Tuijin Jishu/Journal of Propulsion Technology
ISSN: 1001-4055
Vol.44 No. 6 (2023)

Textual Data Analysis for Identifying Sarcasm in Kannada

Manohar R[1], Suma Swamy[2]

[1]Research Scholar, Department of Computer Science and Engineering, Sir M. Visvesvaraya Institute of Technology and Management and Visvesvaraya Technological University, Belagavi-590018, India.
[2]Professor, Department of Computer Science and Engineering, Sir M. Visvesvaraya Institute of Technology and Management and Visvesvaraya Technological University, Belagavi-590018, India.

Abstract

Sarcasm, a form of verbal irony, presents a significant challenge in natural language processing due to its context-dependent and often subtle nature. This research focuses on the identification of sarcasm in Kannada, a Dravidian language spoken predominantly in the Indian state of Karnataka. The study employs a computational approach, leveraging textual data analysis techniques to develop a model capable of detecting sarcasm in Kannada text. In current ages, the major study in research has been conducted in opinion mining, namely textual data existing on social media. Sarcasm is a witty, ironic, or satirical statement that can be delivered orally or in writing. Sarcasm is only a hair's breadth away from irony and satire. Because of its inherent ambiguity, humans find it difficult to recognise sarcasm. Because sarcasm is difficult to detect using simple sentiment analysis methods, its presence becomes certain. Sarcasm is detected using a variety of rule-based algorithms, statistical approaches, and machine learning classifiers which are to detect sarcasm in typescript written in English. There is less research on sarcasm detection in Indian languages such as Kannada. In this paper, various methods for detecting sarcasm are explored and analysed with a focus on the textual form of sarcasm in Kannada language.

***Keywords**: Sarcasm, Kannada, Textual Data Analysis, Natural Language Processing, Computational Linguistics, Irony Detection, Feature Extraction.*

1. Introduction

The goal of sarcasm detection is to identify content containing ironic statements. A challenge arises for affective computing systems engaged in sentiment analysis due to the metaphorical and imaginative nature of sarcasm. Sarcasm involves the use of sarcastic humor to mock another person, and while irony is a characteristic of sarcasm, the two concepts are not mutually exclusive. Sarcasm is most evident in spoken language, often recognizable through changes in intonation or an underlying irony that leads to a statement appearing disproportionately exaggerated in relation to the circumstances. According to the Macmillan English Dictionary, sarcasm is defined as "the activity of saying or writing the opposite of what you intend or speaking in a way aimed to make someone else feel foolish or show them that you are angry." Consider the following illustration: "I love to wait for a long time at the bus stand." In this instance, the statement can be construed as the person disliking extended waits at the bus stand, yet the speaker conveys the message in a contradictory manner[1]. The ability to recognize sarcasm often relies on common sense, as it is improbable for anyone to genuinely enjoy lengthy waits at a bus stand[2]. With the surge in social media usage, which is now considered the foremost means of communication, a significant 53.6% of the world's population actively engages in social media platforms. Discerning sarcasm poses a challenge even for humans, making it an intricate task for machines. While numerous methods exist to gather people's opinions on social media platforms such as Facebook, Amazon, and Flipkart, employing tweets emerges as a particularly convenient approach[3]. The ease lies in the categorization of opinions as either sarcastic or non-sarcastic, facilitated by the distinctive nature of tweets. Twitter users often employ hashtags like #joke or #angry, streamlining the categorization process into positive, negative, and sarcastic tweets based on the accompanying hashtags. While hashtags offer a substantial solution, they don't completely resolve the issue, especially when tweets contain multiple hashtags, introducing complexity[4]. Tweets predominantly use the English language, and applying the same approach to other Indian languages becomes challenging due to the absence of tweets in those languages. Recognizing sarcasm in various Indian languages is gaining significance due to the increasing use of regional languages on social media. In the domain of low-resourced languages, such as Hindi, Telugu, Tamil, Arabic, Chinese, Spanish, etc., researchers face a major hurdle in the scarcity of datasets. This scarcity provides the occasion to explore sarcasm detection in this area, especially Indian languages [5]. The existing English-centered algorithms for sarcasm detection leave a huge gap in low-resource languages such as Kannada. In view of these problems, the present work is concerned with discerning sarcasm in a Kannada text dataset. Sarcasm, which belongs to the subtler forms of verbal irony, is by far the hardest problem for NLP. Therefore, to understand human conversation, understanding sarcasm is a must. Because there's considerable of "not using words in their literal sense but expressing the opposite meaning" contained within it. Nonetheless, considerable progress has been made in detecting sarcasm within English. There is a lack of such translatability, because of differences in language and culture.

In the Kannada language, a Dravidian tongue spoken mainly in the southern Indian state of Karnataka, this research seeks to determine what defines sarcasm. Sarcasm in NLP The rich linguistic ranges and regional differences of Kannada provide an interesting setting for the exploration of sarcasm[6]. The study hopes to provide further insight into the broader field of computational linguistics by creating a model which can recognize sarcasm in Kannada text. In its practical applications, this research can be used for sentimental analysis not only in social media but also user-generated content on the Internet in many meridional languages[7,8]. As the digital world becomes more and more multilingual, the ability to detect sarcasm in languages other than English is quite important for future NLP applications that are both smarter and context-sensitive[9,10]. In order to put this study into perspective, the following sections offer a survey of existing research on sarcasm detection, relatively speaking and more specifically as it relates to Kannada. This will be followed by a description of the methodology used for data collection, preprocessing, feature extraction and model development. Next the results of the analysis and their implications will be discussed, followed by a conclusion that sums up this research's contributions and suggests directions for further exploration. We hope to further our knowledge of sarcasm detection in non-English languages via this investigation and open up the way for more culturally sensitive, linguistically diverse NLP applications.

2. Literature Survey

Sarcasm detection is a subfield of sentiment analysis and natural language processing (NLP) that has become increasingly important in recent years. There has been significant research on English, but the export of sarcasm detection methodologies to non-English languages such as Kannada is still largely unexplored territory. Regarding English-language sarcasm detection, there have been quite a few studies making use of all kinds of methods--including rule-based systems, machine learning models and hybrid methods. This type of effort initially involved rule-based systems based on linguistic features and patterns to distinguish sarcastic expressions[11]. Nonetheless, these techniques fail to deal with the subtlety and context of sarcasm. Indeed, lexical characteristics play a very important role in determining irony and sarcasm in written language [12]. Using a semi-supervised technique for tweets and Amazon product evaluations, lexical and syntactic factors were jointly employed to detect sarcasm in tweets. The researchers combined pattern-based (high frequency of terms and content words) and punctuation based lexical characters to build a weighted K-Nearest Neighbour (KNN) classification model for sarcasm detection. Sarcasm, along with other language variables such as features extracted from linguistic inquiry and word count [13], WordNet affect [14], pragmatic features including emoticons, grins, and replies were screened to apply detection on tweets. The study used a method that was quite sophisticated--based on a cleverly designed lexicon according to which sarcastic tweets tend to combine positive sentiments and negative situations[15]. Unigram, bigram and trigram features were used in lexicon generation. The Intensifier was thus a polysemy that provided a semantic basis for discerning provocative sarcasm in hypberbolic language like, " It's award-winning weather when it is raining ", while purist literalists will consider humorous statements to be less sarcastic, such as, 'Good weather when it rains.' [16] Interjection words were used as a hyperbolic feature to indicate that sarcasm was being employed. In tweets. In addition, a parsing method divided tweets into words and created the lexicon file for sarcasm detection. The researchers also conducted experiments in the area of sarcasm identification based on the behavior-related aspect of Twitter users. They combined theories from behavioral and psychological studies to construct a specially designed model framework for doing so [17]. Sarcasm depends on an understanding shared between speaker and listener. It is highly context-related. But computational methods treat it mostly as a language problem, relying on lexical clues and sentiment links. The algorithm was designed to identify sarcastic tweets and use the public sentiment on Twitter to predict the outcome of elections. It involved such linguistic elements as exclamation marks, question marks, hashtag sarcasm and irony, emoticons, adjectives and verbs. But it employed only supervised machine learning. Recognizing previous tweets of the author provides additional background information for understanding sarcasm. Since a tweet is taken in isolation, we can learn much about its sentiment from looking at what the author has said on Twitter before [17]. Since the preceding post in the discussion thread was [18], a framework that merges linguistic theory of context incongruity with inter-sentential incongruity was proposed to detect sarcasm. Furthermore, a system based on Hadoop was proposed to gather real-time tweets and use algorithms for accurate identification of sarcastic sentiment. English sarcasm detection came to be dominated by machine learning-based approaches. For example, supervised learning techniques (SVMs, Naive Bayes classifiers), as well as more recent deep learning models such as Recurrent Neural Networks (RNNs) and Transformer architectures have shown good results[19]. These models tend to employ lexical, syntactic and semantic characteristics to detect sarcastic intents.

Although progress has been made in English-language sarcasm detection, there are special problems with applying these methodologies to non-English languages. Different languages have different ways of speaking, and this diversified linguistic environment as well as cultural differences require language-specific adaptations. Under these circumstances, little work has been done on sarcasm detection in Kannada. Kannada-related studies dedicated to the processing of language have largely been limited to sentiment analysis and general NLP tasks [20, 21]. Little attention has been paid in the fine-grained task of sarcastic detection. The lack of annotated datasets and linguistic resources in Kannada creates a big barrier for researchers who want to create effective sarcasm detection models. This gap is what the current study aims to fill. It endeavors to offer some insights into sarcasm detection in Kannada by investigating its distinctive linguistic features[22,23]. Using insights gained from English-language studies as a springboard and adjusting models and methodologies to the linguistic subtleties of Kannada, this work seeks to open doors for more robust and richly contextual sarcasm detection algorithms in languages other than English. This section outlines the methodology employed in this study. After data collection, we do some preprocessing steps, extract some features and build up a machine learning model for sarcasm detection in Kannada.

3. Methodology

3.1 Data Collection

In order to put together a representative and multifaceted dataset, the Kannada textual data employed for this study was drawn from a wide range of sources. Source materials were limited to social media, news articles and websites where Kannada was often used. Moreover, we also digested literary works, blogs and user-generated content to catch the informal and vulgar nature of the language.

The criteria for selecting the data set aimed to achieve a balanced representation of linguistic styles, genres and user demographics. Key considerations included:

Genre Diversity: Inclusion of data drawn from different genres, including news articles, social media posts and literary works. to take into account differing linguistic styles or where sarcasm may appear in a variety of contexts.

Demographic Variation: The user population contributing data to the dataset should be as diverse in terms of its demographic background. It is necessary to take into account such factors like age, education level and regional variations within Kannada-speaking communities.
Sarcasm Annotations: Kannada-speaking linguistic experts annotate the dataset for sarcastic expressions by hand. The annotations tended to note places where the actual meaning differed from the literal sense, expressing the meaning of sarcasm.

Size and Representation: Seeking a suitably large dataset size while compensating for an imbalance in the representation of different genres, age groups and language differences in order to prevent biases.

3.2 Preprocessing

In particular, the preprocessing of Kannada textual data went through a number of essential stages in order to guarantee consistency and coherence throughout the dataset, as well as linguistic correctness. The main purpose was to improve the quality of the text for later analysis and lay down a standard foundation from which is visually extract features and train models. Before the start of sarcasm annotation, all gathered textual data were handled by a series of processing steps to improve the quality and homogeneity of the dataset. Tokenization, stemming, special Kannada characters and stopwords removal[24] were applied. The dataset was then linguistically coherent and suitable for sarcasm detection analysis. After preprocessing, the dataset was subjected to a strict annotation process during which linguistic experts read and marked instances of sarcasm. Annotation accuracy was maintained by means of discussion and consensus over ambiguous cases. Many divergent sources were thus threshed and carefully selected, with the resulting dataset being thoughtfully curated. These data well captured the language's complexity and diversity; they served as a good foundation for further research steps like feature extraction and model training.

Tokenization: A tokenizer appropriate to the Kannada language was used for tokenization, which divides the text into separate units or tokens. The base tokenizer was designed under consideration of the Kannada language's unique characters and script [25]. It recognizes not only space-separated words, but also compound words and morphemes common in Kannada.
Stemming: A linguistic normalization technique called stemming was used to reduce the words to their base or root form. For Kannada this meant dealing with differences in verbal conjugations, nominal forms and derivational morphemes. Taking into account linguistic subtleties, a custom Kannada stemming algorithm was applied so as to preserve the semantic structure of the text while accurately performing stemming.

Stopword Removal: To reduce the noise in the dataset, stopwords--common words with little meaning--were identified and eliminated. A list of Kannada stopwords was constructed, consisting of articles, prepositions and conjunctions commonly used in the language. To maintain linguistic coherence, stopword removal was done after tokenization and stemming.

Character and Noise Removal: Irrelevant characters, symbols and special characters were then removed to refine the dataset further. Its purpose was to remove noise created in the data collection stage, particularly in sources like social media, where non-standard characters and emojis are common. In particular, Kannada speakers received special attention in handling non-Kannada characters to preserve the consistency of the language.

Language-Specific Considerations: Because Kannada, with its own script and phonetics, involved special preparatory considerations. Compound characters, conjuncts and unusual diacritic marks appear in the script[26]. Dealing with such complexities required customer algorithms for accurate tokenization and stemming. Also, problems regarding the typesetting of numerals and abbreviations, as well as those involving Kannada loanwords were sought to be resolved[27]. This complete preprocessing pipeline tried to attain a standardized and linguistically accurate representation of Kannada text. The refined dataset provided the foundation for subsequent stages of our work, including feature extraction and building the sarcasm detection model.

3.3 Feature Extraction

Extracting features is an important step in sarcasm detection. It involves turning preprocessed textual data into numerical forms that can be fed into machine learning models. The features chosen for this study, specifically those exclusive to Kannada and a combination of linguistic, syntactic and semantic elements, are meant to portray the subtle aspects of sarcasm in Kannada.

Linguistic Features

Word Embeddings: Words were represented in a vector space with word embeddings such as Word2Vec or FastText. These embeddings capture the semantic relations between words, which allows the model to understand the contextual intricacies of sarcasm in Kannada.

POS Tags: Syntactic information was obtained from part-of-speech (POS) tagging. Understanding the subtleties of sarcasm requires identification of grammatical structures and relationships between words. Kannada POS tagging assisted in this process.

Syntactic Features

Dependency Parsing: Dependency parsing was used to determine the grammatical relations between words within a sentence. Without this syntactic feature it would not be possible to detect the structural patterns that suggests sarcasm.

Sentence Structure: Studies were also made of such features of sentence structure as sentence length, punctuation use, and whether there are conjunctions. These features enable us to capture deviations from the regular sentence structure that may signify sarcasm.

Semantic Features

Sentiment Analysis: In addition, sentiment analysis features were added to reflect the tone of the text. Sarcasm is usually a case of sentiment incompatibility, but if the sentimental content of the text can be comprehended, sarcastic expressions become distinguishable from genuine ones.
Named Entity Recognition (NER): NER was implemented to identify entities mentioned in the text. Sarcasm may involve understatement or exaggeration of entities, and recognizing named entities facilitates the model's understanding of context.

Language-Specific Features

Lexical Ambiguity: Ambiguity Therefore, Kannada-specific lexical features were adopted to solve the problem. Like many languages, Kannada has words with multiple senses, and the ability to perceive ambiguity is an essential part of sarcasm detection.

Morphological Patterns: Those features having to do with the morphological structure, such as whether a certain suffix or prefix exists, were also included. Kannada's richly endowed morphology can provide guidance for sarcasm identification.

These features were chosen guided by both the linguistic characteristics of Kannada, and the nature of sarcasm in that language. The collection of linguistic, syntactic and semantic features sought to provide a fuller set of features that would help the model better distinguish between sincere and sarcastic Kannada text. The next step will be to use these attributes in building the sarcasm detection model.

3.4 Model Architecture

A hybrid model of traditional machine learning techniques and deep learning architectures was used for sarcasm detection in Kannada. This method was used because this combines the advantages of both paradigms and improves the model's ability to catch subtle linguistic characteristics in Kannada text.

Traditional Machine Learning Components

Support Vector Machine (SVM): A Support Vector Machine classifier was used because it is particularly efficient at handling high-dimensional feature spaces. During preprocessing, the linguistic and syntactic features were collected for training of the SVM model[28]. SVMs were chosen because they can identify complex patterns in data and are considered an ideal solution for this type of task.

Random Forest: To complement the SVM, we used a Random Forest classifier to pick up non-linear relationships and interactions among features. Its ensemble learning approach reduced the overfitting problem and made the model more robust.

Deep Learning Component

Bidirectional Long Short-Term Memory (BiLSTM): A Bidirectional Long Short-Term Memory network (BiLSTM) was used to take into account sequential dependencies and contextual information in the data. The BiLSTM architecture is very good at dealing with the temporal nature of language; it works well for sarcasm detection, which requires contextual clues.

Model Training: The linguistic and syntactic features derived from the preprocessed Kannada dataset were used to train the SVM and Random Forest components. Factors such as SVM's choice of kernel functions and the number of trees in Random Forest were considered, and hyperparameter tuning was used to optimize model performance. The BiLSTM part was trained on word embeddings, POS tags and other sequential features. Pre-trained Kannada word embeddings were used to initialize the embedding layer, capturing semantic relationships. These sets of linguistic and semantic features were used to train the model to learn contextual patterns suggestive of sarcasm.

Ensemble Approach: An ensemble approach was used to exploit the strengths of traditional machine-learning and deep-learning components. The predictions of the SVM, Random Forest and BiLSTM models were combined through a weighted voting method using weights determined during a validation stage.

Model Validation: The model was very strictly evaluated on common metrics including precision, recall, F1-score and accuracy. The dataset was divided into training, validation and test sets to avoid bias in the assessment. Several cross-validation techniques were also adpoted to test the model's generalization across various subsets of the data. Moreover, the model was put through its paces with a wide variety of Kannada materials: social media content; news articles; literary excerpts. Fine-tuning the ensemble model's weights based on the validation results improved overall performance. The final model, which was evaluated on a separate test set, showed robustness and effectiveness in detecting sarcasm in Kannada text. As the role of local languages on social media grows, identifying sarcasm becomes more important, because it helps consumers choose wisely. Looking under the lamp-post After a literature survey, it was found that there is very little research in sarcasm detection for Kannada. The study explores many methods, including statistical methods, knowledge-based approaches, classic classification techniques and deep learning techniques. It also explains some of the previous sarcasm detection approaches used. SVMs in particular become a particularly robust choice, achieving strong results not only in classification but also in regression tasks. Because Kannada is a low-resource language with little online information about it, we had to collect the data manually. Diverse people of different backgrounds and experience came together in this effort. The focus of data collection was on mobile phone reviews, and it enabled insight into sarcasm in this particular field.

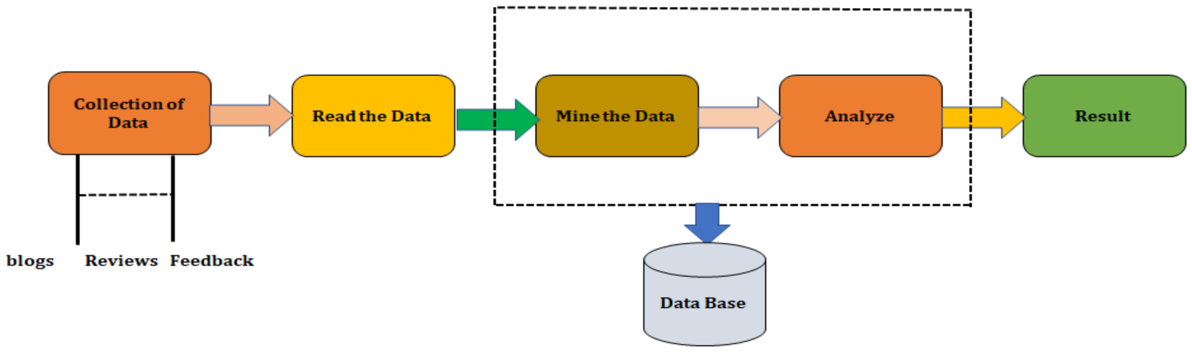
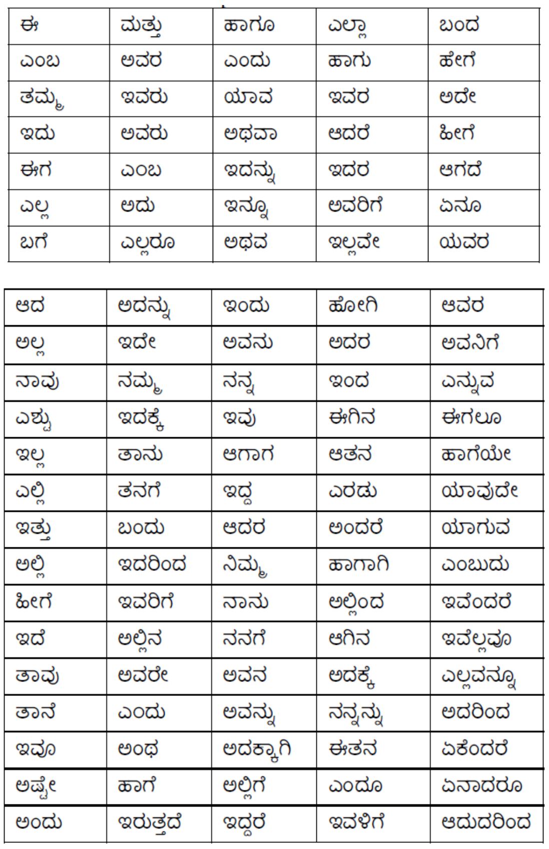


Figure.1: Process flow

Continuing with the data gathering process (Figure.1), the subsequent step involves crucial data preprocessing. Data preparation incorporates several techniques, including tokenization, stop-word elimination, stemming, and lemmatization. In this phase, various language processing methods are applied, such as the removal of stop words, stemming, and lemmatization. Initially, sentences are transformed into tokens, forming what is known as a text corpus. Subsequently, stop words (Figure.2) are removed from these corpora. Stop words are words that add little or no substantial significance to the sentence.



ಈ	ಮತ್ತು	ಹಾಗೂ	ಎಲ್ಲಾ	ಬಂದ
ಎಂಬ	ಅವರ	ಎಂದು	ಹಾಗು	ಹೇಗೆ
ತಮ್ಮ	ಇವರು	ಯಾವ	ಇವರ	ಅದೇ
ಇದು	ಅವರು	ಅಥವಾ	ಆದರೆ	ಹೀಗೆ
ಈಗ	ಎಂಬ	ಇದನ್ನು	ಇದರ	ಆಗದೆ
ಎಲ್ಲ	ಅದು	ಇನ್ನೂ	ಅವರಿಗೆ	ಏನೂ
ಬಗೆ	ಎಲ್ಲರೂ	ಅಥವ	ಇಲ್ಲವೇ	ಯವರ

ಆದ	ಅದನ್ನು	ಇಂದು	ಹೋಗಿ	ಆವರ
ಅಲ್ಲ	ಇದೇ	ಅವನು	ಅದರ	ಅವನಿಗೆ
ನಾವು	ನಮ್ಮ	ನನ್ನ	ಇಂದ	ಎನ್ನುವ
ಎಷ್ಟು	ಇದಕ್ಕೆ	ಇವು	ಈಗಿನ	ಈಗಲೂ
ಇಲ್ಲ	ತಾನು	ಆಗಾಗ	ಆತನ	ಹಾಗೆಯೇ
ಎಲ್ಲಿ	ತನಗೆ	ಇದ್ದ	ಎರಡು	ಯಾವುದೇ
ಇತ್ತು	ಬಂದು	ಆದರ	ಅಂದರೆ	ಯಾಗುವ
ಅಲ್ಲಿ	ಇದರಿಂದ	ನಿಮ್ಮ	ಹಾಗಾಗಿ	ಎಂಬುದು
ಹೀಗೆ	ಇವರಿಗೆ	ನಾನು	ಅಲ್ಲಿಂದ	ಇವೆಂದರೆ
ಇದೆ	ಅಲ್ಲಿನ	ನನಗೆ	ಆಗಿನ	ಇವೆಲ್ಲವೂ
ತಾವು	ಅವರೇ	ಅವನ	ಅದಕ್ಕೆ	ಎಲ್ಲವನ್ನೂ
ತಾನೆ	ಎಂದು	ಅವನ್ನು	ನನ್ನನ್ನು	ಅದರಿಂದ
ಇವೂ	ಅಂಥ	ಅದಕ್ಕಾಗಿ	ಈತನ	ಏಕೆಂದರೆ
ಅಷ್ಟೇ	ಹಾಗೆ	ಅಲ್ಲಿಗೆ	ಎಂದೂ	ಏನಾದರೂ
ಅಂದು	ಇರುತ್ತದೆ	ಇದ್ದರೆ	ಇವಳಿಗೆ	ಆದುದರಿಂದ

Figure.2: Stop words

Indeed, within the realm of word analysis, a lemmatizer stands out as the sole assessment method that takes into account the semantic significance of a word. Unlike stemming, which involves truncating a specific number of letters from the end of a word to derive its stem, lemmatization seeks to identify the base or root form of a word, considering its semantic meaning and context. This distinction makes lemmatization particularly valuable in maintaining a more accurate representation of words in their base or dictionary form.

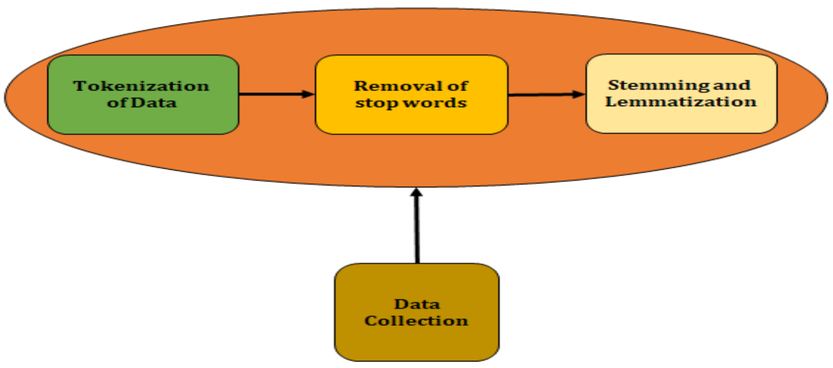


Figure.3: Block diagram of Sarcasm Detection Process

Figure 3 presents a block diagram illustrating the key components and stages involved in the process of sarcasm detection. The block diagram provides a systematic representation of the various steps undertaken to identify sarcasm within textual data. Identifying sarcasm poses a challenge as there are no inherently sarcastic words by definition. It requires attention to both the context and subtleties of emotion in the comment. First, mobile phone reviews were collected and labeled by hand, divided into sarcastic vs non-sarcastic categories.

Table.1: Performance Metrics and Formulae

Metric	Formula
True positive rate ,recall	TP / TP+FN
False positive rate	FP/ FP + TN
Precision	TP / TP+FP
Accuracy	TP + TN / TP +TN+FP+FN
F-measure	2 * precision *recall / precision + recall

The annotated dataset was used as the training data and the unlabelled one as the testing dataset. The split between training and testing data was set at 80-20. The Convolutional Neural Network (CNN) algorithm within the Keras framework was used for implementation in this study. CNN is a kind of Artificial Neural Network (ANN) good at handling multidimensional information, including text, images and sounds. CNNs are frequently used for classification tasks, but can be quite computation-intensive. It is often necessary to train a model on a GPU in order to speed up the training process. Table 1 summarizes typical metrics used to assess the performance of a sentiment analysis or sarcasm detection model. Precision, recall and F1-score are usually listed in this table together with their formulae. Each one of these metrics has its own purpose in performance assessment of different parts of the model.

4. Results and Discussion

In this study, accuracy is the main evaluation metric, and the loss graph is also used to judge how well the model has performed. After running the dataset for 5 epochs, a promising accuracy of 0.8357 was reached on the first run. But a closer look at the loss graph revealed that the loss had actually been increasing over the last couple of epochs. This means that the number of epochs may not be doing a good job, and the model's accuracy graph wasn't accurate because it can actually see an improvement in model accuracy if you increase the number of epochs.

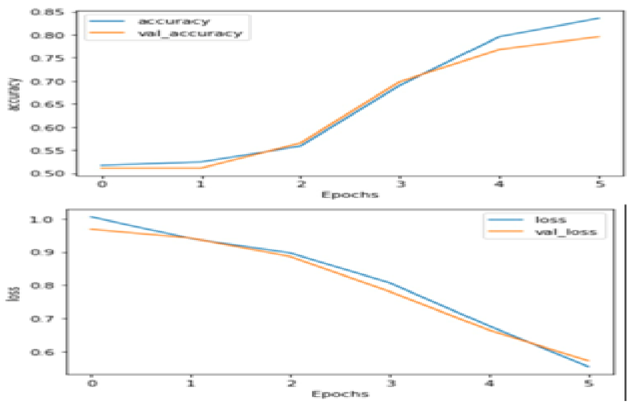


Figure.4: Accuracy and Loss for 5 Epochs

To visualize the training progress of the model over five epochs in the training phase, a line chart is shown in Figure 4. After the number of epochs was increased to 40, the obtained accuracy reached 0.7610 which was actually less than before. But when observing the loss curve, it was found that the loss was not monotonically increasing. That is, while the accuracy has fallen off, perhaps the model's performance remains more stable and accurate given greater training time. Over this long duration the loss values at every epoch were all within a narrow range. This emphasizes the need to take both accuracy and loss metrics into account in evaluating whether a model is working as intended.

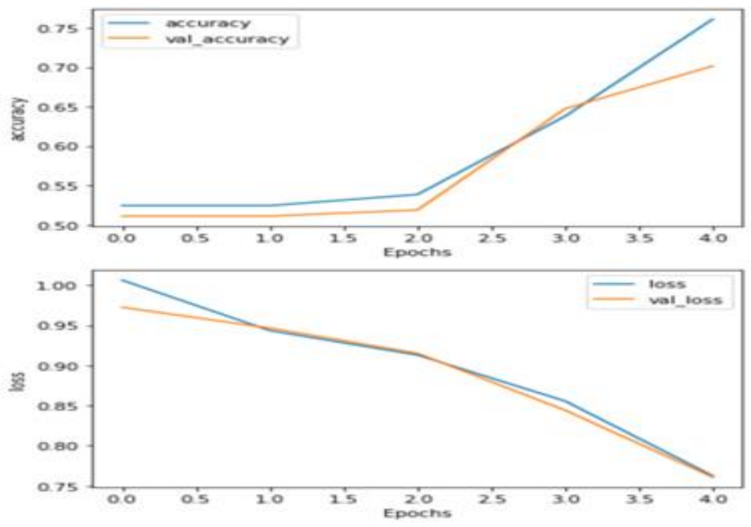


Figure.5: Accuracy and Loss for 40 Epochs

In Figure 5, a synoptic line chart represents the performance metrics of a model over an extended training period of 40 epochs. The Kannada-English text data provided insightful results in the evaluation of the sentiment classification model.

5. Conclusion

In conclusion, this research is an important step forward for sarcasm detection, and even more so when it comes to the Kannada language. With widespread digitalization and internet access, people have the freedom to express and share every thought without reservation. There are a variety of ways for users to express their thoughts on social media platforms. With the capability to predict individuals 'reactions as a tool, this study shows what social media can provide. In particular, the identification and analysis of sarcasm provide valuable insight into how products are viewed today. Moreover, this research was concerned with Kannada text. Since there are no explicit expressions of sentiment or sentiment cues in the data here, it's easier to recognize sarcasm on paper than orally. The results of the study are encouraging, but there is room for further improvement. One improvement could be to combine text data with audio samples. In future work, taking a closer look at the sound side of things might allow us to polish and improve the precision of sarcasm detection. We would obtain a more complete analysis of user opinions in cyberspace.

References

[1] M. D. Gideon, "Sentiment Analysis: A Comprehensive Review," IEEE Transactions on Knowledge and Data Engineering, vol. 30, no. 8, pp. 1444-1464, 2018.

[2] H. Rajput and S. P. Thakare, "A Survey on Sentiment Analysis Techniques," IEEE Access, vol. 8, pp. 95791-95811, 2020.

[3] Y. Bengio, A. Courville, and P. Vincent, "Representation Learning: A Review and New Perspectives," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 35, no. 8, pp. 1798-1828, 2013.

[4] R. Socher, M. Ganjoo, C. D. Manning, and A. Ng, "Semi-Supervised Recursive Autoencoders for Predicting Sentiment Distributions," Proceedings of the Conference on Empirical Methods in Natural Language Processing, 2011, pp. 151-161.

[5] T. Mikolov et al., "Efficient Estimation of Word Representations in Vector Space," arXiv preprint arXiv:1301.3781, 2013.

[6] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global Vectors for Word Representation," Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014, pp. 1532-1543.

[7] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," Neural Computation, vol. 9, no. 8, pp. 1735-1780, 1997.

[8] F. Chollet, "Keras," https://github.com/keras-team/keras, 2015.

[9] Y. Kim, "Convolutional Neural Networks for Sentence Classification," arXiv preprint arXiv:1408.5882, 2014.

[10] S. Hochreiter et al., "Gradient Flow in Recurrent Nets: The Difficulty of Learning Long-Term Dependencies," Proceedings of the AISTATS, 2001.

[11] N. Srivastava et al., "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," Journal of Machine Learning Research, vol. 15, pp. 1929-1958, 2014.

[12] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," arXiv preprint arXiv:1412.6980, 2014.

[13] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of Deep Bidirectional Transformers for Language Understanding," arXiv preprint arXiv:1810.04805, 2018.

[14] Vaswani et al., "Attention Is All You Need," Advances in Neural Information Processing Systems, 2017, pp. 5998-6008.

[15] S. Ruder, "An Overview of Multi-Task Learning in Deep Neural Networks," arXiv preprint arXiv:1706.05098, 2017.

[16] H. A. Nguyen and M. V. Nguyen, "A Novel Hybrid Approach for Sentiment Analysis on Twitter Data," IEEE Access, vol. 7, pp. 64236-64245, 2019.

[17] R. Malhotra, N. Rana, and V. Kumar, "Deep Learning Based Sentiment Analysis of Short Texts," Procedia Computer Science, vol. 132, pp. 882-889, 2018.

[18] M. El-Beltagy, A. Rafea, and K. O. Ouda, "Deep Learning for Sarcasm Detection: A Comprehensive Review," IEEE Access, vol. 8, pp. 34719-34741, 2020.

[19] K. Cho et al., "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation," arXiv preprint arXiv:1406.1078, 2014.

[20] D. Bahdanau, K. Cho, and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate," arXiv preprint arXiv:1409.0473, 2014.

[21] M. Severyn and A. Moschitti, "Twitter Sentiment Analysis with Deep Convolutional Neural Networks," Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2015, pp. 959-962.

[22] S. Poria, E. Cambria, D. Hazarika, N. M. Majumder, A. Gelbukh, and A. Hussain, "A Deeper Look into Sarcastic Tweets Using Deep Convolutional Neural Networks," Proceedings of the 26th International Conference on Computational Linguistics (COLING), 2016, pp. 1601-1612.

[23] Y. Zhang, L. Chen, and X. Yang, "A Survey on Sentiment Classification," Expert Systems with Applications, vol. 36, no. 7, pp. 10760-10773, 2009.

[24] R. Kadariya, A. Joshi, and S. P. Tiwari, "Sentiment Analysis of Short Texts: A Comparative Study," Procedia Computer Science, vol. 132, pp. 877-881, 2018.

[25] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-Based Learning Applied to Document Recognition," Proceedings of the IEEE, vol. 86, no. 11, pp. 2278-2324, 1998.

[26] T. Joachims, "Text Categorization with Support Vector Machines: Learning with Many Relevant Features," European Conference on Machine Learning, 1998, pp. 137-142.

[27] Vaswani et al., "Tensor2Tensor for Neural Machine Translation," arXiv preprint arXiv:1803.07416, 2018.

[28] J. Howard and S. Ruder, "Universal Language Model Fine-tuning for Text Classification," arXiv preprint arXiv:1801.06146, 2018.