Research Article

Aibi research, management and engineering journal. Volume 13, Number 2, Pag 1-11, May - August 2025, ISSN 2346-030X

Gujarati (ગુજરાતી) spell checker using Norvig algorithm with grammatical rules of Rhasva, Dīrgha and Anusvāra

Brijeshkumar Y. Panchal[1], Apurva Shah[2]
[1,2]*The Maharaja Sayajirao University of Baroda, Gujarat - India*
[1]*Gujarat Technological University, Gujarat - India*

Received: november 27, 2024.
Accepted: april 04, 2025.
Publicado: may 01, 2025.

***Abstract*—** This paper focused on Gujarati Spell Checker Using Norvig Algorithm with Grammatical Rules of Rhasva, Dīrgha, Anusvāra. Researchers make a note of Gujarati language history with a brief explanation of consonants and vowels. A total of 5, 5 and 14 rules are available respectively for Anusvāra, Rhasva, and Dīrgha. Many Indian languages use the term Anusvara to denote a nasal sound with a specific symbol. A single mātrā Small E-i (હ્રસ્વ ઇ) / U-u (હ્રસ્વ ઉ – ુ) is called Rhasva in gujarati literature while double mātrā Long E-e (દીર્ઘ ઈ – ી) / Long U-oo (દીર્ઘ ઊ- ૂ) known as Dīrgha respectively. This manuscript constitutes an innovative effort in the domain of Gujarati Natural Language Processing (GNLP) investigation, presenting a comprehensive repository of grammatical rules for the Gujarati language, inclusive of examples in both English and Gujarati. The proposed system will use given grammatical rules. Researcher uses two version of dataset for Gujarati Spell Checker with Norvig Algorithm in replace and insert function. One has whole barakhdi and second one focuses on Rhasva, Dīrgha, Matras, and Anusvāra in algorithm process. Output of this paper may explain that the deep learning approach is more appropriate compare to other approach.

Keywords: rule-based, gujarati, machine learning (ml), grammatical rule rhasva, dīrgha, anusvāra, natural language processing (nlp), gnlp, artificial intelligence (ia).

*Corresponding author: panchalbrijesh02@gmail.com (Brijeshkumar Y. Panchal).
Peer reviewing is a responsability of the Universidad de Santander.
How to cite this article: B. Y. PanchalandA.Shah, “Gujarati (ગુજરાતી) spell checker using Norvig algorithm with grammatical rules of Rhasva, Dīrgha and Anusvāra”, *Aibi research, management and engineering journal*, vol. 13, no. 2, pp. 1-11 2025, doi: 10.15649/2346030X.4446

I. INTRODUCTION

In this paper, researchers focus more on historical aspects of Gujarati languages and the technical bilingual rules of Gujarati grammar, as well as propose a rule-based system.

a. Gujarati language History

Gujarati, being the language associated with Gandhi, has a noteworthy significance that warrants our attention. The Gujarati language extends beyond the boundaries of Gujarat State, serving as the primary language of traders throughout India and even beyond to East Africa. The Gujarati language incorporates borrowings from Persian, Arabic, Portuguese, and English.

A Gujarati language, similar to Marathi, Hindi, Punjabi, Oriya, and several other Indian languages, is classified within the Aryan linguistic group and is considered a descendant of Sanskrit. The language in question exhibits its closest linguistic connections with Western Panjabi on the one hand and Braj Bhasha, an archaic variant of Hindi, on the contrary. In addition to the regional dialects spoken in Gujarat, there are three primary forms of the written and spoken language.

The Hindi language, namely the Gujarati dialect, has been officially accepted by the Government as the standard and hence is studied in schools. Furthermore, Parsi Gujarati refers to the language used by the Parsi community in both spoken and written forms. This variant of Gujarat deviates from the standard form by including a significant amount of original Persian terms, particularly in relation to religious subjects. Additionally, it includes several Arabic and Urdu loanwords. Furthermore, its grammar exhibits a state of instability and irregularity. Furthermore, the region of Muhammadan Gujarat, similar to Parsi Gujarat, extensively incorporates a significant quantity of lexicon derived from Hindustani, which in turn has its origins in Persian and Arabic. However, although the lexicon of the language exhibits significant variation depending on whether the speaker identifies as a Hindu, a Parsi, or a Muslim, the syntax remains mostly consistent when spoken properly. The user's composition lacks an academic tone and structured organization. It requires a transformation to embody a more refined style. By acquiring proficiency in this dialect, students will effortlessly grasp the subtle variations in form that emerge in different dialects, especially since these distinctions primarily relate to orthography [1].

Throughout history, three additional prominent dialects have been recognized: Surati, named after the vibrant city of Surat in the southern realm of Gujarat; Kathiawari, predominantly heard in the Saurashtra region (Kathiawar) that lies to the southwest of Ahmedabad; and a northern dialect, occasionally called Pattani, taking its name from the historic city of Patan, found in the distant northern reaches of Ahmedabad. The aforementioned descriptions of dialects lack clarity and fail to encapsulate a thorough exploration of Gujarati dialects, which remains elusive at present [1].

b. Gujarati Alphabet

The enchanting Gujarati alphabet finds its roots in the ancient Sanskrit, echoing its elegance with remarkable fidelity. The fundamental difference arises from the absence of the characteristic headline that typically links the majority of characters in the Devanagari script, imparting Gujarati with its distinctive allure. Mirroring its Sanskrit sibling, the Gujarati language dances elegantly from left to right. Within the captivating realm of linguistics, it is customary to classify letters, referred to as varn, into two main categories: vowels, known as svar, and consonants, which are pronounced as vyanjan or venjan [2].

The Gujarati language boasts a unique script of its own. This script can be identified as a syllabic alphabet, specifically a captivating abugida, whose roots trace back to the ancient Brāhmī script. In this intriguing writing system, every consonant naturally carries the vowel [ə]. The principles underlying this script bear striking similarities to those of the Devanāgarī script. Diacritic marks for vowels are deftly employed to signify non-initial vowels, which may elegantly appear before, after, above, or below a consonant. The Gujarati script constitutes a harmonious assembly of 47 characters, skillfully designed upon the principles of phonetic insight. Each character is embellished with its conventional transliteration paired with its counterpart in the International Phonetic Alphabet. The arrangement commences with the essential vowels, fluidly progresses into the syllabic vowels, and finally culminates in the diphthongs (most notably, the vowels e and o, which trace their lineage back to ancient diphthongs and were acknowledged as such by the indigenous grammarians). Following the vowels, the stops and nasal consonants are meticulously grouped into five captivating clusters, each housing five letters, classified by their place of articulation, transitioning from the back of the throat to the forefront. For each collection, the order is as follows: voiceless unaspirated stop, voiceless aspirated stop, voiced unaspirated stop, voiced aspirated stop, and nasal. After these five classifications, the semivowels (liquids and glides) are further organized based on their points of articulation. Next, we shall explore the fricatives that begin with the sibilant tones. The final two clusters consist of biconsonantal pairs, which are solely present in Sanskrit loanwords and have historically been woven into the fabric of the alphabet [3-5].

- In the transcription system used, the vowel sounds [e] and [ɛ] are both denoted by the letter "e." The vowel sound [ə] is represented by the letter "a," and the sound [a] is denoted by "ā." Similarly, the vowel sounds [o] and [ɔ] are represented by the letter "o." Lastly, the long vowels [i] and [u] are indicated by the letters "ī" and "ū," respectively.
- The syllabic vowel "ri," transliterated as "ṛ," appears exclusively in Sanskrit loanwords.
- Aspirated stops and affricates are indicated using digraphs, such as "ph" for /pʰ/ and "dh" for /dʰ/.
- The retroflex stops represented by the IPA symbols [ʈ] and [ɖ] are typically transliterated as [ṭ] and [ḍ], respectively.
- The affricates tʃ and dʒ are transcribed as "c" and "j," respectively.

Aibi research, management and engineering journal. Volume 13, Number 2, May - August 2025, ISSN 2346-030X PP 1-11.

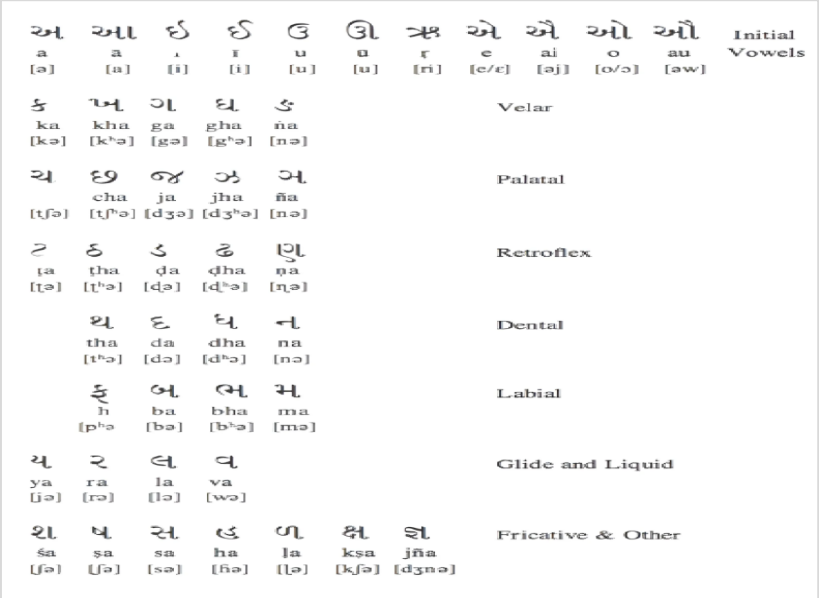


Figure 1: Gujarati vowels and Consonants [6].
Source: Own elaboration.

1. The Gujarati script encompasses three distinct symbols representing sibilant sounds, namely ś, ṣ, and s. However, it is important to note that both the symbols ś and ṣ are phonetically pronounced as the voiceless postalveolar fricative sound, represented by the International Phonetic Alphabet (IPA) symbol ʃ.
2. The glottal fricative ɦ is often represented by the transliteration h.
3. The nasal sound represented by the symbol ŋ is transcribed as ṇ. The script has symbols for the palatal and velar nasals (ñ, ṅ); nevertheless, these symbols are phonetically realized as the sound /n/.
4. The retroflex liquid sound represented by the International Phonetic Alphabet symbol [ɭ] is often transliterated as [l].
5. The glide "w" is transcribed as the letter "v" in transliteration [7].

The melodious Gujarati tongue dances with a symphony of eight enchanting vowel sounds. Except for the charming notes of [e] and [o], each vowel reveals a delightful twist of nasalization, accompanied by whispered and bold variants. Within the vibrant landscape of the Gujarati language, one can find both petite and elongated vowels; yet, they shy away from being distinctive phonemes.

	Front	Central	Back
High	i		u
High-mid	e		o
Mid		ə	
Low-mid	ɛ		ɔ
Low		a	

Figure 2: Gujarati Vowels [7].
Source: Own elaboration.

The melodic cadence of vowel length in Sanskrit has sadly faded away. Vowels stretch their wings when they are accompanied by nasal sounds or when they gracefully conclude a word. Aside from the sounds /e/ and /o/, Gujarati elegantly separates the realms of oral and nasal, as well as murmured and non-murmured vowels. In the definitive embrace of word-final status, the elevated and subdued vowels of the /e/ and /o/ families exhibit distinct variations. The phonetic characters // and // blossomed into existence during the 15th century. Over time, Old Gujarati gracefully transformed into Rajasthani and Middle Gujarati. Words borrowed from English generously contribute to the /ae/ sound [8].

The Gujarati language presents an intricate tapestry composed of 31 consonants, systematically classified into 20 stops, 3 fricatives, 3 nasals, and 5 liquids and glides. These stops and nasals emerge from five distinct points of articulation: labial, dental, retroflex, palatal, and velar. The palatal stops may be regarded as affricates in their unique charm. Each collection of stops in the Indo-Aryan lineage showcases a splendid fusion of voiceless and voiced consonants, alongside unaspirated and aspirated forms. This captivating four-way distinction holds its ground within the Indo-Aryan realm amidst the vast expanse of Indo-European tongues, as the ancient Proto-Indo-European dialect offered only a simpler three-way variation.

The appealing retroflex consonants that adorn Gujarati, generated immediately beneath the alveolar ridge, lack their origins in Indo-European ancestry, even though they are incorporated into Sanskrit. The genesis of these linguistic traits is likely intertwined with the influence of the Dravidian language. Additionally, Gujarati boasts a retroflex liquid that has no lineage traced back to Sanskrit [9].

		Labial	Dental	Retroflex	Palatal	Velar	Glottal
Stop	Voiceless	p pʰ	t tʰ	ʈ ʈʰ		k kʰ	
	Voiced	b bʰ	d dʰ	ɖ ɖʰ		g gʰ	
Affricate	Voiceless				tʃ tʃʰ		
	Voiced				dʒ dʒʰ		
Fricative	Voiceless		s		ʃ		
	Voiced						ɦ
Nasal		m	n	ɳ			
Liquid			l r	ɭ			
Glide		w			j		

Figure 3: Gujarati Consonants [7].
Source: Own elaboration.

Same as other languages Gujarati has a three genders, masculine (Narajāti), feminine (Nārījāti) and neuter (Nān'yatara jāti). One can identify the masculine (Narajāti), feminine (Nārījāti) and neuter (Nān'yatara jāti) by asking question respectively કેવો (kēvō), કેવી (Kēvī), કેવું (kēvuṁ) [10].

1. Anusvara

The term "Anusvara" refers to a sign that is often used in several Indic scripts to indicate a nasal sound. This symbol is generally transliterated as ⟨ṃ, ṁ, M,m⟩. The precise pronunciation of a word might change depending on its position inside the word and the specific language in which it is used. Within the framework of ancient Sanskrit, the term "anusvara" designates the specific nasal sound in question, irrespective of its written manifestation. There are many rules of anusvar in gujarati grammar, but researcher focus only five rules. Let us discuss with example [7–11].

Table 1: Description of Anusvara Rule.

Rule Number	Description
A1	Masculine (Narajāti), words never have an anusvar, nor do related inflected adjectives or inflected forms of verbs.
A2	There is no Anusvara to feminine words, but If the feminine word is complimentary (Mānārtha) plural, then it comes as anuswar.
A3	Neuter (Nān'yatara jāti), words have a anusvar.
A4	Anusvara occurs when the subjects (Kartā) in the sentence belong to different genders.
A5	When two more than two subjects (Kartā) joined with or [Athavā], anusvar take a place as per last word gender.

Source: Own elaboration.

Table 2: Example of Anusvara Rule.

Rule Number	Gujarati	English	Description
A1	ભાઈ સત્યપ્રેમી **હતા**. (Bhā'ī satyaprēmī hatāṁ.) [Right Sentence.] ભાઈ સત્યપ્રેમી **હતાં**. (Bhā'ī satyaprēmī hatā. [Wrong Sentence.]	Brother was a truth lover.	Brother is a Masculine (Narajāti). Because of that reason **હતા** has no anusvar.
A2	મોટાં બહેન **આવ્યાં**. (Mōṭāṁ bahēna āvyāṁ.) [Right Sentence.] મોટાં બહેન **આવ્યા**. (Mōṭāṁ bahēna āvyā.) [Wrong Sentence.]	Elder sister came.	Sister is feminine (Nārījāti), Because of that reason **આવ્યાં** has anusvar.
A3	નાનું છોકરું **રમતું હતું**. (Nānuṁ chōkaruṁ ramatuṁ hatuṁ.) [Right Sentence.] નાનું છોકરું **રમતુ હતુ**. (Nānuṁ chōkaruṁ ramatu hatu.) [Wrong Sentence.]	The little boy was playing.	છોકરું (chōkaruṁ) is a Neuter word, because of that reason **રમતું હતું** has anusvar.
A4	નર-નારી અને બાળકો સૌ કોઈ સૂઈ **ગયાં હતાં**. (Nara-nārī anē bāḷakō sau kō'ī sūī gayāṁ hatāṁ.) [Right Sentence.] નર-નારી અને બાળકો સૌ કોઈ સૂઈ **ગયા હતા**. (Nara-nārī anē bāḷakō sau kō'ī sūī gayā hatā.) [Wrong Sentence.]	Men, women and children were all asleep.	Men, women and children, are many gender in one sentence [masculine (Narajāti), feminine (Nārījāti)], because of that reason **ગયાં હતાં** has a anusvar.
A5	મેં કોઈ પત્ર અથવા રાજીનામું **આપ્યું** નથી. (Mēṁ kō'ī patra athavā rājīnāmuṁ āpyuṁ nathī.) [Right Sentence.] મેં કોઈ પત્ર અથવા રાજીનામું **આપ્યુ** નથી. (Mēṁ kō'ī patra athavā rājīnāmuṁ āpyu nathī.) [Wrong Sentence.]	I have not submitted any letter or resignation.	Last subject (Kartā) રાજીનામું is a neuter (Nān'yatara jāti), because of that reason **આપ્યું** has a anusvar.

Source: Own elaboration.

Now, if we examine the masculine (Narajāti), feminine (Nārījāti) and neuter (Nān'yatara jāti) by asking question respectively કેવો (kēvō), કેવી (Kēvī), કેવું (kēvuṁ) from given example,

Bhai [ભાઈ]……. કેવો (kēvō)

Sister [બહેન]…….. કેવી (Kēvī)

Little Boy [છોકરું]… કેવું (kēvuṁ)

Please note that English Translation of કેવો (kēvō), કેવી (Kēvī), કેવું (kēvuṁ) is – 'How…' only. No one can predict, when one ask English translated question for identification to know gender [7, 12, 13].

2. Rhasva and Dīrgha

Generally all students feel hesitant while spelling. Especially Short-Rhasva E-i (હ્રસ્વ ઇ) and Dīrgha-Long E-e (દીર્ઘ ઈ – ી), as well as Short-Rhasva U-u (હ્રસ્વ ઉ – ુ) and Dīrgha-Long U-oo (દીર્ઘ ઊ- ૂ) [7, 12, 13].

Table 3: Description and Example of Rhasva and Dīrgha Rule.

Rule Number	Description	Example
B1	In words starting with 'Tri (ત્રિ)' and 'Pri' (પ્રિ), put a short 'i'.	Triphala (ત્રિફળા), Trishul (ત્રિશુળ), Trishanku (ત્રિશંકુ), Trirango (ત્રિરંગો), etc.
B2	In words starting with 'Pri' (પ્રિ), put a short 'i'.	Print (પ્રિન્ટ), Prince (પ્રિન્સ), Priye-Beloved (પ્રિય) etc.
B3	When In words with both letters e and i, long 'e-ઈ-ી' in the first letter then short 'i-ઇ-િ' in the second letter.	Rīti (રીતિ), prīti (પ્રીતિ), bhīti, (ભીતિ), gīti, (ગીતિ), kīrti, (કીર્તિ), śrīti(શ્રીતિ)
B4	In the words related to 'it-ઇત', use હ્રસ્વ ઇ – િ.	Prērita (પ્રેરિત), prōtsāhita (પ્રોત્સાહિત), utsāhita (ઉત્સાહિત), kalaṅkita (કલંકિત), icchita (ઇચ્છિત).
B5	In words when 'ek-ઈક' is found at the end of the word, only the Mātrā 'ઇ' –િ 'i' is used.	નૈતિક (Naitika), પ્રમાણિક (pramāṇika), દૈનિક (dainika), ભૌગોલિક (bhaugōlika), વૈજ્ઞાનિક (vaijñānika).
B6	In words related to 'Iya-ઈયા', use only use હ્રસ્વ ઇ – િ.	દરિયા (Dariyā), રૂપિયા (rūpiyā), વાણિયા (vāṇiyā), કડિયા (kaḍiyā), ગાંઠિયા (gāṇṭhiyā).
B7	In words when 'ee-ઇય' is found at the end of the word, add long 'e-ઈ'- ી.	રાજકીય (Rājakīya), રાષ્ટ્રીય (rāṣṭrīya), વિદ્યાકીય (vidyākīya), માનનીય (mānanīya).
B8	Some words where both 'i-' ઇ – િ available.	સ્થિતિ (Sthiti), તિથિ (tithi), ટિકિટ (ṭikiṭa), ગિરિ (giri), મિતિ (miti), ભૂમિતિ (bhūmit) , સમિતિ (samiti)
B9	Letter which come before ref (રેફ) has a long 'E-U' - 'ઈ-ઊ'.	કીર્તન (Kīrtana), તીર્થ (tīrtha), જીર્ણ, (jīrṇa), મૂર્તિ (mūrti), સ્ફૂર્તિ (sphūrti), ચૂર્ણ (chūrṇa), સૂર્ય (sūry)
B10	Letter which come before 'Y-ય', has a short 'i'	નિષ્ક્રિય (niṣkriya), પ્રિય (priya), નિયામક (niyāmaka), ક્ષત્રિય (kṣatriya)
B11	If 'ઈશ-ish', 'ઈન્દ્ર-Indra' is arrive end of a word, then Long 'e-ઈ' is there.	સત્તાધીશ (Sattādhīśa), ન્યાયાધીશ (n'yāyādhīśa), યોગેન્દ્ર (yōgēndra), ભોગીન્દ્ર (bhōgīndra)
B12	Hrasva 'e' comes in feminine form of word.	તપસ્વી-તપસ્વિની (Tapasvī-tapasvinī), વિદ્યાર્થી-વિદ્યાર્થિની (vidyārthī-vidyārthinī), યોગિ-યોગિની (yōgi-yōginī), માયાવી-માયાવિની (māyāvī-māyāvinī)
B13	After adding the noun suffix 'તા-ta' or 'ત્વા-tva', the long 'ઈ-e' at the end is lost.	ઉપયોગી-ઉપયોગિતા (Upayōgī-upayōgitā), તેજસ્વી-તેજસ્વિતા (tējasvī-tējasvitā), સ્વામી-સ્વામિત્વ (svāmī-svāmitva), ઓજસ્વી-ઓજસ્વિતા (ōjasvī-ōjasvitā).
B14	The word has an e-ઈ, u-ઊ hrasva before a jōḍākṣara (two constant connected).	ઉત્સાહ (Utsāha), રુદ્ર (rudra), લુચ્ચો (luccō), ક્લિષ્ટ (kliṣṭa), પરિશિષ્ટ (pariśiṣṭa), હુલ્લડ (hullaḍa), જુસ્સો (jus'sō).

Source: Own elaboration.

In AI, researchers have two approaches: Rule-based and machine learning for computational linguistics work. In rule-based systems, the decision logic of the system is established at the outset and offers little flexibility after it is deployed. Contrarily, artificially intelligent machines acquire knowledge in an ongoing manner with the need for human guidance. The decision between a rule-based system and a system that uses machine learning is contingent upon the level of stringency required for parameters, considerations about effectiveness as well as training expenses, and whether the rules will be devised by a data science team or an algorithm. Here researchers discuss rule based work [14-15].

II. RULE BASED

A rule-driven mechanism in the realm of computer science is a computational architecture that harnesses rules to articulate domain-specific insights and utilizes universal reasoning to tackle challenges within that sphere. In the vibrant decade of the 1970s, two distinct varieties of rule-driven mechanisms emerged within the field of artificial intelligence. Production frameworks apply if-then rules to derive actions contingent upon specific conditions. Logic programming frameworks employ rules that produce conclusions from scenarios predicated on if-then propositions. The contrasts and correlations between these two types of rule-driven mechanisms have sparked considerable misunderstanding and confusion. Both varieties of rule-driven mechanisms utilize either forward or backward chaining, unlike imperative programming, which executes commands in a linear fashion. Logic programming frameworks embody a logical interpretation, whereas production frameworks do not possess this characteristic [16–18].

a. Benefits and drawbacks of rule-based systems [19–21]

- Precision: Rule-based systems function based on the principles of cause and effect, and are limited to operating within the boundaries of their predefined set of rules. Therefore, the rules of the system function as protective barriers to guarantee exactness and correctness.
- User-friendly interface: Rule-based systems may effectively complete jobs and repetitive procedures using small amounts of uncomplicated data. This facilitates the process of developers in creating, using, and troubleshooting them.
- Velocity: Rule-based systems, when properly trained, can produce prompt and effective judgments, since they do not allow for any ambiguity or development. Their restricted boundaries guarantee prompt replies.
- However, due to its simple nature, rule-based systems often fail to meet the following criteria:
- Narrow focus: Rule-based systems are deterministic and lack the ability to acquire new knowledge, thereby restricting their functionality to the boundaries of their initial programming. The addition of excessive regulations may impede the efficiency of a system and bring intricacy.
- Immutable: Inherently, rule-based systems are static and lack scalability. Modifying current regulations or integrating new ones might generate laborious and costly complexities.
- Limited cognitive abilities: The effectiveness of a rule-based system is solely reliant on the quality of the rules established by its authors, and it lacks the ability to make autonomous judgments outside of those regulations. Consequently, the system's actions are limited to its stated programming and will expose any deficiencies or oversights in the initial set of principles.

b. Proposed a Rule Based Error Detection and Correction System [22–24]

Rule-based Error Detection and Correction System is intriguing. It operate by utilizing a predefined set of guidelines that identify and correct frequent spelling and typography mistakes, then applying these guidelines to the incorrectly written word. These principles are the opposite or reverse of typical mistakes, based on intuition. Every accurate word produced by this procedure is considered a recommendation for modification. The rules are equipped with probabilities, allowing for the ranking of ideas by aggregating the probabilities associated with the applicable rules. Distance editing is a specific instance of a rule-based approach but with restrictions on the available rules.

Fig. 4 is a proposed rule-based system. In this system, firstly, input text is tried to match the rules. If all rules match, a final suggestion or answer may come. Or, through matching with a dictionary word, a final suggestion or answer will be there as a final state. Please note that the whole system works on lexicon-based work. Rules come from tables 1, 2, and 3. These are very helpful rules for identifying spelling mistakes.

III. NORVIG SPELL CHECKER ALGORITHM

It implements an Gujarati corpus to Norvig's algorithm to create a spell checker. Spell checking technologies learn proper spelling from a corpus and use the right word as a reference if a word is misspelled. Norvig's algorithm learns word-by-word spellings from a text or list. The most likely rectification issue may be formalized as a sequence decoding problem. Suppose w is received. Thus, we seek a word c from every candidate correction that maximizes its intended corrective given the word w:

$$\text{argmax}_{c \in candidates} P(c|w) \qquad (1)$$

This is Bayes' Theorem corresponding:

$$\text{argmax}_{c \in candidates} P(c)\, P(w|c) / P(w) \qquad (2)$$

One can factor P(w) out since it's the same for every candidate c:

$$\text{argmax}_{c \in candidates} P(c)\, P(w|c) \qquad (3)$$

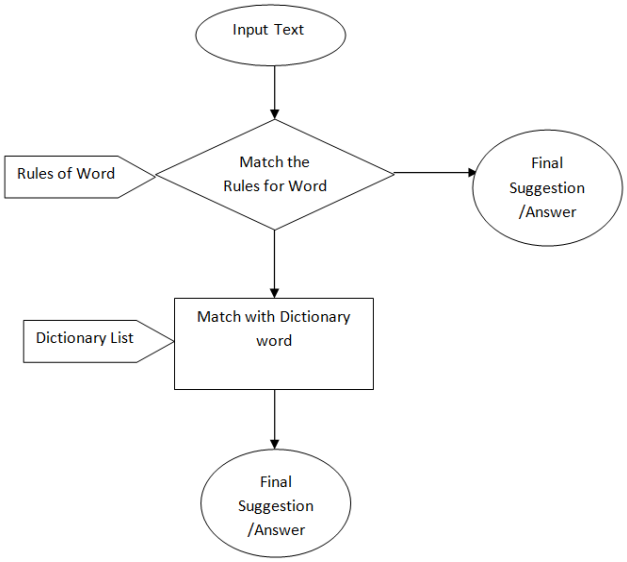


Figure 4: System for Rule Based Error Detection and Correction System.
Source: Own elaboration.

- Selection Mechanism: The candidate with the highest overall probability is chosen using the argmax function.
- Candidate Model: ccc represents a member of the candidates. It provides information on the candidate corrections, ccc, that should be considered.
- Language Model: P(c)P(c)P(c) represents the probability that ccc appears as a word in English text. For example, the word "the" appears in approximately 7% of English text, so P(the)P(\text{the})P(the) would be 0.07.
- Error Model: P(w|c)P(w|c)P(w|c) indicates the probability that the writer intended to type ccc but mistakenly typed www. For example, P(teh|the)P(\text{teh}|\text{the})P(teh|the) is relatively high, while P(theeexyz|the)P(\text{theeexyz}|\text{the})P(theeexyz|the) would be extremely low.

The code provided below is a core component of the Norvig Algorithm for the English language.

```
def edits1(word):
    "All edits that are one edit away from `word`."
    letters    = 'abcdefghijklmnopqrstuvwxyz'
    splits     = [(word[:i], word[i:])    for i in range(len(word) + 1)]
    deletes    = [L + R[1:]               for L, R in splits if R]
    transposes = [L + R[1] + R[0] + R[2:] for L, R in splits if len(R)>1]
    replaces   = [L + c + R[1:]           for L, R in splits if R for c in
letters]
    inserts    = [L + c + R               for L, R in splits for c in letters]
    return set(deletes + transposes + replaces + inserts)

def edits2(word):
    "All edits that are two edits away from `word`."
    return (e2 for e1 in edits1(word) for e2 in edits1(e1))
```

Figure 5: Norvig Program- Letters a to z.
Source: Own elaboration.

Now we are just replacing the English letters with Gujarati whole barakhdi as per given below data.

```
def replace or insert(word):

 letters                                                                =
"અઆઇઈઉઊઋૠઌૡએઐઓઔઅંઅઃકકાકિકીકુકૂકૃકૄકૢકૣકંકઃખખાખિખીખુખૂખૃખૄખૢખૣખંખઃગગાગિગીગુગૂગૃગૄગૢગૣગંગઃઘઘાઘિઘીઘુઘૂ
ઘૃઘૄઘૢઘૣઘંઘઃચચાચિચીચુચૂચૃચૄચૢચૣચંચઃછછાછિછીછુછૂછૃછૄછૢછૣછંછઃજજાજિજીજુજૂજૃજૄજૢજૣજંજઃઝઝાઝિઝીઝુઝૂઝૃઝૄઝૢઝૣઝંઝઃટટા
ટિટીટુટૂટૃટૄટૢટૣટંટઃઠઠાઠિઠીઠુઠૂઠૃઠૄઠૢઠૣઠંઠઃડડાડિડીડુડૂડૃડૄડૢડૣડંડઃઢઢાઢિઢીઢુઢૂઢૃઢૄઢૢઢૣઢંઢઃણણાણિણીણુણૂણૃણૄણૢણૣણંણઃતતાતિતી
તુતૂતૃતૄતૢતૣતંતઃથથાથિથીથુથૂથૃથૄથૢથૣથંથઃદદાદિદીદુદૂદૃદૄદૢદૣદંદઃધધાધિધીધુધૂધૃધૄધૢધૣધંધઃનનાનિનીનુનૂનૃનૄનૢનૣનંનઃપપાપિપીપુપૂપૃપૄપૢપૣપંપઃફફાફિફીફુફૂફૃફૄફૢફૣફંફઃબબાબિબીબુબૂબૃબૄબૢબૣબંબઃભભાભિભીભુભૂભૃભૄભૢભૣભંભઃમમામિમીમુમૂમૃમૄમૢમૣમંમઃયયાયિયીયુ
યૂયૃયૄયૢયૣયંયઃરરારિરીરુરૂરૃરૄરૢરૣરંરઃલલાલિલીલુલૂલૃલૄલૢલૣલંલઃવવાવિવીવુવૂવૃવૄવૢવૣવંવઃશશાશિશીશુશૂશૃશૄશૢશૣશંશઃષષાષિષીષુષૂષૃ
ષૄષૢષૣષંષઃસસાસિસીસુસૂસૃસૄસૢસૣસંસઃહહાહિહીહુહૂહૃહૄહૢહૣહંહઃળળાળિળીળુળૂળૃળૄળૢળૣળંળઃક્ષક્ષાક્ષિક્ષીક્ષુક્ષૂક્ષૃક્ષૄક્ષૢક્ષૣક્ષંક્ષઃજ્ઞજ્ઞા
જ્ઞાજ્ઞીજ્ઞુજ્ઞૂજ્ઞૃજ્ઞૄજ્ઞૢજ્ઞૣજ્ઞંજ્ઞઃ"
  return [l + c + r[1:] for l, r in split(word) if r for c in letters]
```

Now, researchers call,

```
checker = SpellChecker("Gujarati2.txt")
```

Here Gujarati2.txt is a corpus of Gujarati language.

```
print(replace("મહાબ્લી"))
['અહાબ્લી', 'આહાબ્લી', 'ઇહાબ્લી', 'ઈહાબ્લી', 'ઉહાબ્લી', 'ઊહાબ્લી', 'ઋહાબ્લી', 'એહાબ્લી', 'ઐહાબ્લી', 'ઑહાબ્લી', 'ઔહાબ્લી',
'અહાબ્લી', 'ંહાબ્લી', 'અહાબ્લી', ':હાબ્લી', 'કહાબ્લી', 'કહાબ્લી',……]
```

Here, researchers use મહાબ્લી word for replace or insert function.
Now, in second way researchers insert or replace with letters ="ા,િ,ી,ુ,ૂ,ૃ,ૄ,ૅ,ે,ૈ,ૉ,ો,ૌ,્"

```
def replace or insert(word):
letters ="ા,િ,ી,ુ,ૂ,ૃ,ૄ,ૅ,ે,ૈ,ૉ,ો,ૌ,્"
return [l + c + r[1:] for l, r in    split(word) if r for c in letters]
```

Now, researchers call,

```
checker = SpellChecker("Gujarati2.txt")
```

Here Gujarati2.txt is a corpus of Gujarati language.

```
print(replace("મહાબ્લી"))
['ાહાબ્લી', ',હાબ્લી', 'િહાબ્લી', ',હાબ્લી', 'ીહાબ્લી', ',હાબ્લી', 'ુહાબ્લી', ',હાબ્લી', 'ૂહાબ્લી', ',હાબ્લી', 'ૃહાબ્લી', ',હાબ્લી',
'ૄહાબ્લી',……]
```

Here, researchers use મહાબ્લી word for replace or insert function.

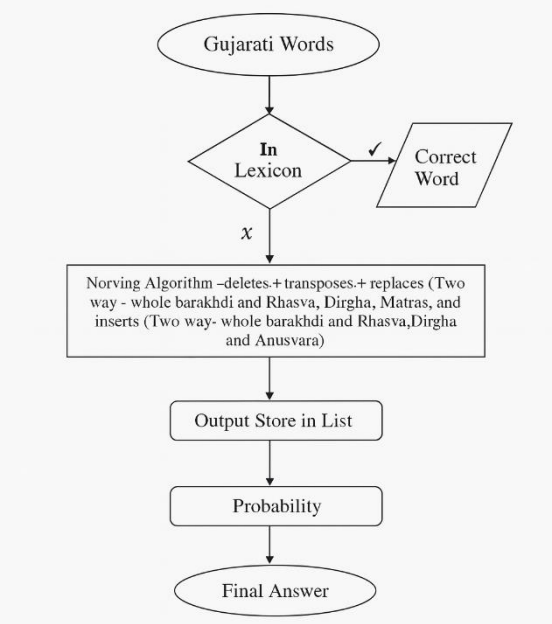


Figure 6: Norvig Algorithm for Gujarati Text.
Source: Own elaboration.

Fig. 6 is a flow diagram of the proposed system using the Norvig algorithm for Gujarati text. First of all, Gujarati text enters in the system; if the words match, no kind of suggestion arrives. Otherwise, four steps will be there: deletes, transposes, replaces (two ways whole barakhdi and Rhasva, Dīrgha, Matras, and Anusvāra) and inserts (two ways whole barakhdi and Rhasva). Following these four steps, researchers will store suggestive words in the list, calculate the probability, and then finalize the process [25-26-27].

IV. RESULT AND ANALYSIS

In result part, Norvig suggest one method. Where right end side incorrect words and left end side correct words. Colon (:) use has a separation. The whole spell-testset1.txt has below data set.

મહાબ્લી: મહાબ્લિ મહાબ્લ મહાબ મબ્હાલ
બ્રિજેશ: બ્રિજશ બ્રજેશ બ્રિજે બજેશ બ્રિશજે બ્રશજે
લોકતંત્રકરણ: લોતંત્રિકરણ લોકતત્રિકરણ લોકતંત્રકરણ લોકતત્રરણ
સૂલભ: સુલભ સલભ સુભ સુલ
આકર્ષિત: આકર્ષિત આકર્ષત આકર્ષિ આકર્ષિ

Now, researchers focus on the following functions developed by Norvig for testing purposes: Where one correct word is on the left end side after a colon (:) and an and an incorrect word on the right end side. Both words match; if the words match, the know percentage has more value, and if they do not match, the unknown percentage has few value. In the output section, researcher's analysis says that if all things match answer should be 100% match and if few words match, among 4 incorrect word 1 word is match only 25% word is match as per output section.

```
def Testset():
    with open('spell-testset1.txt') as fp:
      for line in fp:
        x=
        (right, wrongs) = line.replace("\n", "").split(':')
        print(line.replace("\n", "").split(':'))
        for wrong in wrongs.split():
          print((right,wrong))
          x.append((right,wrong))
        spelltest(x)

result=Testset()
print(result)
```

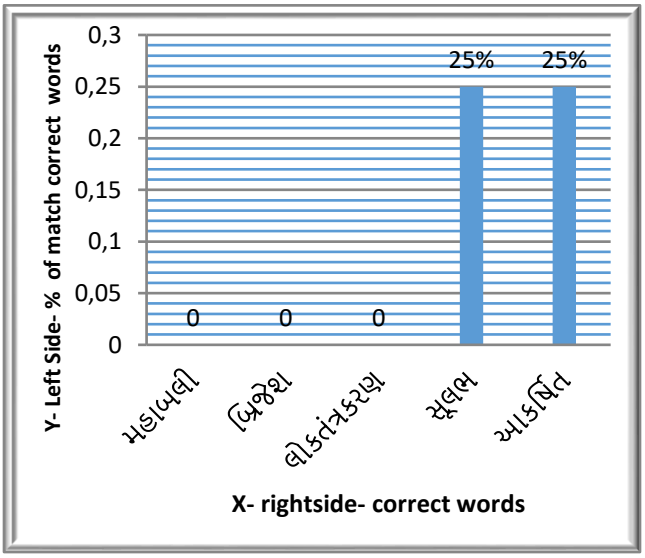
Output-

```
[('લોકતંત્રકરણ', 'લોતંત્રિકરણ'), ('લોકતંત્રકરણ', 'લોકતત્રિકરણ'), ('લોકતંત્રકરણ', 'લોકતંત્રકરણ'), ('લોકતંત્રકરણ', 'લોકતત્રરણ')]
['લોતંત્રિકરણ']
લોતંત્રિકરણ 0
correction(લોતંત્રિકરણ) => લોતંત્રિકરણ (0); expected લોકતંત્રકરણ (0)
0% of 4 correct (25% unknown)
['લોકતત્રિકરણ']
લોકતત્રિકરણ 0
correction(લોકતત્રિકરણ) => લોકતત્રિકરણ (0); expected લોકતંત્રકરણ (0)
0% of 4 correct (50% unknown)
['લોકતંત્રકરણ']
લોકતંત્રકરણ 1
['લોકતત્રરણ']
લોકતત્રરણ 1
correction(લોકતત્રરણ) => લોકતત્રરણ (0); expected લોકતંત્રકરણ (0)
25% of 4 correct (75% unknown)
['સૂલભ', ' સુલભ સલભ સુભ સુલ']
('સૂલભ', 'સુલભ')
('સૂલભ', 'સલભ')
('સૂલભ', 'સુભ')
('સૂલભ', 'સુલ')
[('સૂલભ', 'સુલભ'), ('સૂલભ', 'સલભ'), ('સૂલભ', 'સુભ'), ('સૂલભ', 'સુલ')]
{'સલ', 'લભ', 'સભ'}
સલ 0
correction(સુલભ) => સલ (15); expected સૂલભ (0)
0% of 4 correct (25% unknown)
{'સલ', 'લભ', 'સભ'}
સલ 0
correction(સલભ) => સલ (15); expected સૂલભ (0)
0% of 4 correct (50% unknown)
{'સભ'}
સભ 0
correction(સુભ) => સભ (5); expected સૂલભ (0)
0% of 4 correct (75% unknown)
{'સલ'}
સલ 0
correction(સુલ) => સલ (15); expected સૂલભ (0)
0% of 4 correct (100% unknown)
['આકર્ષિત', ' આકર્ષિત આકર્ષત આકર્ષિ આકર્ષિ']
('આકર્ષિત', 'આકર્ષિત')
('આકર્ષિત', 'આકર્ષત')
('આકર્ષિત', 'આકર્ષિ')
('આકર્ષિત', 'આકર્ષિ')
[('આકર્ષિત', 'આકર્ષિત'), ('આકર્ષિત', 'આકર્ષત'), ('આકર્ષિત', 'આકર્ષિ'), ('આકર્ષિત', 'આકર્ષિ')]
['આકર્ષિત']
```


Aibi research, management and engineering journal. Volume 13, Number 2, May - August 2025, ISSN 2346-030X PP 1-11.

આકર્ષિત 1
['આકર્ષત']
આકર્ષત 1
correction(આકર્ષત) => આકર્ષત (0); expected આકર્ષિત (0)
25% of 4 correct (25% unknown)
['આકર્ષિ']
આકર્ષિ 1
correction(આકર્ષિ) => આકર્ષિ (0); expected આકર્ષિત (0)
25% of 4 correct (50% unknown)
['આકર્ષિ']
આકર્ષિ 1
correction(આકર્ષિ) => આકર્ષિ (0); expected આકર્ષિત (0)
25% of 4 correct (75% unknown)
None

Graph 1 is testing analysis of correct and incorrect words. Researchers found that among many words, right end side words match with left end side, where only two words, સૂલભ and આકર્ષિત, are 25% correct words from the left end side list. From 4 words, only 1 word is a match [28-29-30].



Graph 1: X-Y Norvig Algorithm Testing.
Source: Own elaboration.

V. CONCLUSION

This study suggested a system for finding and fixing mistakes in Gujarati texts that uses the Norvig Algorithm and the ideas of rhasva, dīrgha, and anusvāra. In this section, the writers provide a brief overview of the Gujarati language, complemented by insights regarding consonants and vowels. A quick look at machine learning and rule-based systems was done to help understand the proposed rule-based system and the Norvig algorithm for two types of text: whole barakhdi with rhasva, dīrgha, and anusvāra. Figures 4 and 5, respectively, show the rules-based and Norvig-based algorithm flowcharts. Finally, for testing purposes, Norvig introduces a new concept in the text file: right and left side words. The Norvig Algorithm's output provides valuable insights. Researchers discovered correct words from a variety of sources during testing. In the development set, researchers received 0, 0, 0, 25, and 25% of the correct phrases from the provided testing words, respectively. The researchers may have used an exceedingly challenging test set, or my straightforward model may simply be incapable of achieving an accuracy rate of 80% or 90%. Through Figure 4 and 5, researcher get an idea rule based and machine learning approach will be getting less accuracy and only Norvig Algorithm cannot solve the spelling error of Gujarati Text. Researcher will be proposed a Deep Learning based hybrid novel approach to solve spelling errors of Gujarati language. For that researchers may use RNN, LSTM, Bi-LSTM, GRU and BERT. Overall, this paper can help to know the grammatical rules like rhasva, dīrgha, and anusvāra for Gujarati Spell Checker using Norvig Algorithm.

VI. REFERENCES

[1] K. Vyas “Vikas, Swarup, and Unviversity Granth Nirman Board”, ISBN: 978-93-81265-98-7.
[2] B. Thakur, M. Upreti, P. Sahagal, H. Joshi, S. Hardikar, V. Ratalani, B.B. Jyoti “Gujarati: A Textbook for Learning Gujarati Through Hindi From the Central Institute of Indian Languages”.
[3] W. S. Tisdall, “A Simplified Grammar of the Gujarati Language: Together With A Short Reading Book and Vocabulary”. London: Kegan Paul, Trench, Trúbner, 1892.
[4] P. J. Mistry, “Gujarati writing,” in The World’s Writing Systems, B. Daniels, Ed. Oxford University Press, 1996.
[5] G. Cardona “A Gujarati Reference Grammar”. University of Pennsylvania Press, 1965.

[6] N. Patel and D. Patel, "'Implementation Approach of Indian Language Gujarati Grammar's Concept "sandhi' using the Concepts of Rule-based NLP," 8th International Conference on Computing for Sustainable Global Development (INDIA Com), New Delhi, India, vol. 2021, pp. 481–485. 2021.
[7] Gujarati. Available at: languagesgulper.com.
[8] W. S. Tisdall, "A simplified grammar of the Gujarati language," vol. 22, Рипол Классик, 1892.
[9] G. Cardona and B. Suthar, "The Indo-Aryan languages," in Gujarati, G. Cardona, Jain, and Dhanesh, Eds. Routledge, ISBN 978-0-415-77294-5, 2003.
[10] B. Suthar, 'A Brief Outline of Gujarati Parts-of-Speech (POS)', A Nirman Foundation Project. Philadelphia: Department of South Asia Studies, University of Pennsylvania, 2003.
[11] R. Soni and G. Lekhan-Paddhati (Gujarati), Publisher: Gurjar, ISBN-: 9351753409.
[12] R. B. K. P. Trivedî, Higher Grammar of the Gujarati Language. Macmillan and Company, 1919.
[13] Y. Vyas and G. B. Vyakran (Gujarati), By Publisher: Balvinod Prakashan, ISBN: 978-93-8478-006-7.
[14] What Is a Rule-Based System? Available at: j-paine.org.
[15] S. V. Maniya, Ms. J. Sheth, and Dr. K. Lad, "Compression Technique based on Dictionary approach for Gujarati Text," Int. J. Eng. Res. Dev. eISSN : 2278-067X, pISSN : 2278-800X. Available at: http://www.ijerd.com, vol. 4, no. 8, pp. 101–108, Nov. 2012.
[16] A. Abraham "Intelligent Systems: A Modern Approach. Springer Science+Business Media," pp 149 2011. ISBN 978-3-642-21004-4.
[17] H. Liu, A. Gegov, and F. Stahl, "Categorization and construction of rule based systems," in Commun. Comput. Inf. Sci.. EANN vol. 459, 2014.
[18] B. G. Buchanan and R. O. Duda, "Principles of rule-based expert systems," in Adv. Comput. Elsevier, vol. 22, 1983. doi: 10.1016/S0065-2458(08)60129-1.
[19] J. V. Julio and C. Pérez, "24th International Florida Artificial Intelligence Research Society", FLAIRS - 24. Sara & González-Cristóbal: Sonia & Serrano, 2011.
[20] A. J. Szanser, "Automatic error-correction in natural languages," Inf. Storage Retrieval, vol. 5, no. 4, pp. 169–174, 1970. doi: 10.1016/0020-0271(70)90045-8.
[21] P. Kumar, A. Kannan, and N. Goel, "Design and implementation of NLP-based spell checker for the Tamil language," in Proc. 1st International Electronic Conference on Applied Sciences. Basel, Switzerland: MDPI, Nov. 10–30, 2020.
[22] C. Patil, R. Rodrigues, and R. Ron, "Auto-spelling checker using natural language processing,nternational research," J. Eng. Technol. (IRJET) e-ISSN: 2395-0056, vol. 07, pp-794–796, Aug 2020.
[23] A. N. M. Fahim Faisal, M. A. Rahman, and T. Farah, "A rule-based Bengali grammar checker," Fifth World Conference on Smart Trends in Systems Security and Sustainability (WorldS4), London, United Kingdom, vol. 2021, pp. 113–117, 2021.
[24] Y. Gondaliya, P. Kalariya, B. Y. Panchal, and A. Nayak, "A rule-based grammar and spell checking," SAMRIDDHI A J. Phys. Sci. Eng. Technol., vol. 14, no. 1, pp. 48–54, 2022.
[25] A. Istiak & J. maliha, T. Zarin, T. Reza, S. M. Salim, Hossain, and Dilshad, "Spell Corrector for Bangla Language Using Norvig's Algorithm and Jaro–Winkler Distance" Bulletin of Electrical Engineering and Informatics. vol. 10. pp. 1997-2005, 2021.
[26] H. Patel, B. Patel, and K. Lad, "Jodani: A spell checking and suggesting tool for Gujarati language," Data Sci. Eng. (Confluence), (Noida, India) 11th International Conference on Cloud Computing, vol. 2021, pp. 94–99, 2021.
[27] A. Bhansali, A. Chandravadiya, B. Y. Panchal, M. H. Bohara, and A. Ganatra, "Language identification using combination of machine learning algorithms and vectorization techniques," 2nd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE), Greater Noida, India, vol. 2022, pp. 1329–1334, 2022.
[28] B. Y. Panchal and A. Shah, "Spell checker using Norvig algorithm for Gujarati language," in. ICSMDI 2024. Algorithms for Intelligent Systems, R. Asokan, D. P. Ruiz, and S. Piramuthu, Eds. Smart Data Intelligence. Singapore: Springer, 2024.
[29] J. Baxi and B. Bhatt, "A bidirectional LSTM-based morphological analyzer for Gujarati," Nat. lang. processing. Cambridge University Press, pp. 1–17, 2024.
[30] N. G. Patel and D. B. Patel, "NLP-based processing of Gujarati compound word sandhi's generation and segmentation," in International Conference on Universal Threats in Expert Applications and Solutions. Singapore: Springer Nature Singapore, pp. 263–271, Jan., 2024.