**An Industry Oriented Project Report on**

# “WORD’S SPELL CHECK USING NLP TECHNIQUE”

Submitted to the

**JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY HYDERABAD**

In partial fulfilment of the requirement for the award of the degree of

**BACHELOR OF TECHNOLOGY**

**IN**

**COMPUTER SCIENCE AND ENGINEERING**

**BY**

**R. POOJITHA (19WJ5A0527)**

# Under the Esteemed Guidance Of

# Mr. P. HARI SHANKAR, Assist. Professor, CSE Dept.



**GURU NANAK INSTITUTIONS TECHNICAL CAMPUS**

**School of Engineering and Technology**

**Ibrahimpatnam, R.R District- 501506**

**2021-2022**



**Department of Computer Science and Engineering**

# CERTIFICATE

This is to certify that this project report entitled “**WORD’S SPELL CHECK USING NLP TECHNIQUE**” by **RAM POOJITHA (19WJ5A0527)**

Submitted in partial fulfilment of the requirements for the degree of **Bachelor of Technology** in **Computer Science and Engineering** of the **Jawaharlal Nehru Technological University Hyderabad** during the academic year 20212022, is a bona fide record of work carried out under our guidance and supervision.

**INTERNAL GUIDE PROJECT CO-ORDINATOR HOD CSE**

**Mr. P. HARI SHANKAR Mrs. V. SWATHI Mr. V. DEVASEKHAR**

**EXTERNAL EXAMINAR**



1

9

WJ

5A0527

Computer science and

Engineering

R.POOJITHA

P

Y

THON

“WORD SPELL CHECK USING NLP

”

RA

M

INNOVATIVE

INFOTECH

## ACKNOWLEDGEMENT

We wish to express our sincere thanks to **Dr. RishiSayal, Associate Director, GNITC** for providing us with the conducive environment for carrying through our academic schedules and Project with ease.

We would like to say our sincere thanks to **Mr. V. Deva Sekhar,** **Associate** **Professor & Head, Department of CSE**, **GNITC** for providing seamless support and right suggestions are given in the development of the project.

We would like to say our sincere thanks to **Mrs. V. Swathi**, **Asst. Professor, Department of CSE,** Major Project Coordinator, for providing seamless support and right suggestions are given in the development of the project.

We have been truly blessed to have a wonderful internal guide **Mr. P. Hari Shankar, Asst. Professor of CSE, GNITC** for guiding us to explore the ramification of our work and we express our sincere gratitude towards him for leading me through the completion of Project.

Finally, we would like to thank our family members for their moral support and encouragement to achieve goals.

**R.POOJITHA (19WJ1A0527)**

**NLP TECHNIQUE USING WORD’S SPELL CHECK USING PYTHON**

## ABSTRACT

Spelling correction is the process of finding the correct word for a misspelled word in a text. This system aimed to fix this error we cannot know the writer’s intent. But at the same time, it should find the word that the user wanted to write. In this study, we trained a Long -short term memory model with dictionary words and used as an oracle. For a misspelled word, this oracle returns a candidate dictionary word. Character level bigram model is used to generate new query words from a misspelled word. These new query words are also given to the trained network for getting more candidate dictionary words. For testing the method’s performance, randomly distorted dictionary words are used. Results showed that the trained network had an acceptable accuracy level. Also finding candidates using generated new query words have a positive impact on accuracy rather than using only misspelled word.

# TABLE OF CONTENTS

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO.** | **TITLE** | **PAGE NO.** |
|  | **ABSTRACT**  **LIST OF FIGURES**  **LIST OF SYMBOLS** | VI  X  XI |
| 1. | **INTRODUCTION**  1.1 GENERAL  1.2 OBJECTIVE  1.3 EXISTING SYSTEM  1.3.1 LITERATURE SURVEY  1.4 PROPOSED SYSTEM | 1  1  2  3  7 |
| 2. | **PROJECT DESCRIPTION**  2.1 GENERAL  2.2 METHODOLOGIES  2.2.1 MODULES NAME  2.2.2 MODULES EXPLANATION  2.3 TECHNIQUE OR ALGORITHM | 8  8  8  9  15 |
| 3. | **REQUIREMENTS**  3.1 GENERAL  3.2 HARDWARE REQUIREMENTS  3.3 SOFTWARE REQUIREMENTS  3.4 FUNCTIONAL REQUIREMENTS  3.5 NON-FUNCTIONAL REQUIREMENTS | 17  17  18  19  19 |
| 4. | **SYSTEM DESIGN**  4.1 GENERAL  4.2 UML  4.2.1 USECASE DIAGRAM  4.2.2 CLASS DIAGRAM  4.2.3 OBJECT DIAGRAM  4.2.4 COMPONENT DIAGRAM  4.2.5 DEPLOYMENT DIAGRAM  4.2.6 SEQUENCE DIAGRAM  4.2.7 COLLABORATION DIAGRAM  4.2.8 STATE DIAGRAM  4.2.9 ACTIVITY DIAGRAM  4.2.10 DATAFLOW DIAGRAM  4.3 SYSTEM ARCHITECTURE | 20    21  22  23  24  25  26  27  28  29  30  32 |
| 5. | **DEVELOPMENT TOOLS**  5.1 PYTHON  5.2 HISTORY OF PYTHON  5.2.3 IMPORTANCE OF PYTHON  5.2.4 FEATURES OF PYTHON  5.2.5 LIBRARIES USED IN PYTHON | 34  34  34  35  36 |
| 6. | **IMPLEMENTATION**  6.1 GENERAL | 37 |

|  |  |  |
| --- | --- | --- |
| 7. | **SNAPSHOTS**  7.1 GENERAL  7.2 VARIOUS SNAPSHOTS | 39  39 |
| 8. | **SOFTWARE TESTING**  8.1 GENERAL  8.2 DEVELOPING METHODOLOGIES  8.3 TYPES OF TESTING  8.3.1. UNIT TESTING  8.3.2 FUNCTIONAL TESTING  8.3.3 SYSTEM TESTING  8.3.4 PERFORMANCE TESTING  8.3.5 INTEGRATION TESTING  8.3.6 ACCEPTANCE TESTING | 41  41    41  42  42  42  42  43 |
| 9. | **FUTURE ENHANCEMENT**  9.1 FUTURE ENHANCEMENTS | 44 |
| 10. | **CONCLUSION**  10.1 CONCLUSION  10.2 REFERENCES | 45  46 |

## LIST OF FIGURES

|  |  |  |
| --- | --- | --- |
| **FIGURE NO** | **NAME OF THE FIGURE** | **PAGE NO.** |
| 4.2.1 | Use case Diagram | 21 |
| 4.2.2 | Class Diagram | 22 |
| 4.2.3 | Object Diagram | 23 |
| 4.2.4 | Component Diagram | 24 |
| 4.2.5 | Deployment Diagram | 25 |
| 4.2.6 | Sequence Diagram | 26 |
| 4.2.7 | Collaboration Diagram | 27 |
| 4.2.8 | State Diagram | 28 |
| 4.2.9 | Activity Diagram | 29 |
| 4.2.10 | Data Flow Diagram | 30 |
| 4.3 | System Architecture | 32 |

|  |  |  |  |
| --- | --- | --- | --- |
| **S.NO** | **NAME** | **NOTATION** | **DESCRIPTION** |
| 1. | Class | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | |  | | --- | | *+ public*  *-private* | | |  | | --- | | *Class Name* | | *-attribute* | | *-attribute* | | | Represents a collection of similar entities grouped together. |
| 2. | Association | |  |  |  | | --- | --- | --- | | Class A | NAME | Class B |      |  |  |  | | --- | --- | --- | | Class A |  | Class B | |  | | Associations represents static relationships between classes. Roles represents the way the two classes see each other. |
| 3. | Aggregation | |  |  |  | | --- | --- | --- | | Class A |  | Class A |      |  |  |  | | --- | --- | --- | | Class B |  |  | | Interaction between the system and external environment |
| 4. | Relation  (extends) | extend | Extends relationship is used when one use case is similar to another use case but does a bit more. |

## LIST OF SYMBOLS

|  |  |  |  |
| --- | --- | --- | --- |
| 5. | Actor |  | It aggregates several classes into a single class. |
| 6. | Communication |  | Communication between various use cases. |
| 7. | State | State | State of the process. |
| 8. | Initial State |  | Initial state of the object |
| 9. | Final state |  | Final state of the object |
| 10. | Control flow |  | Represents various control flow between the states |
| 11. | Decision box |  | Represents decision making process from a constraint |

|  |  |  |  |
| --- | --- | --- | --- |
| 12. | Use case | Uses case | Interact ion between the system and external  environment |
| 13. | Component |  | Represents physical modules which is a collection of components. |
| 14. | Node |  | Represents physical modules which are a collection of components. |
| 15. | Data  Process/State |  | A circle in DFD represents a state or process which has been triggered due to some event or action. |
| 16. | External entity |  | Represents external entities such as keyboard, sensors, etc. |
| 17 | Relation | uses | Used for additional process communication. |

1

8

.

(

uses

)

Transition

communication

Represents

between

that

occurs

processes.

1

9

.

Object Lifeline

Represents

the

vertical

dimensions that the object

communications.

20

.

Message

Message

Represents

the

message

exchanged.

# CHAPTER 1

# INTRODUCTION

## 1.1 GENERAL

Spelling errors occur in many kinds of texts regardless of their formality (whether they are formal texts or informal texts). Spelling errors are very common in human generated texts and the cause of them can vary from accidental hand or mind slips to writer’s lack of spelling knowledge. For reading purpose, spell detecting and correcting might be trivial for humans since humans can naturally detect patterns easily that make them able to read and understand texts even though the texts have spelling errors, while machines cannot do it without being instructed or trained to. In Natural Language Processing (NLP) applications, data normalization is very important because it can improve the performance and the accuracy of NLP applications. By performing spell checking before other normalization task for various NLP applications such as information retrieval, machine translation, text classification, and opinion mining, spell checking can reduce out-ofvocabulary (OOV), reduce the size of bag of words representation, and produce better stemming or lemmatization result. For humans, spell checking can help when they are writing texts that must contain no mistakes (often texts in formal context) or for a better readability of texts.

## 1.2 OBJECTIVE

The proposed system for the natural language allows the users to select a password in the Tamil language which can be easy to remember for the native speakers. This system is mostly concerned on “who is speaking?” rather than “what they are speaking?” and hence it acts as a speech-based authentication system.

# 1.2 EXISTING SYSTEM

Computers and similar digital devices are used daily for producing texts for various purposes. These texts may contain different types of errors like semantic, grammar or spelling related. Since user-generated texts generally contain misspellings, correcting spelling errors is very important especially for practical purposes. That’s why websites generally provide spelling correction as suggestions or automatic corrections. Correcting obvious errors by auto-correction is acceptable. On the other hand, suggestions are more convenient for users to accept a proposed correction without retyping. Spelling correction is a hard task especially without context information. Such as queries with a few words entered into a search engine in order to find products. Also, context information is not useful for all cases like general purpose words (e.g., the, this, what). In this paper, we focused on developing a spelling correction method which doesn’t require context information.

## 1.3 LITERATURE SURVEY

**Title:** Design and Implementation of NLP-Based Spell Checker for the Tamil Language

**Author:** Pawan Kumar, Abhishek Kannan and Nikita Goel

**Year:** 2020

**Description:**

A spell checker is a tool used for analyzing and validating spelling mistakes in the text. Recently, the role of a spell checker has diversified, and it is also used to suggest possible corrections to the detected spelling mistakes. Tamil is one of the oldest surviving and international spoken languages of the world, and it is grammatically very rich. Grammar is vital for effective communication and information transmission. However, learning the language rules and the old teaching methodology becomes a challenge for the researchers. The amalgamation of computer and language using natural language processing (NLP) provides a solution to this problem. In this paper, an advanced NLP technique is used to detect wrongly spelled words in the Tamil language text, and to provide possible correct word suggestions and the probability of occurrence of each word in the corpus.

**Title:** A study of spell-checking techniques for Indian Languages

**Author:** Rakesh Kumar, Minu Bala, Kumar Sourabh

**Year:** 2018

**Description:**

There are many commercial as well as non-commercial spelling error detection and correction tools available in the market for almost all popular languages. And every tool works on word level with the help of integral dictionary/Wordnet as the backend database for correction and detection. Every word from the text is looked up in the speller lexicon. When a word is not in the dictionary, it is detected as an error. In order to correct the error, a spell checker searches the dictionary/Wordnet for the word that is most resembled to the erroneous word. These words are then suggested to the user to choose the intended word. Spelling checking in used in various applications like machine translation, search, information retrieval etc. Spell checking technique comprises of two stages mainly error detection and error correction. In this paper we have studied various issues related spell checking techniques available so far as well as developmental approaches for error detection and correction for Indian languages.

**Title:** Swedish Natural Language Processing with Long Short-term Memory Neural Network

**Author:** Johan Gudmundsson Francis Menkes

**Year:** 2018

**Description:**

Natural Language Processing (NLP) is a field studying computer processing of human language. Recently, neural network language models, a subset of machine learning, have been used to great effect in this field. However, research remains focused on the English language, with few implementations in other languages of the world. This work focuses on how NLP techniques can be used for the task of grammar and spelling correction in the Swedish language, in order to investigate how language models can be applied to non-English languages. We use a controlled experiment to find the hyperparameters most suitable for grammar and spelling correction on the Göteborgs-Posten corpus, using a Long Short-term Memory Recurrent Neural Network. We present promising results for Swedish-specific grammar correction tasks using this kind of neural network; specifically, our network has a high accuracy in completing these tasks, though the accuracy achieved for language-independent typos remains low.

**Title:**  N-Gram based Automatic Spelling Correction Tool to Improve Retrieval Effectiveness

**Author:** Farag Ahmed, Ernesto William De Luca, and Andreas Nuernberger

**Title: 2009**

**Description:**

We present a language-independent spell-checker that is based on an enhancement of the n-gram model. The spell checker is proposing correction suggestions by selecting the most promising candidates from a ranked list of correction candidates that is derived based on n-gram statistics and lexical resources. Besides motivating and describing the developed techniques, we briefly discuss the use of the proposed approach in an application for keyword- and semanticbased search support. In addition, the proposed tool was compared with state-of-the-art spelling correction approaches. The evaluation showed that it outperforms the other methods.

## 1.4 PROPOSED SYSTEM

The proposed method is composed of two parts; a trained RNN (LSTM) and a character level bigram model. LSTM is used to query with a misspelled word in order to find a candidate word. With bigram, new query words are generated from the misspelled word, so that trained network can be queried with these generated words. Using these new query words, we expect to increase the correction accuracy. For LSTM training, words are encoded similar to RNN.

A given word’s first letter, end letter and in between letters are encoded to 3 input vectors and compose a sequence. Different than RNN, previous two words are not used as time steps. Instead, a sequence of 3 vectors that are computed from a word is used. So, any word correction is only based on its character sequence. A simple representation of our LSTM model. A given misspelled (nondictionary) word is used to query the trained LSTM network in order to find the desired word. Long Short-term Memory (LSTM) has shown to give an extraordinary result in solving sequential problems, including spelling correction. In this paper, we propose an LSTM model that encodes input word at character level, that also uses word and POS tag contexts as features.

# CHAPTER 2 PROJECT DESCRIPTION

**2.1 GENERAL:**

Speech Recognition system for natural language can be achieved using two important phases such as Feature Excerption phase and Feature Matching phase. Feature excerption is the process of deriving the required data while discarding the noise and other disturbances in the sample. In the feature matching phase, the MFCC features of voice signal are compared with the reference templates in the database that contains the user details with the help of Dynamic Time warping

## 2.2 METHODOLOGIES

**2.2.1 MODULES:**

1. Error detection techniques
2. Error correction techniques

## 

## ERROR DETECTION TECHNIQUES

A string of character in separated by space bar or punctuation marks may be called a candidate word. A candidate word is a valid if it has a meaning else it is a non-word. There are two techniques for error detection is N-gram analysis and dictionary lookup. The error detection process usually consists of checking to see if an input string is a valid index or dictionary word. Efficient techniques have been devised for detecting such types of errors. The two most known techniques are n-gram analysis and dictionary lookup. Spellcheckers rely mostly on dictionary lookup and text recognition systems rely on n-gram techniques.

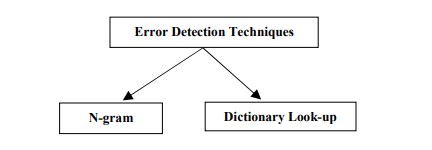


Fig2.2.1: Error Detection Techniques

**N-GRAM ANALYSIS**

N-gram analysis is described as a method to find incorrectly spelled words in a mass of text. Instead of comparing each entire word in a text to a dictionary, just n-grams are controlled. A check is done by using an n-dimensional matrix where real ngram frequencies are stored. If a non-existent or rare n-gram is found the word is flagged as a misspelling, otherwise not. An ngram is a set of consecutive characters taken from a string with a length of whatever n is set to. If n is set to one then the term used is a unigram, if n is two then the term is a Bigram, if n is three then the term is trigram. The n-gram algorithm was developed as one of the benefits is that it allows strings that have differing prefixes to match and the algorithm is also tolerant of misspellings. Each string that is involved in the comparison process is split up into sets of adjacent n-grams. The ngrams algorithms have the major advantage that they require no knowledge of the language that it is used with and so it is often called language independent or a neutral string-matching algorithm. Using n-grams to calculate for example the similarity between two strings is achieved by discovering the number of unique n-grams that they share and then calculating a similarity coefficient, which is the number of the n-grams in common (intersection), divided by the total number of n-grams in the two words (union)

**DICTIONARY LOOKUP**

A dictionary is a list of words that are assumed to be correct. Dictionaries are represented in many ways, each with their own characteristics like speed and storage requirements. Large dictionary might be a dictionary with most common word combined with a set of additional dictionaries for specific topics such as computer science or economy. Big dictionary also uses more space and may take longer time to search. The non-word errors can be detected as mentioned above by checking each word against a dictionary. The drawbacks of this method are difficulties in keeping such a dictionary up to date, and sufficiently extensive to cover all the words in a text. At the same time, one should keep down system response time. Dictionary lookup and construction techniques must be tailored according to the purpose of the dictionary. Too small a dictionary can give the user too many false rejections of valid words, too large it can accept a high number of valid low frequency words. Hash tables are the most common used technique to gain fast access to a dictionary. In order to lookup a string, one has to compute its hash address and retrieve the word stored at that address in the pre constructed hash table. If the word stored at the hash address is different from the Input string, a misspelling is flagged. Hash tables main advantage is their random-access nature that eliminated the large number of comparisons needed to search the dictionary. The main disadvantage is the need to devise a clever hash function that avoids collisions. To store a word in the dictionary we calculate each hash function for the word and set the vector entries corresponding to the calculated values to true. To find out if a word belongs to the dictionary, you calculate the hash values for that word and look in the vector. If all entries corresponding to the values are true, then the word belongs to the dictionary, otherwise it does not

**2. ERROR CORRECTION TECHNIQUES**

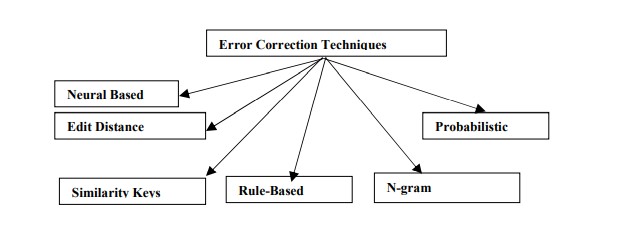
Correction of spelling errors is an old problem. Much research has been done in this area over the years. Most existing spelling correction techniques focus on isolated words, without taking into account the textual context in which the string appears. Error correction consists of two steps: the generation of candidate corrections and the ranking of candidate corrections. The candidate generation process usually makes use of a precompiled table of legal n-grams to locate one or more potential correction terms. The ranking process usually invokes some lexical similarity measure between the misspelled string and the candidates or a probabilistic estimate of the likelihood of the correction to rank order the candidates. These two steps are most of the time treated as a separate process and executed in sequence. Some techniques can omit the second process though, leaving the ranking and final selection to the user. The isolated-word methods that will be described here are the most studied spelling correction algorithms, they are: edit distance, similarity keys, rule-based techniques, n-gram-based techniques, probabilistic techniques and neural networks. All of these methods can be thought of as calculating a distance between the misspelled word and each word in the dictionary or index. The shorter the distance the higher the dictionary word is ranked.

Fig2.2.

1:

Error

Correction Techniques



**NEURAL BASED:**

Neural networks are also an interesting and promising technique, but it seems like it has to mature a bit more before it can be used generally. The current methods are based on backpropagation networks, using one output node for each word in the dictionary and an input node for every possible n-gram in every position of the word, where n usually is one or two. Normally only one of the outputs should be active, indicating which dictionary words the network suggests as a correction. This method works for small (< 1000 words) dictionaries, but it does not scale well. The time requirements are too big on traditional hardware, especially in the learning phase.

**EDIT DISTANCE:**

Edit distance is a simple technique. Simplest method is based on the assumption that the person usually makes few errors if ones, therefore for each dictionary word. The minimal number of the basic editing operations (insertion, deletions, substitutions) necessary to covert a dictionary word in to the non-word. The lower, the number ,the higher the probability that the user has made such errors. Edit distance is useful for correcting errors resulting from keyboard input, since these are often of the same kind as the allowed edit operations. It is not quite as good for correcting phonetic spelling errors, especially if the difference between spelling and pronunciation is big as in English or French.

**SIMILARITY KEYS:**

An index/key is assigned to each dictionary word for comparing with the key computed for the non-word. The word for which the keys computed for the non-word. The word for which the keys are most similar are selected as suggestion. Such an approach is speed effective as only the words with similar keys have to be processed with a good transformation algorithm this method can handle keyboard errors.

**RULE-BASED TECHNIQUES:**

Rule-based methods are interesting. They work by having a set of rules that capture common spelling and typographic errors and applying these rules to the misspelled word.

Intuitively these rules are the “inverses” of common errors. Each correct word generated by this process is taken as a correction suggestion. The rules also have probabilities, making it possible to rank the suggestions by accumulating the probabilities for the applied rules. Edit distance can be viewed as a special case of a rule-based method with limitation on the possible rules.

**N-GRAM-BASED TECHNIQUES:**

N-grams can be used in two ways, either without a dictionary or together with a dictionary. Used without a dictionary, n-grams are employed to find in which position in the misspelled word the error occurs. If there is a unique way to change the misspelled word so that it contains only valid ngrams, this is taken as the correction. The performance of this method is limited. Its main virtue is that it is simple and does not require any dictionary. Together with a dictionary, n-grams are used to define the distance between words, but the words are always checked against the dictionary. This can b done in several ways, for example check how many n-grams the misspelled word and a dictionary word have in common, weighted by the length of the words.

**PROBABILISTIC TECHNIQUES:**

They are, simply put, based on some statistical features of the language. Two common methods are transition probabilities and confusion probabilities. Transition probabilities are similar to n-grams. They give us the probability that a given letter or sequence of letters is followed by an16 another given letter. Transition probabilities are not very useful when we have access to a dictionary or index. Given a sentence to be corrected, the system decomposes Each string in the sentence into letter n-grams and retrieves word candidates from the lexicon by comparing string n-grams with lexicon entry n-grams. The retrieved candidates are ranked by the conditional probability of matches with the string, given character confusion probabilities. Finally, a wordbigram model and a certain algorithm are used to determine the best scoring word sequence for the sentence. They claim that the system can correct non-word errors as well as real word errors and achieves a 60.2 % error reduction rate for real OCR text.

## 2.3 TECHNIQUE USED OR ALGORITHM USED

**2.3.1 EXISTING TECHNIQUE: -**

## ➢ RNN

Recurrent neural networks (RNN) are feed forward neural networks to sequences processing. RNN computes hidden states at each time step as a sequence [h1, h2, ..., hk] for a given sequence [x1, x2, ..., xk] where xi  Rn. At time-step activation of the hidden state is computed as a function f of the current input xt and previous hidden state an output can be computed in every time-step. An output sequence can be generated y1, y2, ..., yk for sequence-to-sequence tasks.

**2.3.2 PROPOSED TECHNIQUE: -**

➢ **LSTM**

Long Short-term Memory (LSTM) has shown to give an extraordinary result in solving sequential problems, including spelling correction. In this paper, we propose an LSTM model that encodes input word at character level, that also uses word and POS tag contexts as features. We performed the experiment on an artificial dataset based on Indonesian Wikipedia articles that we made by simulating some artificial spelling errors at character level and tested it on real dataset, mostly are Indonesian online news articles.

In this work, we use LSTM as an encoder and decoder on sequence-to-sequence model, which examines word by word at character level. One feed-forward pass of this model examines one word. For each time step, character at the time step is encoded to one-hot vector. Previous work and next word are also used as feature, those words are represented by embedding vector by summing their respective word vector and divide it by the number of context words that are used. Other feature that is used is POS tag of previous and next word that is also encoded to one-hot vector, previous work POS tag one-hot vector and next POS tag one-hot vector will be concatenated. For every time step, those vectors are concatenated to produce a feature vector, so every time step input in one forward pass has different character one-hot vector (unless they are the same character) but same context word-embedding vector and context POS tags vector. In encoder, the input that is used is feature vector that consists of character one-hot vector, word embedding vector and POS tags one-hot vector. In decoder however, the input is only the character one-hot vector, this also applies for the output, so each time step only produces character vector as output. Fig is the architecture of the model. xn is the feature vector that is the concatenation of character one-hot vector at time step, word embedding vector and POS tags one-hot vector n. yn is an output vector at time step in that only consists of character vector. n is the length of the longest word in the data train. So, before the training phase, we first have to define the maximum word length based on the data train. We then use this to transform every word that has length less than the maximum length to be padded to maximum length by replacing the rest of the length with an asterisk (this process is performed after adding the artificial spelling errors). For example, the longest word in the data train is word” berkewarganegaraanb” that has length of 19, suppose there is a word” gimansa” that has length of 7, this word is then transformed to” gimansa

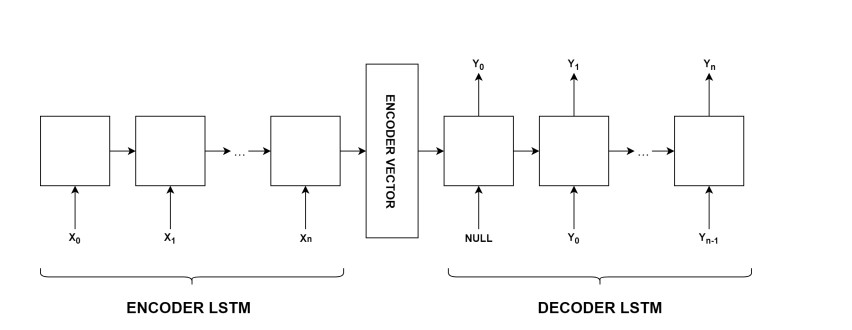


Fig: LSTM Architecture Model

**CHAPTER 3**

**REQUIREMENTS ENGINEERING**

## 3.1 GENERAL

The interpretation of the handwriting character by developing techniques and methods such as improvement of character classification techniques. The accurate and rapid classification for accurate information retrieval, sound classification, stock price forecasting.

## 3.2 HARDWARE REQUIREMENTS

The hardware requirements may serve as the basis for a contract for the implementation of the system and should therefore be a complete and consistent specification of the whole system. They are used by software engineers as the starting point for the system design. It should what the system does and not how it should be implemented.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| • | Processor |  | - | Intel corei5 |
| • | Speed |  | - | 2.60 GHz |
| • | Ram |  | - | 256 MB |
| • | Hard Disk |  | - | 20 GB |
| • | Key Board |  | - | Standard Windows Keyboard |
| • | Mouse |  | - | Two or Three Button Mouse |
| • | Monitor |  | - | SVGA |
|  |  |  |  |  |

## 3.3 SOFTWARE REQUIREMENTS

The software requirements document is the specification of the system. It should include both a definition and a specification of requirements. It is a set of what the system should do rather than how it should do it. The software requirements provide a basis for creating the software requirements specification. It is useful in estimating cost, planning team activities, performing tasks and tracking the teams and tracking the team’s progress throughout the development activity.

## SOFTWARE REQUIREMENTS

* Operating System - Windows 10
* Platform - Spyder
* Coding Language - Python

## 3.4 FUNCTIONAL REQUIREMENTS

A functional requirement defines a function of a software-system or its component. A function is described as a set of inputs, the behavior, Firstly, the system is the first that achieves the standard notion of semantic to the NLP process.

## 3.5 NON-FUNCTIONAL REQUIREMENTS

**EFFICIENCY**

Our multi-modal event tracking and evolution framework is suitable for multimedia documents from various social media platforms, which can not only effectively capture their multimodal topics, but also obtain the evolutionary trends of social events and generate effective event summary details over time. Our proposed LSTM model can exploit the multi-modal property of social event, which can effectively model social media documents including long text with related images and learn the correlations between textual and visual modalities to separate the visualrepresentative topics and non-visual-representative topics.

**CHAPTER 4**

# DESIGN ENGINEERING

## 4.1 GENERAL

Design Engineering deals with the various UML [Unified Modelling language] diagrams for the implementation of project. Design is a meaningful engineering representation of a thing that is to be built. Software design is a process through which the requirements are translated into representation of the software. Design is the place where quality is rendered in software engineering. Design is the means to accurately translate customer requirements into finished product.

## 4.2 UML DIAGRAMS

### 4.2.1 USE CASE DIAGRAM

Dictionary Lookup

n-Gram Analysis

Error Detection Techniques

Error Collection

Techniques

Checksum

Bunch Of Key

Probalistic Language Model

Marlov Model

Cyclic Rebundancy Check

Two Dimension Parity Check

Sample

NLP

Similarity Key

Technique

Dataset

STS

#### Fig4.2.1: Use case Diagram for NLP Technique using word’s spell check

**EXPLANATION:**

The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted. The above diagram consists of user as actor. Each will play a certain role to achieve the concept.

### 4.2.2 CLASS DIAGRAM



Fig4.2.2: Class Diagram for NLP Technique using word’s spell check

## EXPLANATION

In this class diagram represents how the classes with attributes and methods are linked together to perform the verification with security. From the above diagram shown the various classes involved in our project.

### 4.2.3 OBJECT DIAGRAM

Similarity Key Techniques

Error Detection Technique

Dictionary Lookup

n-Gram Analysis

Dataset

Error Collection Techniques

#### Fig4.2.3**:** Object Diagram for NLP Technique using word’s spell check

**EXPLANATION:**

In the above digram tells about the flow of objects between the classes. It is a diagram that shows a complete or partial view of the structure of a modeled system. In this object diagram represents how the classes with attributes and methods are linked together to perform the verification with security.

### 4.2.4 COMPONENT DIAGRAM

Error Detection

Techniques

Two Dimension

parity

Checksum

Cyclic Rebudancy

Check

PLM

n-Gram

Analysis

Launch Key

Dictionary

Lookup

Markov

Model

Error Collection

Techniques

Sampling

Similarity Key

Technique

Dataset

STS

NLP

Fig4.2.4:Component Diagram For NLP Technique using word’s spell check

## EXPLANATION

In the Unified Modeling Language, a component diagram depicts how components are wired together to form larger components and or software systems. They are used to illustrate the structure of arbitrarily complex systems. User gives main query and it converted into sub queries and sends through data dissemination to data aggregators. Results are to be showed to user by data aggregators. All boxes are components and arrow indicate dependencies.

### 4.2.5 DEPLOYMENT DIAGRAM

Similarity Key

Techniques

Error Detection

Techniques

Dictionary

Lookup

n-Gram

Analysis

Dataset

Error Collection

Techniques

#### Fig4.2.5: object Diagram for NLP Technique using word’s spell check

**EXPLANATION:**

Deployment Diagram is a type of diagram that specifies the physical hardware on which the software system will execute. It also determines how the software is deployed on the underlying hardware. It maps software pieces of a system to the device that are going to execute it.

### 4.2.6 SEQUENCE DIAGRAM

Error Detection

Techniques

Dictionary

Lookup

Dataset

Similarity Key

Technique

Error Collection

Techniques

n-Gram

Analysis

Two Dimension Parity

Checksum

Launch of Key

PML

Markov Model

Sampling

NLP

STS

#### Fig4.2.6: Sequence Diagram for NLP Technique using word’s spell check

**EXPLANATION:**

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario.

### 4.2.7 COLLABORATION DIAGRAM

Markov Model

7:

Error Detection

Techniques

Dictionary

Lookup

Dataset

n-Gram

Analysis

Error Collection

Techniques

Similarity Key

Technique

1:

Two Dimension Parity

NLP

2:

Launch of Key

3:

4:

Checksum

5:

PML

6:

Sam

pli

ng

ST

S

8:

#### Fig4.2.7: Collaboration Diagram for NLP Technique using word’s spell check

**EXPLANATION:**

A collaboration diagram, also called a communication diagram or interaction diagram, is an illustration of the relationships and interactions among software objects in the Unified Modeling Language (UML). The concept is more than a decade old although it has been refined as modeling paradigms have evolved.

### 4.2.8 STATE DIAGRAM

State1

Error Detection Technoques

Two Dimension Parity Check

Checksum

Dictionary Lookup

Launch of Key

n-Gram Analysis

PML

Marlkov Model

Error Collection Technioques

Sampling

Similarity Key Technique

STS

Dataset

#### Fig4.2.8: State Diagram for NLP Techinque using word’s spell check

**EXPLANATION:**

State diagram are a loosely defined diagram to show workflows of stepwise activities and actions, with support for choice, iteration and concurrency. State diagrams require that the system described is composed of a finite number of states; sometimes, this is indeed the case, while at other times this is a reasonable abstraction. Many forms of state diagrams exist, which differ slightly and have different semantics.

### 4.2.9 ACTIVITY DIAGRAM

Dataset

Error Detection Techniques

Two Dimension Parity

Checksum

Dictionary Lookup

Bunch key

n-Gram Analysis

PLM

Markov Model

Error Collection Techniques

Sampling

Similarity Key Technique

NLP

STS

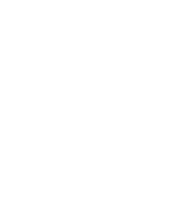
Dataset

#### Fig4.2.9: Activity Diagram for NLP Technique using word’s spell check

**EXPLANATION:**

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

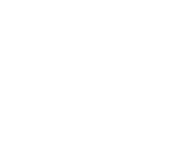
**4.2.10 DATA FLOW DIAGRAM LEVEL-0:**



Two

Dimension

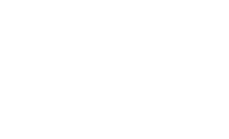
Parity



Classificati

on

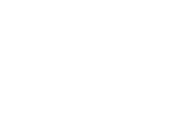
Technique



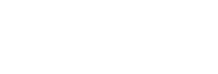
Checksum

Fig4.2.10:Data Flow Diagram for Level-0

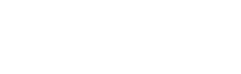
**LEVEL-1:**



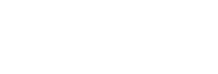
NLP



PLM



STS

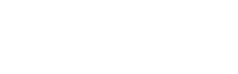


n

-

Graph

Analysis



Markov Model

Fig4.2.10:Data Flow Diagram for Level-1

**EXPLANATION**:

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modeling its process aspects. Often, they are a preliminary step used to create an overview of the system which can later be elaborated. DFDs can also be used for the visualization of data processing (structured design). A DFD shows what kinds of data will be input to and output from the system, where the data will come from and go to, and where the data will be stored.it does not show information about the timing of processes will operate in sequence or in parallel

**4.3**

**SYSTEM ARCHITECTURE**

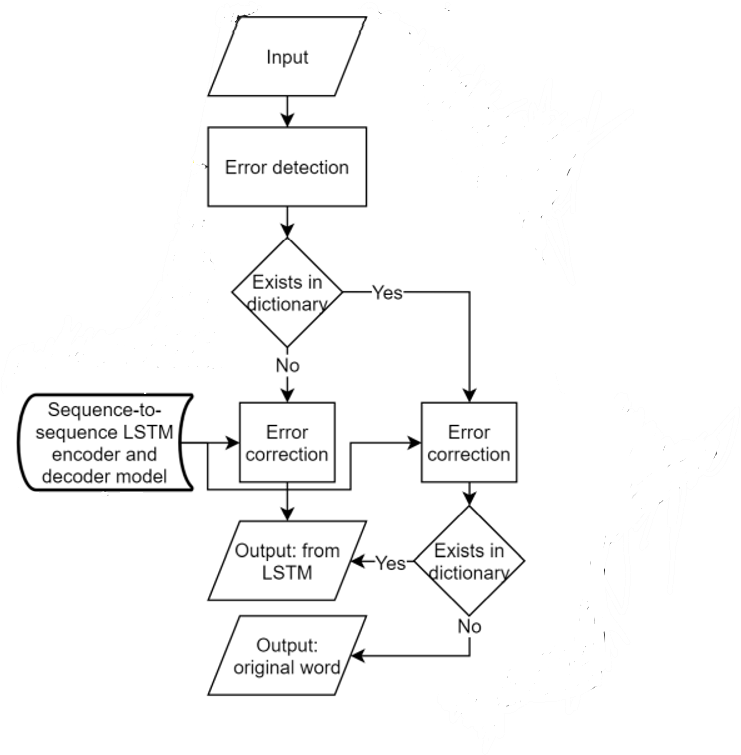


Fig: System Architecture of NLP Technique Using word’s spell check

**EXPLANATION:**

System Architecture is a diagram that specifies the physical hardware on which the software system will execute. It also determines how the software is deployed on the underlying hardware.

It maps software pieces of a system to the device that are going to execute it.

# CHAPTER 5

**DEVELOPMENT TOOLS**

## Python

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

## HISTORY OF PYTHON

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.

Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol68, Smalltalk, and Unix shell and other scripting languages.

Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).

Python is now maintained by a core development team at the institute, although Guido

van Rossum still holds a vital role in directing its progress.

### IMPORTANCE OF PYTHON

* **Python is Interpreted** − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
* **Python is Interactive** − You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
* **Python is Object-Oriented** − Python supports Object-Oriented style or technique of programming that encapsulates code within objects.

* **Python is a Beginner's Language** − Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

### FEATURES OF PYTHON

* **Easy-to-learn** − Python has few keywords, simple structure, and a clearly defined syntax.

This allows the student to pick up the language quickly.

* **Easy-to-read** − Python code is more clearly defined and visible to the eyes.
* **Easy-to-maintain** − Python's source code is fairly easy-to-maintain.
* **A broad standard library** − Python's bulk of the library is very portable and crossplatform compatible on UNIX, Windows, and Macintosh.
* **Interactive Mode** − Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
* **Portable** − Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
* **Extendable** − You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
* **Databases** − Python provides interfaces to all major commercial databases.
* **GUI Programming** − Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
* **Scalable** − Python provides a better structure and support for large programs than shell scripting.

Apart from the above-mentioned features, Python has a big list of good features, few are listed below −

* It supports functional and structured programming methods as well as OOP.
* It can be used as a scripting language or can be compiled to byte-code for building large applications.
* It provides very high-level dynamic data types and supports dynamic type checking.
* IT supports automatic garbage collection.
* It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

**Libraries used in python:**

* NumPy - Mainly useful for its N-dimensional array objects.
* Pandas - Python data analysis library, including structures such as data frames.
* Matplotlib - 2D plotting library producing publication quality figures.
* Scikit-learn - The machine learning algorithms used for data analysis and data mining tasks.

**CHAPTER 6**

# IMPLEMENTATION

**6.1 GENERAL CODING:**

from text blob import Text Blob #importing text blob library t = 1 while t:

a = input ("Enter the word to be checked: - ") # incorrect spelling print ("original text: "+str(a)) #printing original text b = Text Blob(a) #correcting the text # Prints the corrected spelling

print ("corrected text: "+str (b. correct ())) t = int (input ("Try Again? 1: 0 "))

# README.MD

# Script Title

<! --Remove the below lines and add yours -->

Here, you can input any word and check if it is having a correct spelling or not.

### Prerequisites

<! --Remove the below lines and add yours -->

First thing which you need to install is text blob library

<! --Install library-->

>pip install text blob

<! --For Jupiter nb-->

You need to run this command in your terminal or your ide terminal. <! --for jp nb-->

If you are using Jupiter Notebook you need to use the below command

<! --for jp nb-->

>import sys

<! --command-->

>! {sys. executable} -m pip install text blob

### How to run the script

<! --Remove the below lines and add yours -->

You can first install the textblob library and then you can run the python script.

**CHAPTER 7**

# SNAPSHOTS

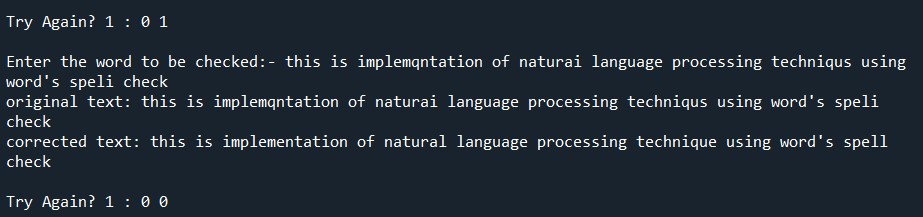
## 7.1 SNAPSHOT-1



EXPLANATION:

If we give one word as an input, if the word is present in the dictionary, it will give the correct text. If the text is correct, we will get the output as 0.

## 7.2 SNAPSHOT-2



EXPLANATION:

If we give one word as input, if the word is present in the dictionary .We will get the output as 0 or else we will get the output as 1. If we get the output as 1, we will continue the process until we get the correct output.

# 

# CHAPTER 8

# SOFTWARE TESTING

## 8.1 GENERAL

The purpose of testing is to discover errors. Testing is the process of trying o discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement.

## 8.2 DEVELOPING METHODOLOGIES

The test process is initiated by developing a comprehensive plan to test the general functionality and special features on a variety of platform combinations. Strict quality control procedures are used. The process verifies that the application meets the requirements specified in the system requirements document and is bug free. The following are the considerations used to develop the framework from developing the testing methodologies.

## 8.3 TYPES OF TESTS

### 8.3.1 UNIT TESTING

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program input produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

### 8.3.2 FUNCTIONAL TESTING

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input: identified classes of valid input must be accepted.

Invalid Input: identified classes of invalid input must be rejected.

Functions: identified functions must be exercised.

Output: identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

### 8.3.3 SYSTEM TESTING

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

### 8.3.4 PERFORMANCE TESTING

The Performance test ensures that the output be produced within the time limits, and the time taken by the system for compiling, giving response to the users and request being send to the system for to retrieve the results.

### 8.3.5 INTEGRATION TESTING

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications,

e.g., components in a software system or – one step up – software applications at the company level – interact without error.

### 8.3.6ACCEPTANCE TESTING

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

**ACCEPTANCE TESTING FOR DATA SYNCHRONIZATION:**

* The Acknowledgements will be received by the Sender Node after the Packets are received by the Destination Node
* The Route add operation is done only when there is a Route request in need
* The Status of Nodes information is done automatically in the Cache Updation process

### 8.2.7 BUILD THE TEST PLAN

Any project can be divided into units that can be further performed for detailed processing. Then a testing strategy for each of this unit is carried out. Unit testing helps to identity the possible bugs in the individual component, so the component that has bugs can be identified and can be rectified from errors.

# CHAPTER 9

# FUTURE ENHANCEMENT

## 9.1 FUTURE ENHANCEMENTS

In this Project, we tried to achieve a less information dependent and context-free spelling correction method. Spelling correction is auto-correction form list. A recurrent neural network model LSTM is used to predict the correct form of a given misspelled word. Using character-based bigram model new query words generated in order to increase correction accuracy. Test results are promising but testing with real-life data would give more accurate results about the method’s success.

# CHAPTER 10

# CONCLUSION

## 10.1 CONCLUSION

The spelling checker in this project is able to detect and correct spelling errors. Since the main purpose of the models is to correct incorrect words, our main evaluation metrics is the recall incorrect. Our Character gives the highest accuracy that is 83.76%. With the given results, we conclude that using LSTM with character as time step feature and additional non-time step features can detect and correct spelling errors in English language. By using more data for training our models, we believe that they will perform better. By using named entity recognition, it should also be able to recognize that named entities should not be evaluated, thus the models can give better accuracies.

## 10.2 REFERENCES

1. F. J. Damerau, “A technique for computer detection and correction of spelling errors,” Communications of the ACM, vol. 7, no. 3, pp. 171–176, 1964.
2. R. C. Angell, G. E. Freund, and P. Willett, “Automatic spelling correction using a trigram similarity measure,” Information Processing & Management, vol. 19, no. 4, pp. 255–261, 1983.
3. M. D. Kernighan, K. W. Church, and W. A. Gale, “A spelling correction program based on a noisy channel model,” in Proceedings of the 13th conference on Computational LinguisticsVolume 2. Association for Computational Linguistics, 1990, pp. 205–210.
4. H. Takahashi, N. Itoh, T. Amano, and A. Yamashita, “A spelling correction method and its application to an OCR system,” Pattern Recognition, vol. 23, no. 3-4, pp. 363–377, 1990.
5. K. W. Church and W. A. Gale, “Probability scoring for spelling correction,” Statistics and Computing, vol. 1, no. 2, pp. 93–103, 1991.
6. E. Mays, F. J. Damerau, and R. L. Mercer, “Context based spelling correction,” Information Processing & Management, vol. 27, no. 5, pp. 517–522, 1991.
7. K. Oflazer and C. Guzey, “Spelling correction in agglutinative languages,” in Proceedings of the fourth conference on Applied natural language processing. Association for Computational Linguistics, 1994, pp. 194–195.
8. A. R. Golding, “A Bayesian hybrid method for context-sensitive spelling correction,” arXiv preprint cmp-lg/9606001, 1996.
9. K. Oflazer, “Error-tolerant finite-state recognition with applications to morphological analysis and spelling correction,” Computational Linguistics, vol. 22, no. 1, pp. 73–89, 1996.
10. E. Brill and R. C. Moore, “An improved error model for noisy channel spelling correction,” in Proceedings of the 38th Annual Meeting on Association for Computational Linguistics.

Association for Computational Linguistics, 2000, pp. 286–293.

1. K. Toutanova and R. C. Moore, “Pronunciation modeling for improved spelling correction,” in Proceedings of the 40th Annual Meeting on Association for Computational Linguistics.

Association for Computational Linguistics, 2002, pp. 144–151.

1. S. Cucerzan and E. Brill, “Spelling correction as an iterative process that exploits the collective knowledge of web users,” in Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, 2004.
2. G. Chrupała, “Text segmentation with character-level text embeddings,” arXiv preprint arXiv:1309.4628, 2013.