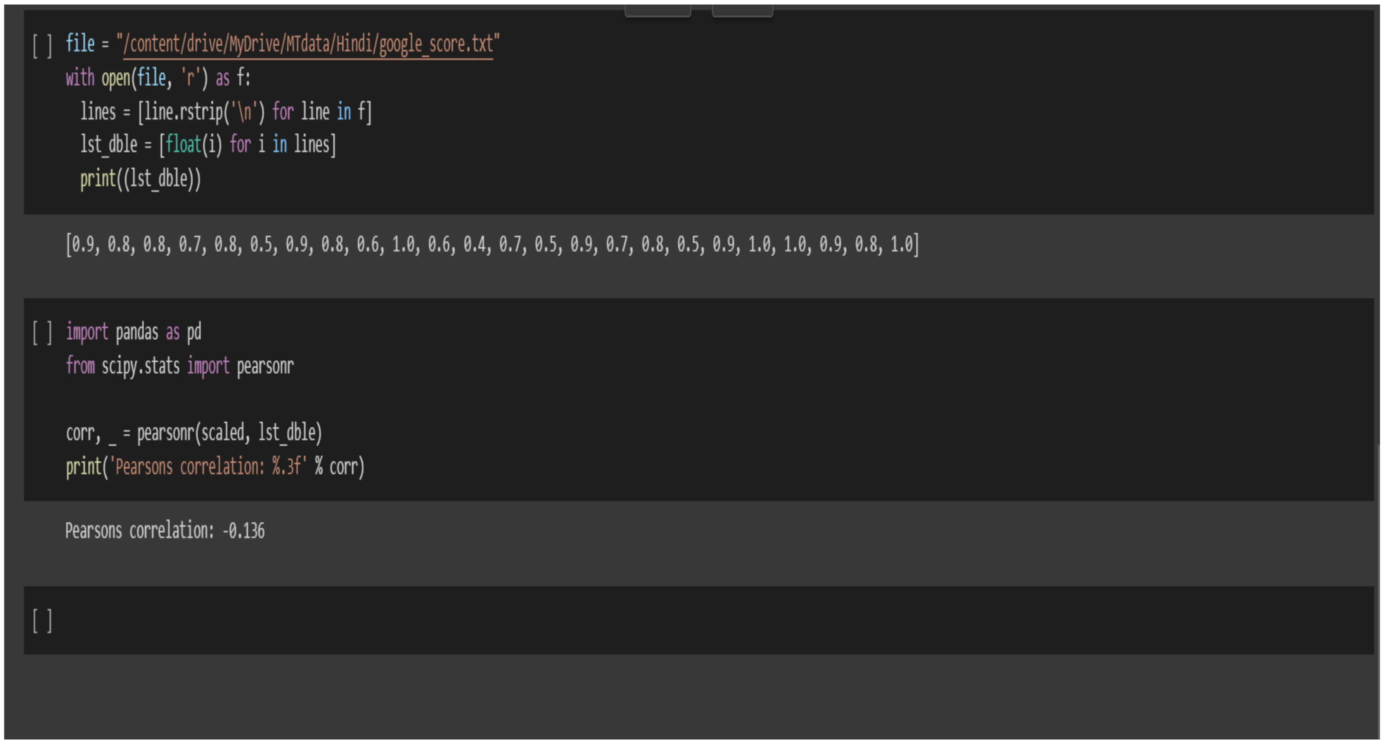
CORPUS BLEU

```
[ ] import nltk
    weights='0.25 0.25 0.25 0.25'
    weight = [float(v) for v in weights.split()]
    with open("/content/drive/MyDrive/PhD/projectfolder/LM/Outputs/kenlm/3L_2/en_hi/En_hn_kenlm_6", 'r') as trans, open("/content/drive/MyDrive/PhD/projectfolder/LM/Outputs/hi_1000.txt",
            tran_list, ref_list = [], []
            for tl in trans:
                rl = ref.readline()
                tran_list.append(tl.strip().split(' '))
                ref_list.append(rl.strip().split(' '))
    print(nltk.translate.bleu_score.corpus_bleu(ref_list, tran_list, weight))
```

```
[ ] from sklearn import preprocessing
    import numpy as np
```

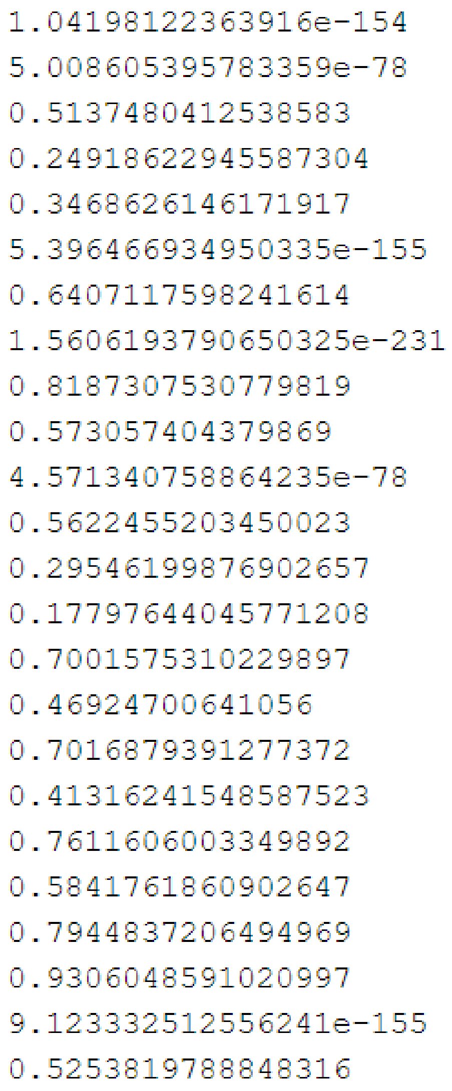
```
[ ] file = "/content/drive/MyDrive/MTdata/Hindi/metrics/bleu/BLEU-h.txt"
    with open(file, 'r') as f:
      lines = [line.rstrip('\n')[:8] for line in f]
      print(lines)
      lst_int = [float(i) for i in lines]
      scaled = []
      for i in lst_int:
        val = (i - min(lst_int))/(max(lst_int)-min(lst_int))    # min-max scaling
        scaled.append(val)
      print(scaled)
```

```
['1.041981', '5.008605', '0.513748', '0.249186', '0.346862', '5.396466', '0.640711', '1.560619', '0.818730', '0.573057', '4.571340', '0.562245', '0.295461', '0.177976', '0.700157', '0.
[0.09658698882414518, 0.5400152883798028, 0.03753590131013231, 0.007960555175221646, 0.018879740504458405, 0.5833742111549277, 0.05172907595851971, 0.15456545273323946, 0.0716297931574
```



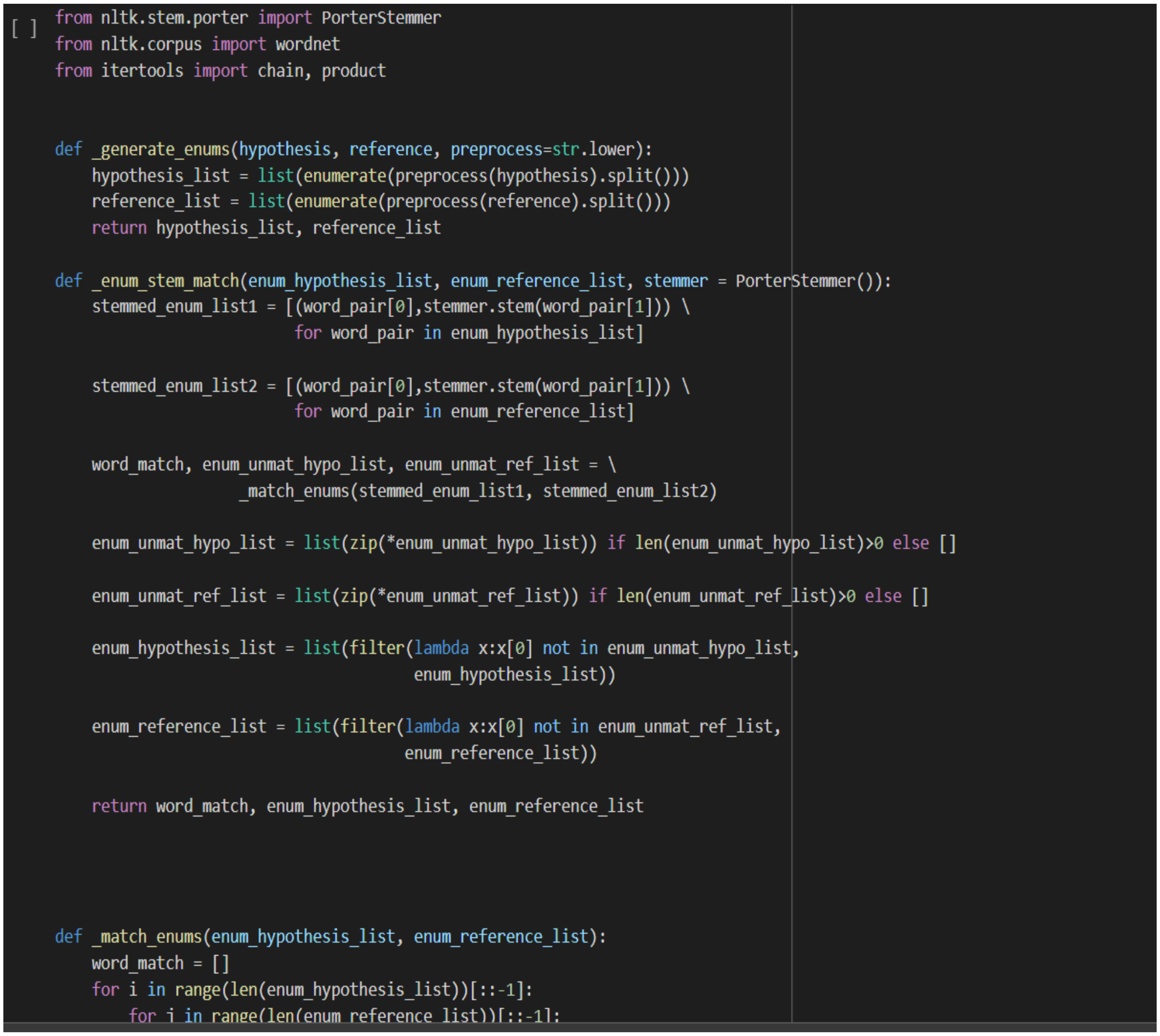
```
[ ] file = "/content/drive/MyDrive/MTdata/Hindi/google_score.txt"
    with open(file, 'r') as f:
      lines = [line.rstrip('\n') for line in f]
      lst_dble = [float(i) for i in lines]
      print((lst_dble))

    [0.9, 0.8, 0.8, 0.7, 0.8, 0.5, 0.9, 0.8, 0.6, 1.0, 0.6, 0.4, 0.7, 0.5, 0.9, 0.7, 0.8, 0.5, 0.9, 1.0, 1.0, 0.9, 0.8, 1.0]

[ ] import pandas as pd
    from scipy.stats import pearsonr

    corr, _ = pearsonr(scaled, lst_dble)
    print('Pearsons correlation: %.3f' % corr)

    Pearsons correlation: -0.136

[ ]
```

6.2 BLEU SCORE

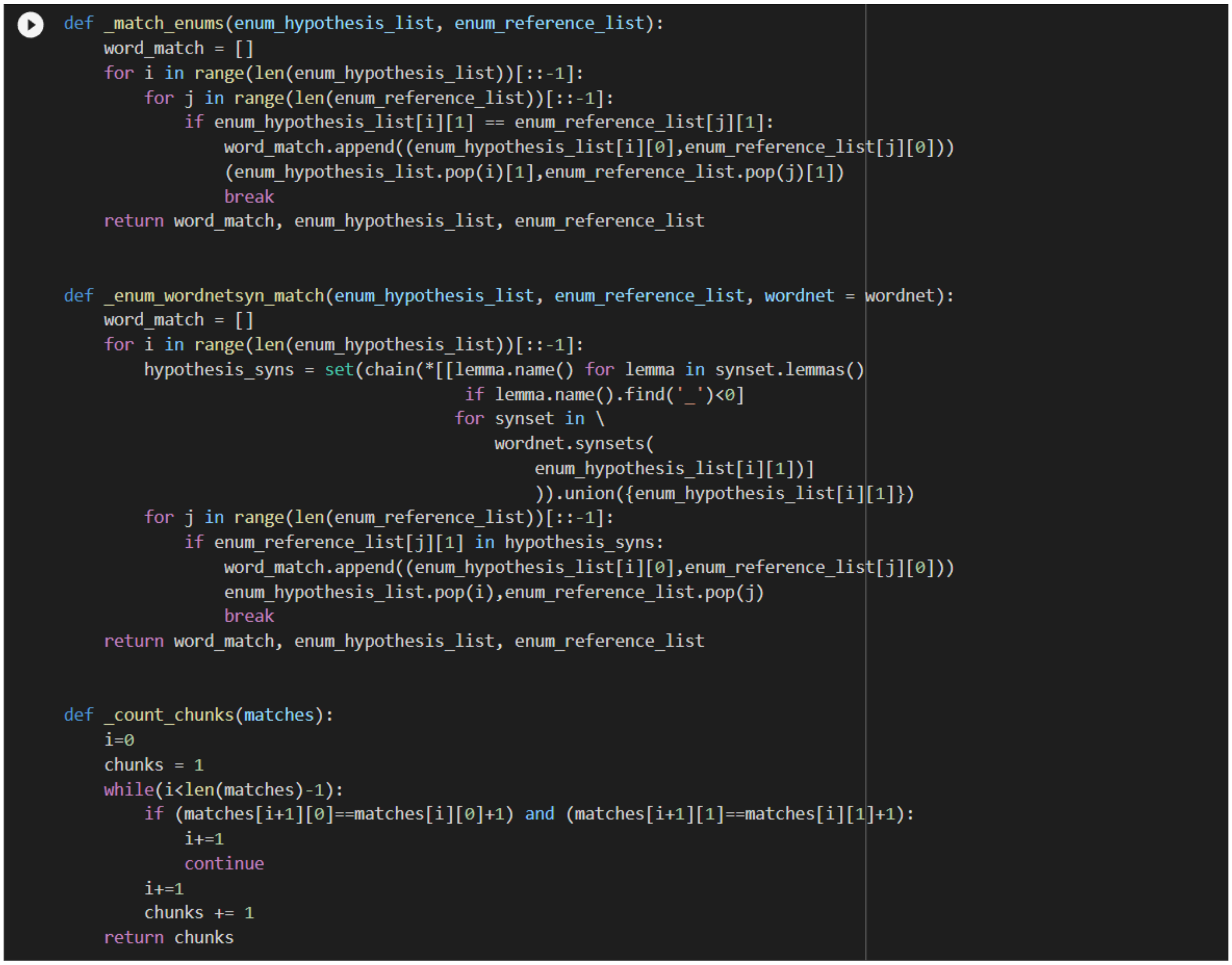


```
1.04198122363916e-154
5.008605395783359e-78
0.5137480412538583
0.24918622945587304
0.346862614617191 7
5.396466934950335e-155
0.6407117598241614
1.5606193790650325e-231
0.8187307530779819
0.573057404379869
4.571340758864235e-78
0.5622455203450023
0.29546199876902657
0.17797644045771208
0.7001575310229897
0.46924700641056
0.7016879391277372
0.41316241548587523
0.7611606003349892
0.5841761860902647
0.7944837206494969
0.9306048591020997
9.123332512556241e-155
0.5253819788848316
```

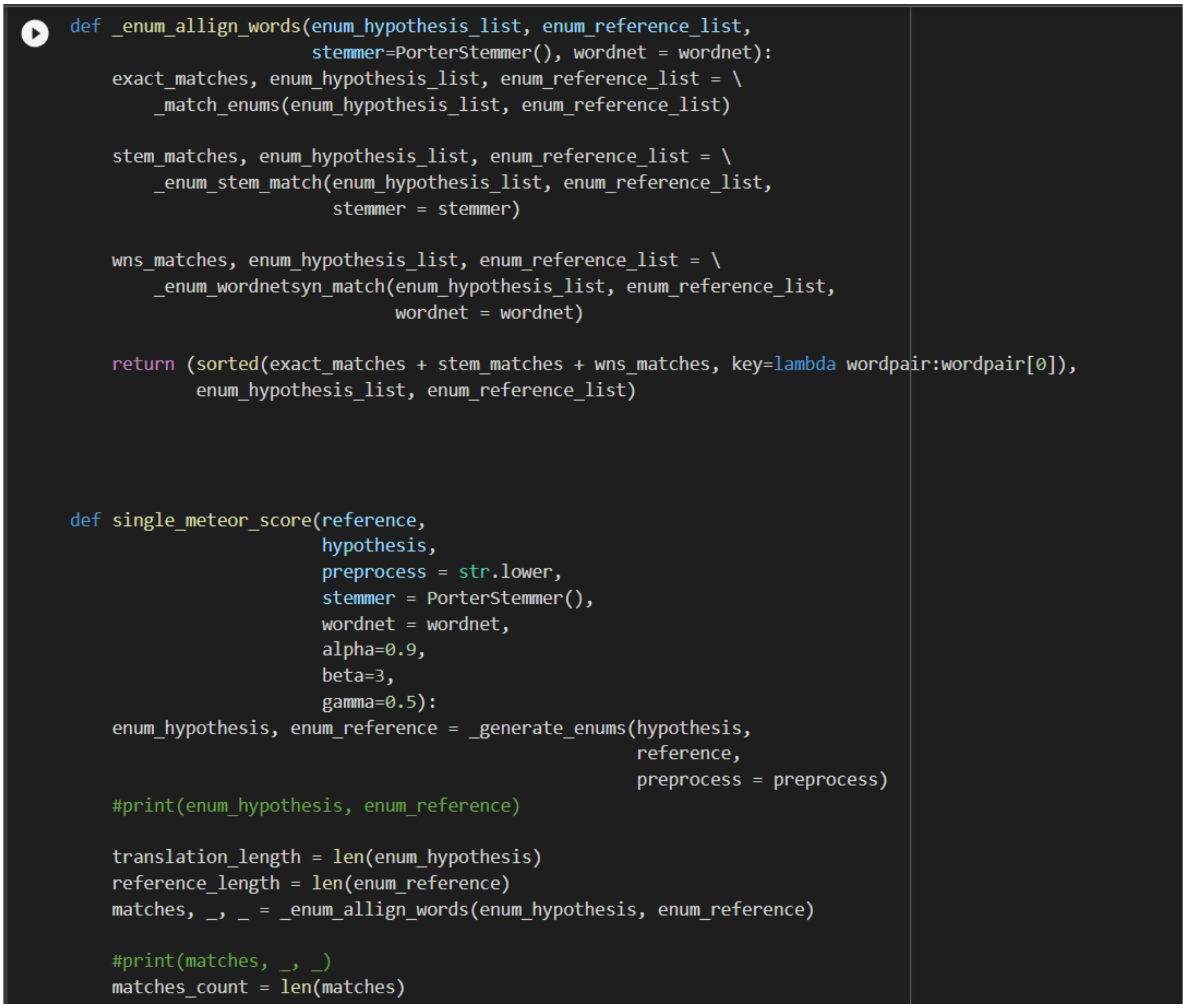
6.3 METEOR CODE



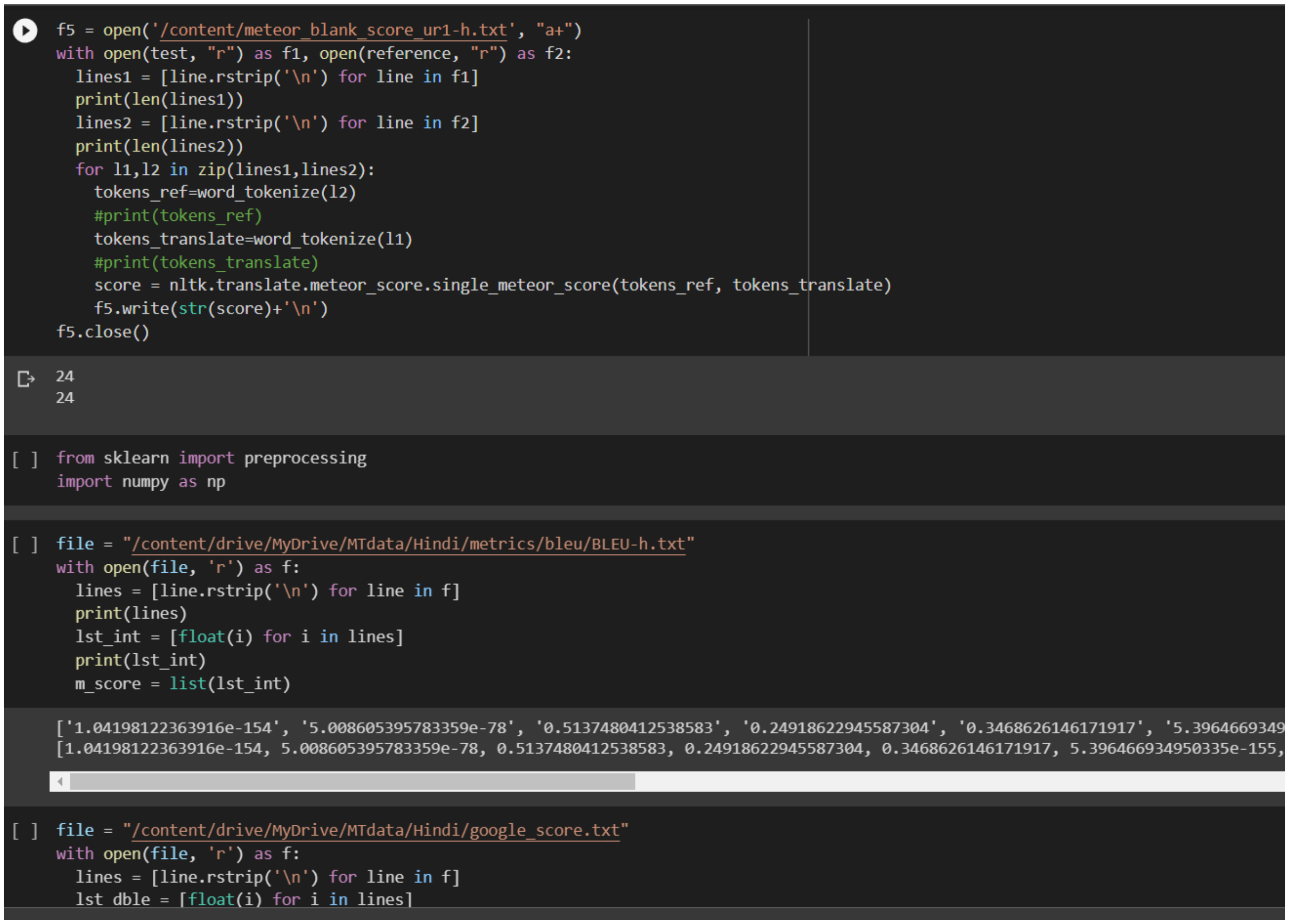
```
[ ] from nltk.stem.porter import PorterStemmer
    from nltk.corpus import wordnet
    from itertools import chain, product


    def _generate_enums(hypothesis, reference, preprocess=str.lower):
        hypothesis_list = list(enumerate(preprocess(hypothesis).split()))
        reference_list = list(enumerate(preprocess(reference).split()))
        return hypothesis_list, reference_list

    def _enum_stem_match(enum_hypothesis_list, enum_reference_list, stemmer = PorterStemmer()):
        stemmed_enum_list1 = [(word_pair[0],stemmer.stem(word_pair[1])) \
                              for word_pair in enum_hypothesis_list]

        stemmed_enum_list2 = [(word_pair[0],stemmer.stem(word_pair[1])) \
                              for word_pair in enum_reference_list]

        word_match, enum_unmat_hypo_list, enum_unmat_ref_list = \
                        _match_enums(stemmed_enum_list1, stemmed_enum_list2)

        enum_unmat_hypo_list = list(zip(*enum_unmat_hypo_list)) if len(enum_unmat_hypo_list)>0 else []

        enum_unmat_ref_list = list(zip(*enum_unmat_ref_list)) if len(enum_unmat_ref_list)>0 else []

        enum_hypothesis_list = list(filter(lambda x:x[0] not in enum_unmat_hypo_list,
                                           enum_hypothesis_list))

        enum_reference_list = list(filter(lambda x:x[0] not in enum_unmat_ref_list,
                                          enum_reference_list))

        return word_match, enum_hypothesis_list, enum_reference_list




    def _match_enums(enum_hypothesis_list, enum_reference_list):
        word_match = []
        for i in range(len(enum_hypothesis_list))[::-1]:
            for j in range(len(enum_reference_list))[::-1]:
```



```
def _match_enums(enum_hypothesis_list, enum_reference_list):
    word_match = []
    for i in range(len(enum_hypothesis_list))[::-1]:
        for j in range(len(enum_reference_list))[::-1]:
            if enum_hypothesis_list[i][1] == enum_reference_list[j][1]:
                word_match.append((enum_hypothesis_list[i][0],enum_reference_list[j][0]))
                (enum_hypothesis_list.pop(i)[1],enum_reference_list.pop(j)[1])
                break
    return word_match, enum_hypothesis_list, enum_reference_list


def _enum_wordnetsyn_match(enum_hypothesis_list, enum_reference_list, wordnet = wordnet):
    word_match = []
    for i in range(len(enum_hypothesis_list))[::-1]:
        hypothesis_syns = set(chain(*[[lemma.name() for lemma in synset.lemmas()
                                        if lemma.name().find('_')<0]
                                       for synset in \
                                           wordnet.synsets(
                                               enum_hypothesis_list[i][1])]
                                               )).union({enum_hypothesis_list[i][1]})
        for j in range(len(enum_reference_list))[::-1]:
            if enum_reference_list[j][1] in hypothesis_syns:
                word_match.append((enum_hypothesis_list[i][0],enum_reference_list[j][0]))
                enum_hypothesis_list.pop(i),enum_reference_list.pop(j)
                break
    return word_match, enum_hypothesis_list, enum_reference_list


def _count_chunks(matches):
    i=0
    chunks = 1
    while(i<len(matches)-1):
        if (matches[i+1][0]==matches[i][0]+1) and (matches[i+1][1]==matches[i][1]+1):
            i+=1
            continue
        i+=1
        chunks += 1
    return chunks
```



```python
def _enum_allign_words(enum_hypothesis_list, enum_reference_list,
                       stemmer=PorterStemmer(), wordnet = wordnet):
    exact_matches, enum_hypothesis_list, enum_reference_list = \
        _match_enums(enum_hypothesis_list, enum_reference_list)

    stem_matches, enum_hypothesis_list, enum_reference_list = \
        _enum_stem_match(enum_hypothesis_list, enum_reference_list,
                         stemmer = stemmer)

    wns_matches, enum_hypothesis_list, enum_reference_list = \
        _enum_wordnetsyn_match(enum_hypothesis_list, enum_reference_list,
                               wordnet = wordnet)

    return (sorted(exact_matches + stem_matches + wns_matches, key=lambda wordpair:wordpair[0]),
            enum_hypothesis_list, enum_reference_list)



def single_meteor_score(reference,
                        hypothesis,
                        preprocess = str.lower,
                        stemmer = PorterStemmer(),
                        wordnet = wordnet,
                        alpha=0.9,
                        beta=3,
                        gamma=0.5):
    enum_hypothesis, enum_reference = _generate_enums(hypothesis,
                                                      reference,
                                                      preprocess = preprocess)
    #print(enum_hypothesis, enum_reference)

    translation_length = len(enum_hypothesis)
    reference_length = len(enum_reference)
    matches, _, _ = _enum_allign_words(enum_hypothesis, enum_reference)

    #print(matches, _, _)
    matches_count = len(matches)
```



```
f5 = open('/content/meteor_blank_score_ur1-h.txt', "a+")
with open(test, "r") as f1, open(reference, "r") as f2:
  lines1 = [line.rstrip('\n') for line in f1]
  print(len(lines1))
  lines2 = [line.rstrip('\n') for line in f2]
  print(len(lines2))
  for l1,l2 in zip(lines1,lines2):
    tokens_ref=word_tokenize(l2)
    #print(tokens_ref)
    tokens_translate=word_tokenize(l1)
    #print(tokens_translate)
    score = nltk.translate.meteor_score.single_meteor_score(tokens_ref, tokens_translate)
    f5.write(str(score)+'\n')
f5.close()
```

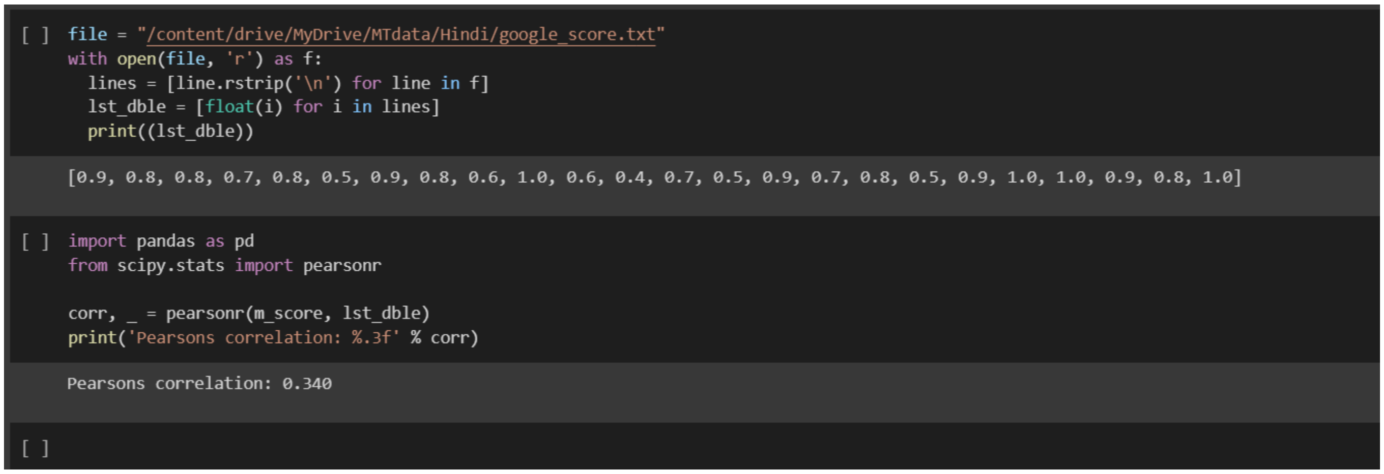
```
24
24
```

```
[ ] from sklearn import preprocessing
    import numpy as np
```

```
[ ] file = "/content/drive/MyDrive/MTdata/Hindi/metrics/bleu/BLEU-h.txt"
    with open(file, 'r') as f:
      lines = [line.rstrip('\n') for line in f]
      print(lines)
      lst_int = [float(i) for i in lines]
      print(lst_int)
      m_score = list(lst_int)
```

```
['1.04198122363916e-154', '5.008605395783359e-78', '0.5137480412538583', '0.24918622945587304', '0.3468626146171917', '5.3964669349
[1.04198122363916e-154, 5.008605395783359e-78, 0.5137480412538583, 0.24918622945587304, 0.3468626146171917, 5.396466934950335e-155,
```

```
[ ] file = "/content/drive/MyDrive/MTdata/Hindi/google_score.txt"
    with open(file, 'r') as f:
      lines = [line.rstrip('\n') for line in f]
      lst_dble = [float(i) for i in lines]
```



```
[ ] file = "/content/drive/MyDrive/MTdata/Hindi/google_score.txt"
    with open(file, 'r') as f:
      lines = [line.rstrip('\n') for line in f]
      lst_dble = [float(i) for i in lines]
      print((lst_dble))
```

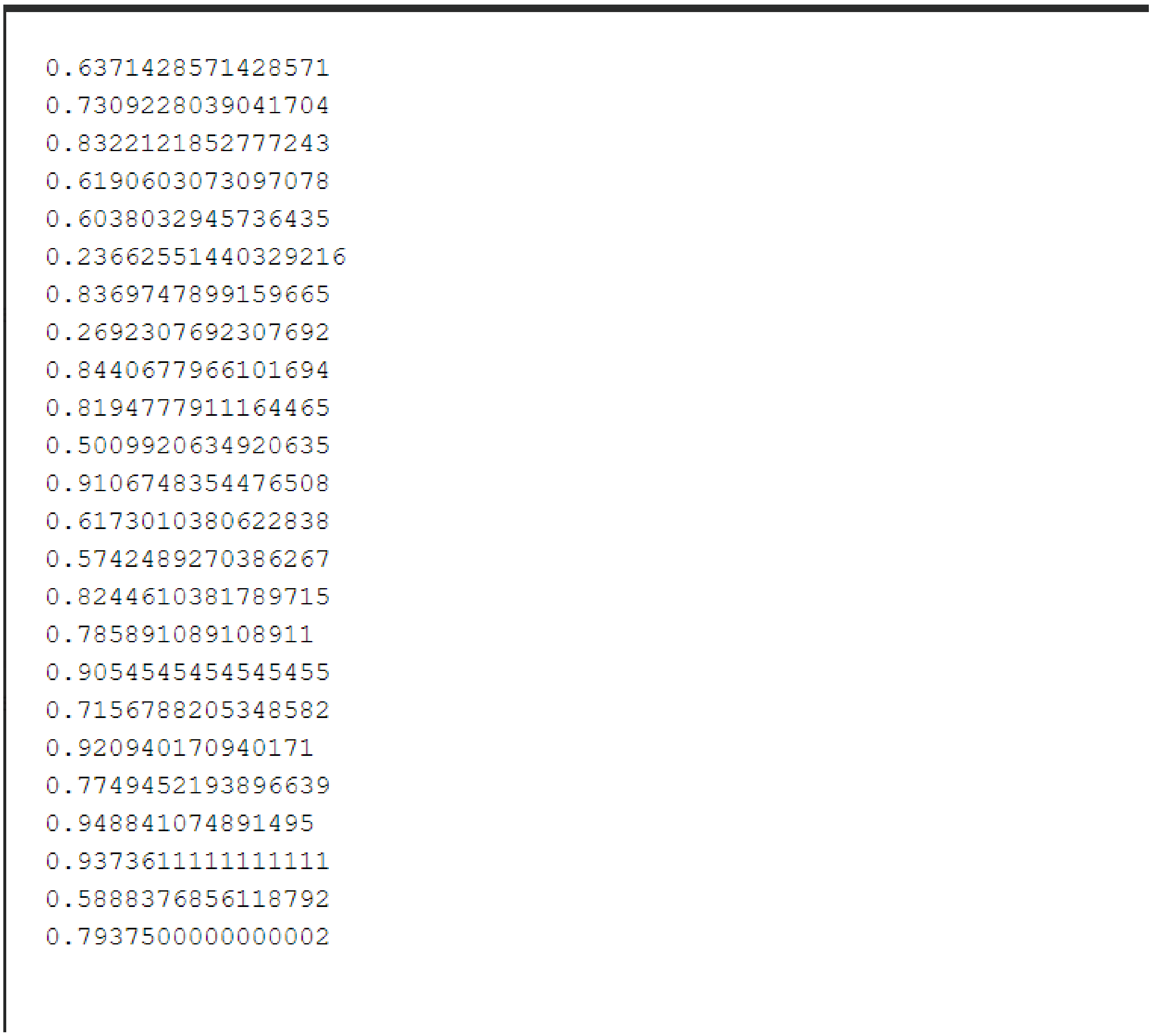
```
[0.9, 0.8, 0.8, 0.7, 0.8, 0.5, 0.9, 0.8, 0.6, 1.0, 0.6, 0.4, 0.7, 0.5, 0.9, 0.7, 0.8, 0.5, 0.9, 1.0, 1.0, 0.9, 0.8, 1.0]
```

```
[ ] import pandas as pd
    from scipy.stats import pearsonr

    corr, _ = pearsonr(m_score, lst_dble)
    print('Pearsons correlation: %.3f' % corr)
```

```
Pearsons correlation: 0.340
```

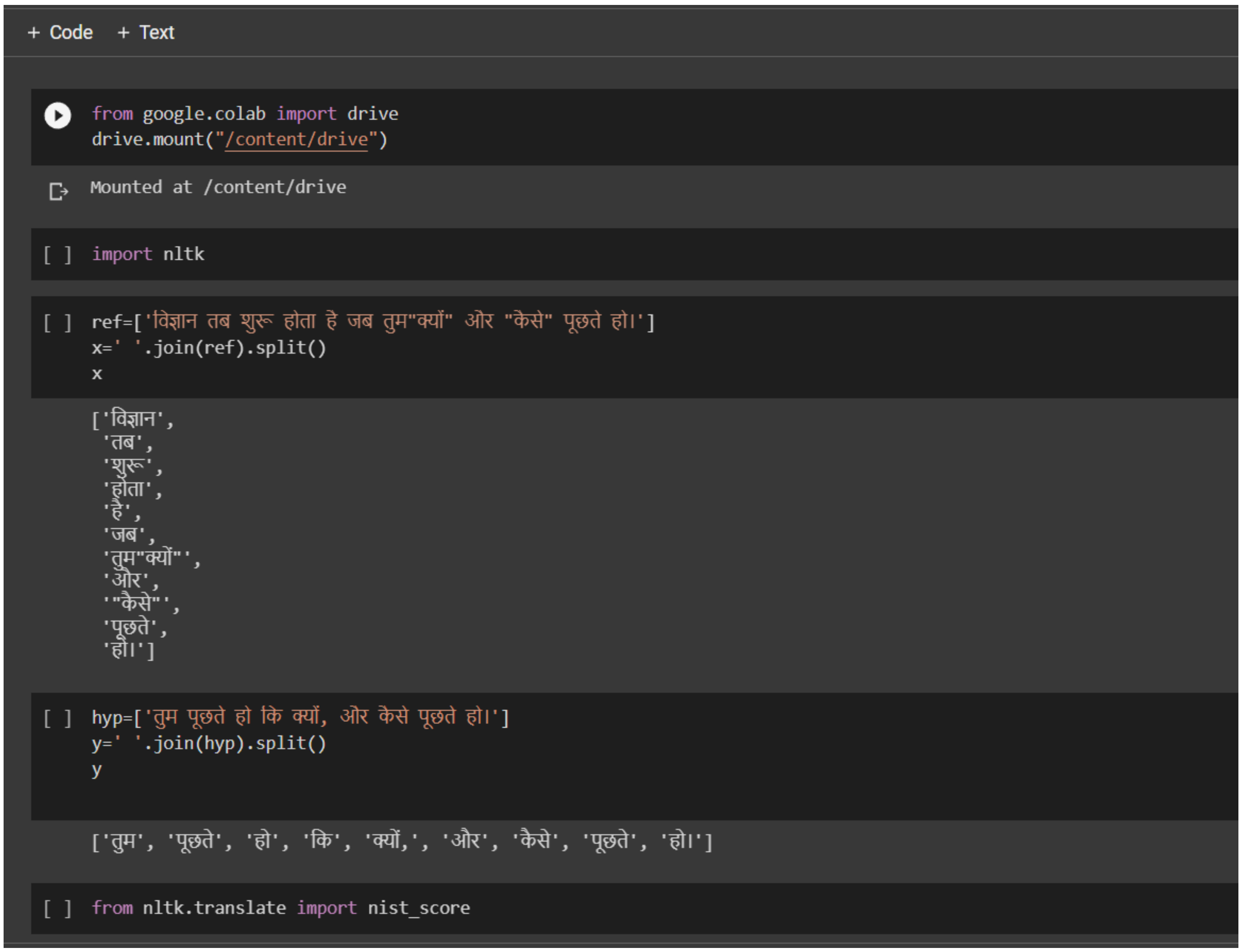
```
[ ]
```

6.4 METEOR SCORE



```
0.6371428571428571
0.7309228039041704
0.8322121852777243
0.6190603073097078
0.6038032945736435
0.23662551440329216
0.8369747899159665
0.2692307692307692
0.8440677966101694
0.8194777911164465
0.5009920634920635
0.9106748354476508
0.6173010380622838
0.5742489270386267
0.8244610381789715
0.785891089108911
0.9054545454545455
0.7156788205348582
0.920940170940171
0.7749452193896639
0.948841074891495
0.9373611111111111
0.5888376856118792
0.7937500000000002
```

6.5 NIST CODE



```
from google.colab import drive
drive.mount("/content/drive")
```

```
Mounted at /content/drive
```

```
import nltk
```

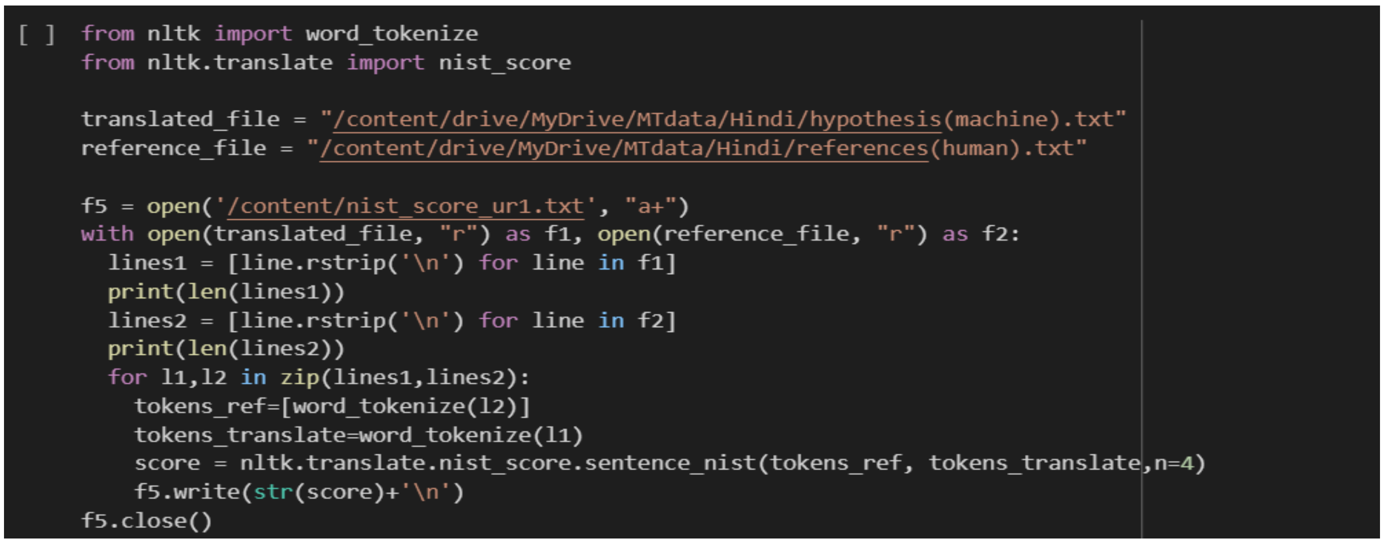
```
ref=['विज्ञान तब शुरू होता है जब तुम"क्यों" और "कैसे" पूछते हो।']
x=' '.join(ref).split()
x
```

```
['विज्ञान',
 'तब',
 'शुरू',
 'होता',
 'है',
 'जब',
 'तुम"क्यों"',
 'और',
 '"कैसे"',
 'पूछते',
 'हो।']
```

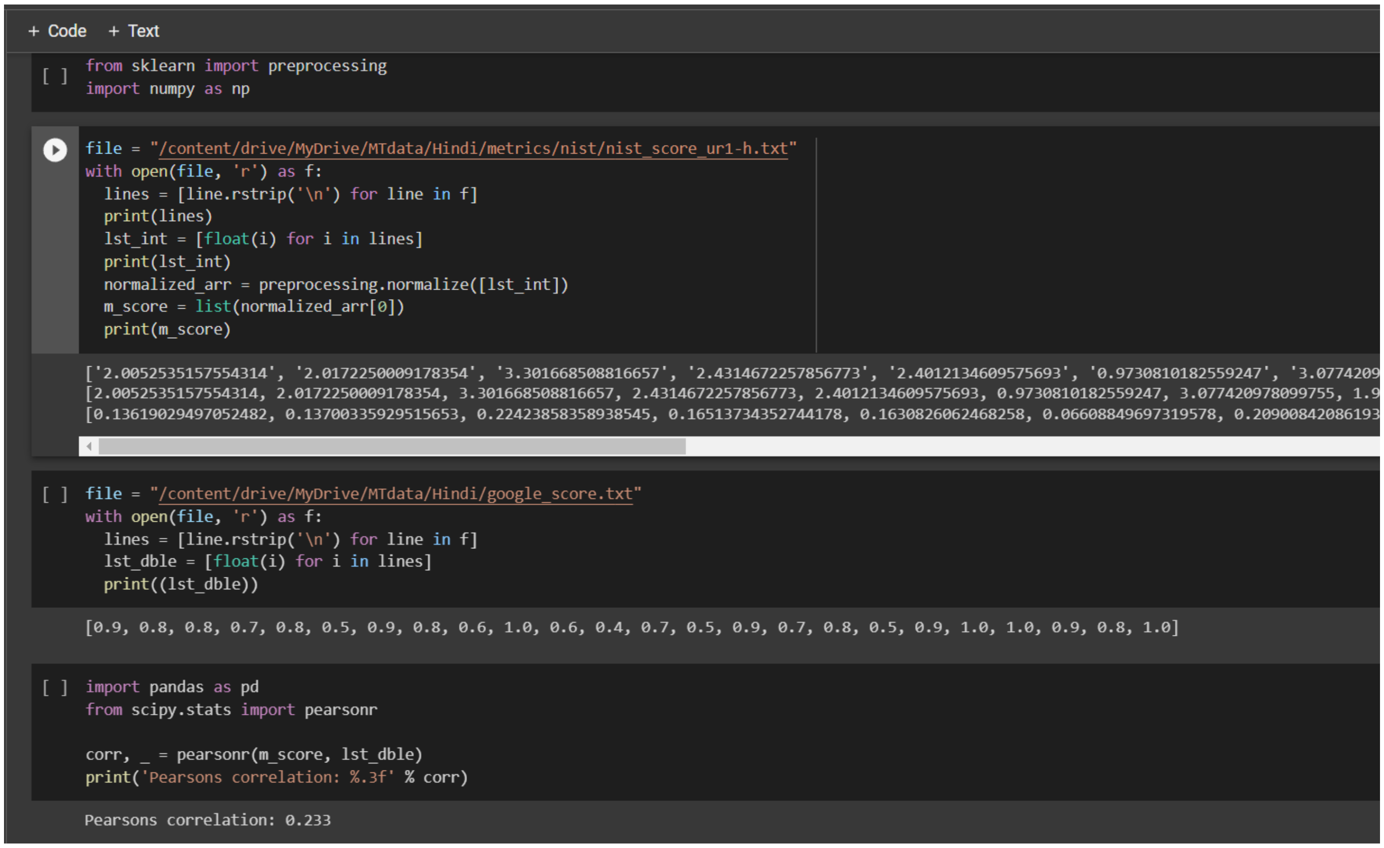
```
hyp=['तुम पूछते हो कि क्यों, और कैसे पूछते हो।']
y=' '.join(hyp).split()
y
```

```
['तुम', 'पूछते', 'हो', 'कि', 'क्यों,', 'और', 'कैसे', 'पूछते', 'हो।']
```

```
from nltk.translate import nist_score
```



```
from nltk import word_tokenize
from nltk.translate import nist_score

translated_file = "/content/drive/MyDrive/MTdata/Hindi/hypothesis(machine).txt"
reference_file = "/content/drive/MyDrive/MTdata/Hindi/references(human).txt"

f5 = open('/content/nist_score_ur1.txt', "a+")
with open(translated_file, "r") as f1, open(reference_file, "r") as f2:
  lines1 = [line.rstrip('\n') for line in f1]
  print(len(lines1))
  lines2 = [line.rstrip('\n') for line in f2]
  print(len(lines2))
  for l1,l2 in zip(lines1,lines2):
    tokens_ref=[word_tokenize(l2)]
    tokens_translate=word_tokenize(l1)
    score = nltk.translate.nist_score.sentence_nist(tokens_ref, tokens_translate,n=4)
    f5.write(str(score)+'\n')
f5.close()
```



```
from sklearn import preprocessing
import numpy as np
```

```
file = "/content/drive/MyDrive/MTdata/Hindi/metrics/nist/nist_score_ur1-h.txt"
with open(file, 'r') as f:
  lines = [line.rstrip('\n') for line in f]
  print(lines)
  lst_int = [float(i) for i in lines]
  print(lst_int)
  normalized_arr = preprocessing.normalize([lst_int])
  m_score = list(normalized_arr[0])
  print(m_score)
```

```
['2.0052535157554314', '2.0172250009178354', '3.301668508816657', '2.4314672257856773', '2.4012134609575693', '0.9730810182559247', '3.0774209
[2.0052535157554314, 2.0172250009178354, 3.301668508816657, 2.4314672257856773, 2.4012134609575693, 0.9730810182559247, 3.077420978099755, 1.9
[0.13619029497052482, 0.13700335929515653, 0.22423858358938545, 0.16513734352744178, 0.1630826062468258, 0.06608849697319578, 0.20900842086193
```

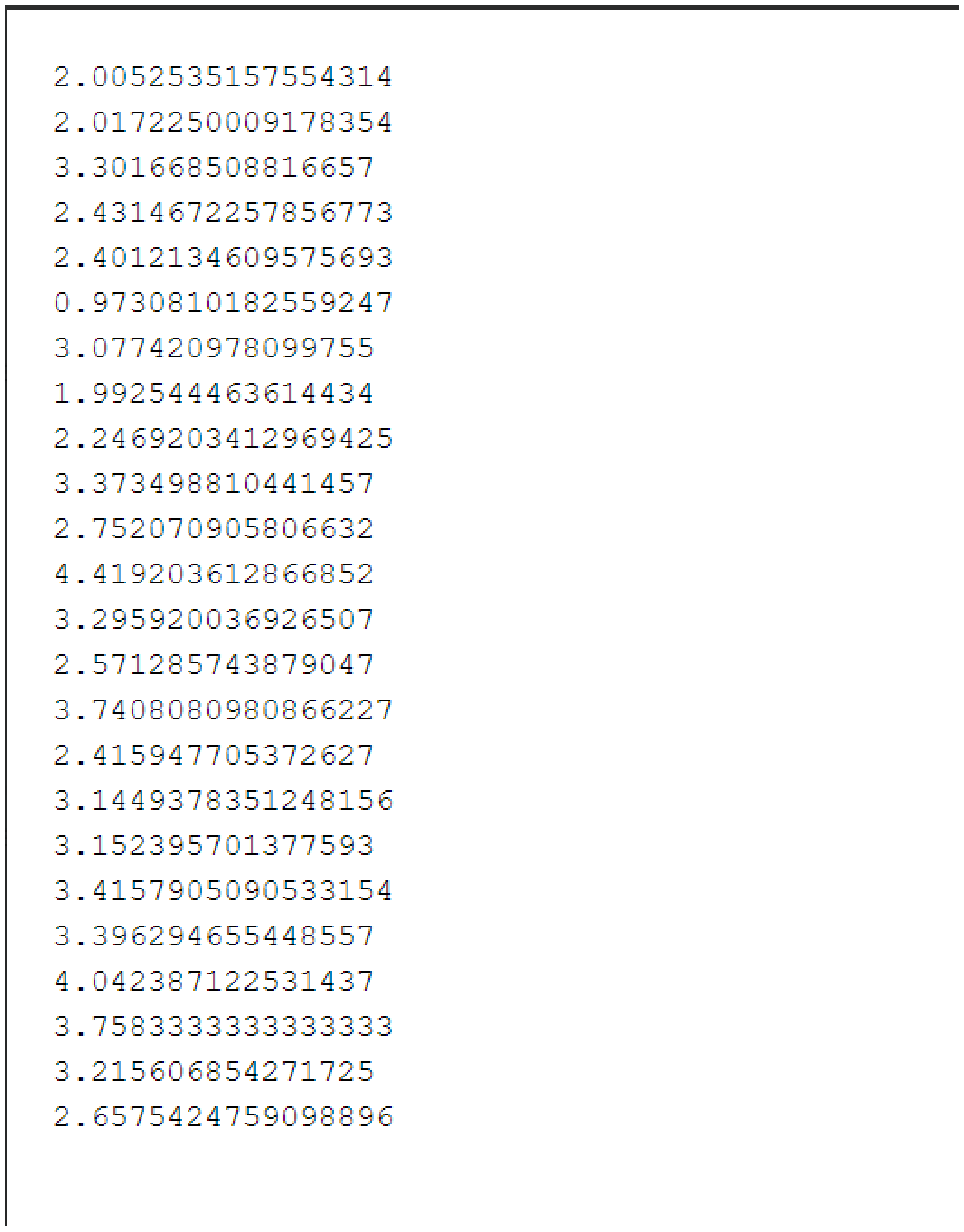
```
file = "/content/drive/MyDrive/MTdata/Hindi/google_score.txt"
with open(file, 'r') as f:
  lines = [line.rstrip('\n') for line in f]
  lst_dble = [float(i) for i in lines]
  print((lst_dble))
```

```
[0.9, 0.8, 0.8, 0.7, 0.8, 0.5, 0.9, 0.8, 0.6, 1.0, 0.6, 0.4, 0.7, 0.5, 0.9, 0.7, 0.8, 0.5, 0.9, 1.0, 1.0, 0.9, 0.8, 1.0]
```

```
import pandas as pd
from scipy.stats import pearsonr

corr, _ = pearsonr(m_score, lst_dble)
print('Pearsons correlation: %.3f' % corr)
```

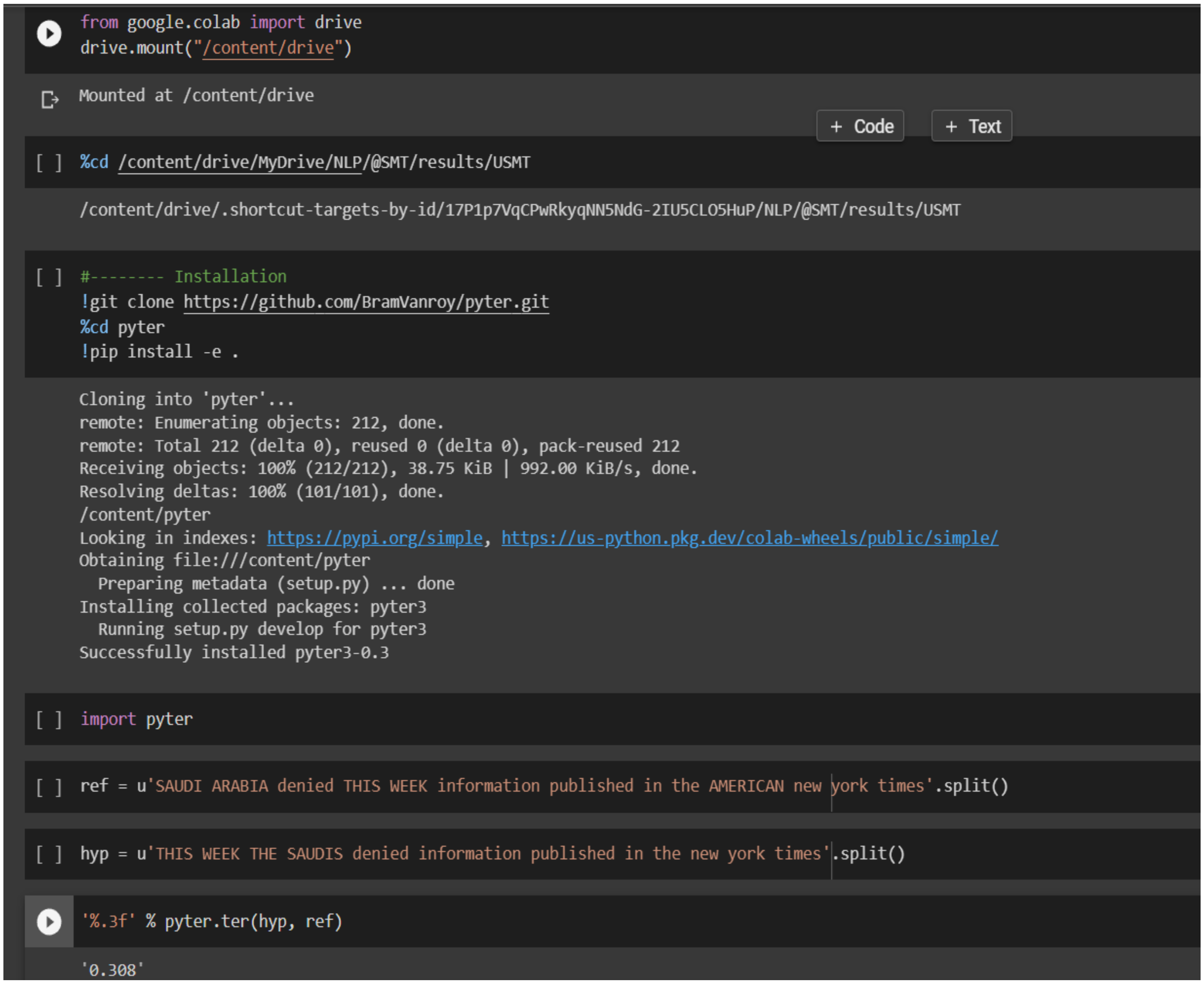
```
Pearsons correlation: 0.233
```

6.6 NIST SCORE



```
2.0052535157554314
2.0172250009178354
3.301668508816657
2.4314672257856773
2.4012134609575693
0.9730810182559247
3.077420978099755
1.992544463614434
2.2469203412969425
3.373498810441457
2.752070905806632
4.419203612866852
3.295920036926507
2.571285743879047
3.7408080980866227
2.415947705372627
3.1449378351248156
3.152395701377593
3.4157905090533154
3.396294655448557
4.042387122531437
3.7583333333333333
3.215606854271725
2.6575424759098896
```

6.7 TER CODE



```
from google.colab import drive
drive.mount("/content/drive")
```

```
Mounted at /content/drive
```

```
%cd /content/drive/MyDrive/NLP/@SMT/results/USMT
```

```
/content/drive/.shortcut-targets-by-id/17P1p7VqCPwRkyqNN5NdG-2IU5CLO5HuP/NLP/@SMT/results/USMT
```

```
#-------- Installation
!git clone https://github.com/BramVanroy/pyter.git
%cd pyter
!pip install -e .
```

```
Cloning into 'pyter'...
remote: Enumerating objects: 212, done.
remote: Total 212 (delta 0), reused 0 (delta 0), pack-reused 212
Receiving objects: 100% (212/212), 38.75 KiB | 992.00 KiB/s, done.
Resolving deltas: 100% (101/101), done.
/content/pyter
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Obtaining file:///content/pyter
  Preparing metadata (setup.py) ... done
Installing collected packages: pyter3
  Running setup.py develop for pyter3
Successfully installed pyter3-0.3
```

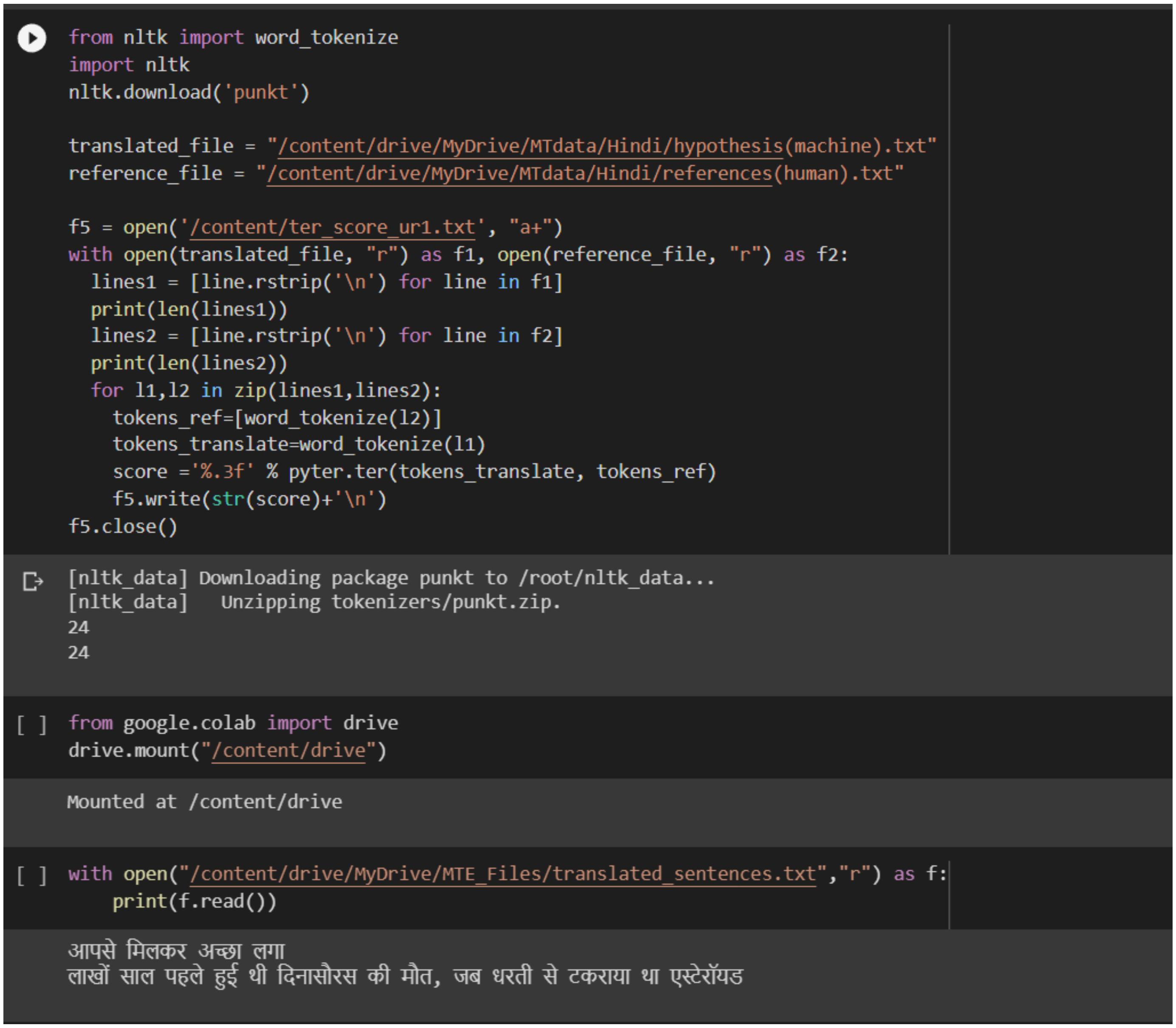
```
import pyter
```

```
ref = u'SAUDI ARABIA denied THIS WEEK information published in the AMERICAN new york times'.split()
```

```
hyp = u'THIS WEEK THE SAUDIS denied information published in the new york times'.split()
```

```
'%.3f' % pyter.ter(hyp, ref)
```

```
'0.308'
```



```
from nltk import word_tokenize
import nltk
nltk.download('punkt')

translated_file = "/content/drive/MyDrive/MTdata/Hindi/hypothesis(machine).txt"
reference_file = "/content/drive/MyDrive/MTdata/Hindi/references(human).txt"

f5 = open('/content/ter_score_ur1.txt', "a+")
with open(translated_file, "r") as f1, open(reference_file, "r") as f2:
  lines1 = [line.rstrip('\n') for line in f1]
  print(len(lines1))
  lines2 = [line.rstrip('\n') for line in f2]
  print(len(lines2))
  for l1,l2 in zip(lines1,lines2):
    tokens_ref=[word_tokenize(l2)]
    tokens_translate=word_tokenize(l1)
    score ='%.3f' % pyter.ter(tokens_translate, tokens_ref)
    f5.write(str(score)+'\n')
f5.close()
```

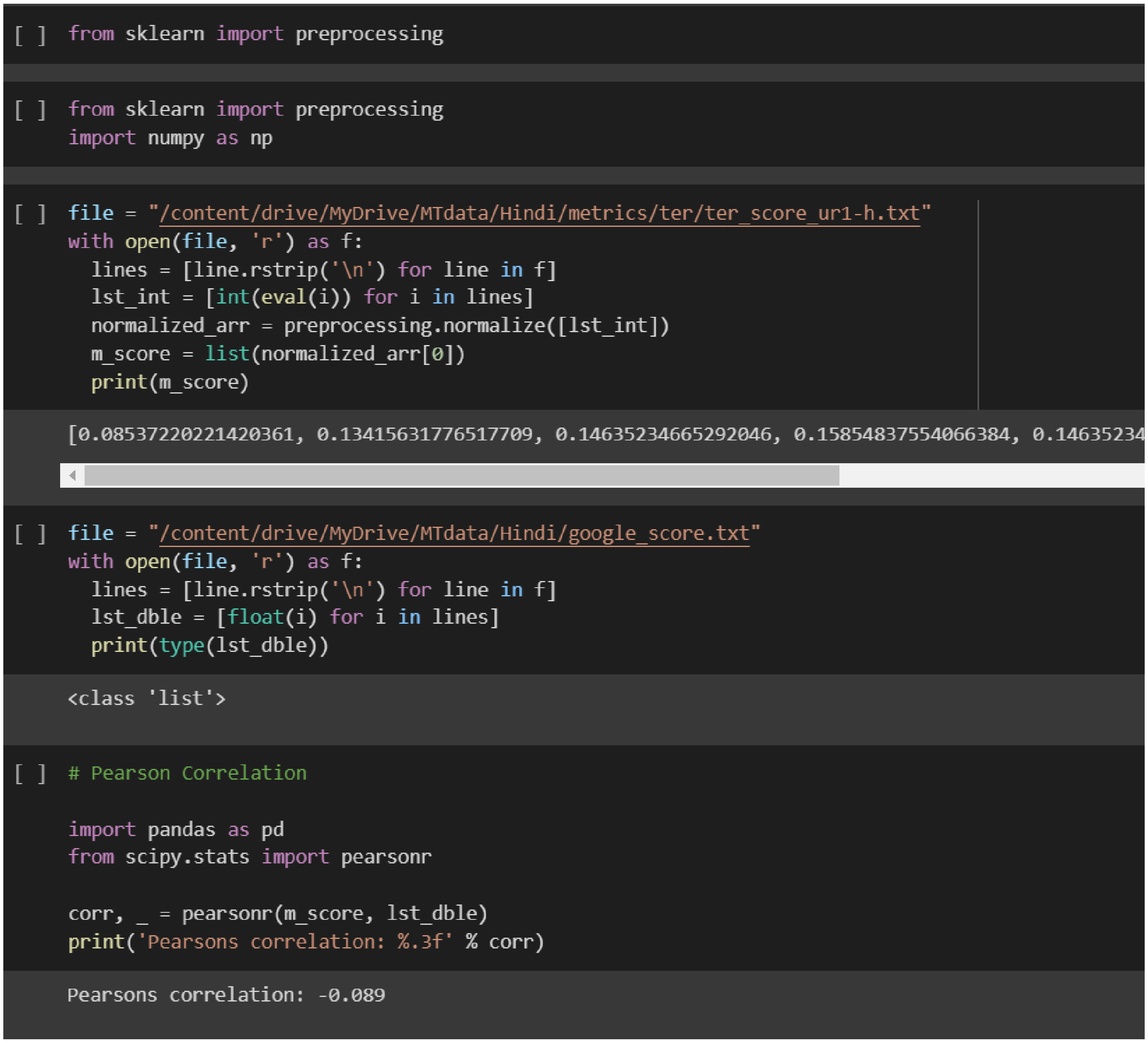
```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
24
24
```

```
[ ] from google.colab import drive
    drive.mount("/content/drive")
```

```
Mounted at /content/drive
```

```
[ ] with open("/content/drive/MyDrive/MTE_Files/translated_sentences.txt","r") as f:
      print(f.read())
```

```
आपसे मिलकर अच्छा लगा
लाखों साल पहले हुई थी दिनासौरस की मौत, जब धरती से टकराया था एस्टेरॉयड
```



```
[ ] from sklearn import preprocessing
```

```
[ ] from sklearn import preprocessing
    import numpy as np
```

```
[ ] file = "/content/drive/MyDrive/MTdata/Hindi/metrics/ter/ter_score_ur1-h.txt"
    with open(file, 'r') as f:
      lines = [line.rstrip('\n') for line in f]
      lst_int = [int(eval(i)) for i in lines]
      normalized_arr = preprocessing.normalize([lst_int])
      m_score = list(normalized_arr[0])
      print(m_score)
```

```
[0.08537220221420361, 0.13415631776517709, 0.14635234665292046, 0.15854837554066384, 0.14635234
```

```
[ ] file = "/content/drive/MyDrive/MTdata/Hindi/google_score.txt"
    with open(file, 'r') as f:
      lines = [line.rstrip('\n') for line in f]
      lst_dble = [float(i) for i in lines]
      print(type(lst_dble))
```

```
<class 'list'>
```

```
[ ] # Pearson Correlation

    import pandas as pd
    from scipy.stats import pearsonr

    corr, _ = pearsonr(m_score, lst_dble)
    print('Pearsons correlation: %.3f' % corr)
```

```
Pearsons correlation: -0.089
```

6.8 TER SCORE

```
7.000
11.000
12.000
13.000
12.000
9.000
11.000
13.000
5.000
17.000
18.000
24.000
28.000
26.000
23.000
11.000
11.000
17.000
13.000
24.000
23.000
16.000
19.000
10.000
```

7. CONCLUSION

In conclusion, evaluation of machine translation systems is a crucial step towards improving their quality and effectiveness. While automatic evaluation metrics such as BLEU, METEOR, and TER provide a quick and objective evaluation of machine translation quality, they have limitations in capturing the nuances of translation quality. Therefore, researchers are exploring alternative evaluation methods that take into account human judgments of translation quality, domain-specificity, and multilingual scenarios. Expert evaluators are often employed to evaluate translations based on parameters such as fluency, adequacy, grammaticality, and coherence. Additionally, the advent of neural machine translation has revolutionized the field of machine translation evaluation, and researchers are exploring new evaluation metrics and methods to capture the quality of NMT systems accurately. In summary, a robust evaluation system that combines automatic evaluation metrics, human evaluation, domain-specific evaluation methods, and multilingual evaluation methods is essential for improving the quality and effectiveness of machine translation systems.

8. FUTURE WORK

In this report, we have presented a comparative analysis of different lexical automatic machine translation evaluation metrics for indic languages. We have evaluated the performance of these metrics on three different datasets of English - Hindi, English - Bengali and English - Telugu translation pairs.

As a future work, we plan to extend our study to other Indic languages and domains. We also aim to incorporate syntactic and pragmatic features to capture the structural and contextual aspects of translation quality. Furthermore, we intend to explore the correlation of the metrics with human judgments and conduct a user study to validate its usefulness and reliability.

Another possible direction for future work is to explore the use of neural machine translation (NMT) models for Indic languages and evaluate them using Lexical Automatic Machine Translation LAMT metrics. NMT models are based on deep learning techniques that learn to translate from large parallel corpora without relying on explicit rules or features. NMT models have shown remarkable results for high-resource languages, but their performance may degrade for low-resource languages, such as many Indic languages. Moreover, NMT models may face challenges in handling the long and complex sentences, the domain mismatch, and the data sparsity of Indic languages. Therefore, it is interesting to examine how NMT models perform for Indic languages and how they compare with statistical machine translation (SMT) models using LAMT metrics. We hope that our work will contribute to the advancement of machine translation research and evaluation for indic languages.

REFERENCES

[1] University CM The METEOR Automatic MT Evaluation Metric, https://www.cs.cmu.edu/~alavie/METEOR/index.html

[2]Indian Languages CI of Anu Kriti (archived feb 2021). archive.org, https://web.archive.org/web/20210211063742/https://www.anukriti.net/

[3]Singh M, Kumar R, Chana I (2021) Machine translation systems for Indian languages: review of modelling techniques, challenges, open issues and future research directions. Arch Comput Methods Eng 28(4):2165–2193

[4]Papineni, K., Roukos, S., Ward, T., & Zhu, W. J. (2002). BLEU: a method for automatic evaluation of machine translation. In Proceedings of the 40th annual meeting on association for computational linguistics (pp. 311-318).

[5]Dandapat, S., & Das, D. (2018). Evaluation of Machine Translation Systems: A Study on Indian Languages. In Proceedings of the 5th Workshop on Indian Language Data: Resources and Evaluation (pp. 25-34).

[6]Sharma, R., & Dave, M. (2021). Evaluation of Machine Translation Metrics for Hindi Language. In Proceedings of the 2021 12th International Conference on Computing, Communication and Networking Technologies (ICCCNT) (pp. 1-6).

[7]Saini, R., & Mittal, A. (2020). Analysis of Machine Translation Metrics for English to Indian Languages. In Proceedings of the 2020 7th International Conference on Signal Processing and Integrated Networks (SPIN) (pp. 41-46).

[8] Singh, J., & Gupta, V. (2016). Text stemming: Approaches, applications, and challenges. ACM Computing Surveys (CSUR), 49(3), 1-46.

[9] Lee, S., Lee, J., Moon, H., Park, C., Seo, J., Eo, S., ... & Lim, H. (2023). A Survey on Evaluation Metrics for Machine Translation. Mathematics, 11(4), 1006.

[10] Pérez-Ortiz, J. A., Sánchez-Martínez, F., Esplà-Gomis, M., Popović, M., Rico, C., Martins, A., ... & Forcada, M. L. (2018). Proceedings of the 21st Annual Conference of the European Association for Machine Translation: 28-30 May 2018, Universitat d'Alacant, Alacant, Spain.

[11] Lopez, A. (2008). Statistical machine translation. ACM Computing Surveys (CSUR), 40(3), 1-49.

[12] Bilbao, V. D., Lopes, J. P., & Ildefonso, T. (2005, December). Measuring the impact of cognates in parallel text alignment. In 2005 portuguese conference on artificial intelligence (pp. 338-343). IEEE.

[13] Sharma, D., Dhiman, C., & Kumar, D. (2023). Evolution of visual data captioning Methods, Datasets, and evaluation Metrics: A comprehensive survey. Expert Systems with Applications, 119773.

[14] Nakanishi, A. (1990). Writing Systems of the World. Tuttle Publishing.

[15] Sanders, G. A., Weiss, B. A., Schlenoff, C., Steves, M. P., & Condon, S. (2013). Evaluation methodology and metrics employed to assess the TRANSTAC two-way, speech-to-speech translation systems. Computer Speech & Language, 27(2), 528-553.

[16] Takakura, S., Han, D., & Furugori, T. (2005). Recognition and utilization of clausal relations in complex sentences for improving the performance of machine translation systems. Journal of Quantitative Linguistics, 12(2-3), 239-261.

[17] Dakwale, P. (2020). Strategies for effective utilization of training data for machine translation. Universiteit van Amsterdam.

[18] Kos, K. (2008). *Adaptation of new machine translation metrics for Czech* (Doctoral dissertation, Bachelor's thesis, Charles University in Prague)