IJNRD.ORG ISSN : 2456-4184
INTERNATIONAL JOURNAL OF NOVEL RESEARCH AND DEVELOPMENT (IJNRD) | IJNRD.ORG
An International Open Access, Peer-reviewed, Refereed Journal

A COMPARATIVE ANALYSIS OF STEMMING TECHNIQUES FOR SARCASM DETECTION

Pratibha Jaisingh[1], Dr. R.K.Dhuware[2]
1. Research Scholar, Department of Electronics and Computer Science, RTMNU, Nagpur
2. Head, Department of Computer Science, Dhote Bandhu Science College, Gondia

Abstract: Sarcasm detection plays a critical role in understanding sentiment in text data. Stemming is extensively used as a pre-processing tool in sarcasm Detection. This study aims to compare the performance of two popular machine learning algorithms, logistic regression and random forest, when applied to sarcasm detection using different stemming techniques. Despite the differences in model complexity, the results indicate minimal variation in performance metrics (accuracy, precision, recall, and F1-score) across the models. However, a qualitative analysis and evaluation of computational efficiency suggest key trade-offs between the two algorithms. Additionally, semantic preservation analysis highlights the importance of preprocessing in maintaining the nuances of sarcastic language. Future research directions include exploring more sophisticated feature extraction techniques and deep learning models for enhanced sarcasm detection.

Keywords: **Semantic preservation. Porter Stemmer, Lancaster Stemmer, Snowball stemmer.**

1.INTRODUCTION

Sarcasm detection is a challenging task in natural language processing (NLP), as sarcastic statements often involve a discrepancy between literal meaning and intent. Preprocessing steps such as stemming and tokenization are crucial for improving model performance.

It is now crucial to provide a variety of language processing technologies that can effectively handle the massive document bases due to the vast amount of digital data that is available for sarcasm detection. Building a vocabulary of terms and language models is a crucial effort in many applications of information retrieval (IR) and natural language processing (NLP). However, a lot of word morphological variants provide a big problem, particularly in languages with a lot of morphological diversity. One helpful pre-processing method that can deal with these variations is text stemming. The process of mapping a word's different morphological variations to their base forms is known as stemming. For example, stemming is used to map the words plays, played, playing, player, and players to their base form, play. [1].

The main lexical unit of a word with the most important semantic information is called the root word. Additionally, it is atomic in that it cannot break down into smaller components. Therefore, a stem as a proxy for a morphological root will reflect the essence of the meaning associated with a whole family of related words [2]. This is what makes stemming valuable to sarcasm Detection tasks as it enables semantic analysis without relying on lexical resources like thesauri, which are not always easily accessible and can be somewhat costly to build.

1.1 Stemming and Lemmatization

Stemming and lemmatization are frequently regarded as similar processes and categorized together. Both methods are interconnected and serve the purpose of simplifying the various forms of words in the text input. The primary distinction between the two techniques is found in their results. The result of stemming is a 'stem,' whereas lemmatization produces a 'lemma.' Stems typically carry different meanings and are often focused on specific tasks. The stem can either be a valid, comprehensible word (a free stem) or an invalid word that requires an affix to create a complete word (a bound stem). For example, 'perish' serves as a free stem, whereas 'dur' acts as a bound stem. Lemmas, in contrast, are recognized linguistic elements and represent the standard form of a lexeme. A lexeme encompasses all the various word forms that share a common meaning, while a lemma is a specific variant used to symbolize the lexeme. For instance, the words run, ran, runs, and running are distinct forms of a lexeme represented by the lemma 'run.

Stemmers are usually simpler to implement and operate more quickly, and the lower accuracy might be acceptable for certain applications. Lemmatizers, on the other hand, are more challenging to implement due to their connection to the semantics and part of speech (POS) of a sentence [3].

1.2 Reasons for carrying out the survey

The aim is to provide an in-depth analysis of different stemming techniques, detailing how each method works and highlighting their unique characteristics. This paper compares the effectiveness of different stemming techniques (Porter, Lancaster, and Snowball) using two machine learning algorithms—logistic regression and random forest—in the context of sarcasm detection. While these models differ in complexity, both are widely used for text classification tasks. The goal is to determine whether stemming techniques and model choice significantly affect the accuracy and efficiency of sarcasm detection models.
The rest of the paper is organized as Section 2 examines work done related to stemming methods. Section 3 outlines the methodology implemented in the experiment. Results are discussed in Section 4. The findings are discussed in section 5 and conclusions and future work are presented in Section 6.

2. Related Work

Lovins (1968) [4] was the first to create a stemmer aimed specifically at information retrieval tasks and proposed the concept of stemming using a collection of common suffixes like *SES, *ING, or *ATION. This algorithm led to the emergence of various subsequent algorithms and, more broadly, popularized the application of stemming as a useful tool in information retrieval. Jasmeet Singh and Vishal Gupta [5] explained that the various areas of information retrieval and natural language processing require certain text pre-processing tools for lexical, morphological, syntactic and semantic level analysis. Stemming is one of the numerous pre-processing tools and is useful in the areas of information retrieval and natural language processing such as text classification, clustering, searching, summarization, POS tagging, etc.

3. Methodology

3.1. Dataset

News Headline Dataset for Sarcasm Detection [6] consist of news headline along with is_sarcastic column which is 1 is the headline is sarcastic otherwise 0. Total 11724 sarcastic and 14985 non-sarcastic news headlines are there in the dataset.

3.2. Preprocessing Techniques

Stemming was performed on the text data to reduce words to their root forms, thereby simplifying the input for the machine learning models. Three stemming techniques were applied:

a) Porter Stemmer:

One of the most commonly used algorithms in NLP, known for its balance between stemming aggressiveness and semantic preservation.
One of the earliest and most used stemming algorithms in natural language processing is the Porter Stemmer. It was created by Martin Porter in 1980 and aims to eliminate frequent suffixes like "-ing," "-ed," and "-es" from English nouns in order to return them to their root forms. In order to iteratively reduce words to their stems, the Porter stemmer applies many rules in successive phases.
For example:

- "running" → "run"
- "happily" → "happi"
- "national" → "nation"

Strengths:
• Balanced Approach: The accuracy and aggression of the Porter stemmer are balanced. It usually retains a certain amount of semantic significance because it doesn't over-stem words.

• Widely Used: The Porter stemmer is frequently employed in information retrieval and text classification tasks because of its ease of use and efficacy.

Weaknesses:
• Rule-Based: The Porter stemmer's rules are set, thus they don't consider the word's context, which could lead to stemming errors (for example, stemming "fishing" and "fish" to "fish").

• Not Appropriate for Every Language: The Porter stemmer might not function as effectively in other languages because it was created for English.

b) Lancaster Stemmer

Compared to the Porter stemmer, the Lancaster Stemmer—also called the Paice-Husk stemmer—is a more aggressive stemming algorithm. Chris Paice developed it in 1990. The Lancaster stemmer iteratively reduces words to their stems using a lookup table and preset rules. Compared to the Porter stemmer, it uses more forceful truncation procedures, frequently reducing words to shorter root forms.

For example:

- "running" → "run"
- "happily" → "hap"
- "national" → "nat"

Strengths:
• Aggressive: When a significant reduction in the number of unique words is needed, the Lancaster stemmer can be helpful because it aggressively reduces words to their stems.

• Iterative Process: Several rules can be implemented in a single pass thanks to the algorithm's iterative application of the stemming rules.

Weaknesses:

• Over-Stemming: The Lancaster stemmer frequently over-stems words because of its aggressive attitude, which could cause meaning to be lost. When "happily" is shortened to "hap," for example, its original connotation is lost.
• Less Semantic Preservation: When it comes to jobs like sarcasm detection, where word choice and minute variations in meaning are crucial, over-stemming can cause words to become unrecognizable or challenging to understand in context.

c) Snowball Stemmer:

A modern algorithm that aims to provide more flexible and language-specific stemming compared to Porter. Martin Porter created the Snowball Stemmer, also known as the Porter2 stemmer, in 2001 as an enhanced and more adaptable version of the Porter stemmer. It is a component of the programming language Snowball, which was created especially for stemming algorithm development. Compared to the original Porter stemmer, the Snowball stemmer is thought to be more reliable, adaptable, and multilingual.

For example:

- "running" → "run"
- "happily" → "happi"
- "national" → "nation"

Strengths:

• Better Semantics: By offering superior stemming rules, the Snowball stemmer outperforms the Porter stemmer in terms of preserving semantic meaning and reducing errors.
• Multilingual Support: The Snowball stemmer is a flexible tool for multilingual applications since, in contrast to the Porter stemmer, it can be modified for use in other languages, including French, German, Spanish, and Italian.
• Balanced Aggression: The Snowball stemmer is appropriate for a variety of NLP tasks because it strikes a balance between the aggressive approach of the Lancaster stemmer and the conservative approach of the Porter stemmer.

Weaknesses:

• A Little More Complex: Although the Snowball stemmer is more precise and adaptable than the Porter stemmer, it is also a little more complicated; however, in actual use, this difference is negligible.

• Language-Specific Rules: Language-specific rules, which may need to be customized for languages other than English, determine how effective the Snowball stemmer is.

Summary of Differences in stemming methods are presented in Table 1

Table 1: Summary of differences between different stemming methods

Feature	**Porter Stemmer**	**Lancaster Stemmer**	**Snowball Stemmer**
Stemming Aggression	Moderate	High	Moderate
Semantic Preservation	Moderate	Low	High
Ease of Use	Easy	Easy	Easy
Customization	English-specific rules	English-specific rules	Multilingual
Typical Use Cases	General NLP tasks	Vocabulary reduction tasks	Multilingual NLP, general NLP tasks

The goal was to evaluate the impact of these stemmers on both the performance of sarcasm detection models and the preservation of meaning in the text.

3.3. Machine Learning Models

Two machine learning algorithms were employed:

- **Logistic Regression**: A linear model that predicts binary outcomes.

In spite of its name, it is a classification technique that forecasts the likelihood of a binary result (0 or 1) rather than a regression procedure. The model converts projected values into probabilities between 0 and 1 using a logistic function, sometimes referred to as the sigmoid function. The instance falls into one class (1 or "sarcastic") if the likelihood is greater than a predetermined threshold (usually 0.5); if not, it falls into the other class (0 or "non-sarcastic").

- **Random Forest**: An ensemble method that builds multiple decision trees and aggregates their predictions.

Random Forest (RF) includes a set of trees (decision trees) that run independently in this algorithm. Each branch has a _Gini index utilized for the acquisition decision branch. Here is the index estimated by Equation:

$Gin = 1- \sum_{i=1}^{c}(pi)$

Hereabouts, pi indicated the probability of i class, and c indicated all number of classes. [7]

Both models were trained on bag-of-words representations of the stemmed text, and their performance was compared using accuracy, precision, recall, and F1-score.

3.4. Semantic Preservation

We evaluated the semantic preservation of the text after stemming by calculating the cosine similarity between the original and stemmed sentences using word embeddings from the spaCy language model.

3.5. Evaluation Metrics

The models were evaluated using the following performance metrics:

- Accuracy: The proportion of correctly classified instances out of the total number of instances.
- Precision: The proportion of true positives out of all predicted positives.
- Recall: The proportion of true positives out of all actual positives.
- F1-Score: The harmonic mean of precision and recall, providing a balanced measure of model performance.
- Vocabulary Reduction**:** The total number of unique tokens after stemming, indicating the degree of word reduction

Additionally, we measured semantic preservation by calculating the cosine similarity between the original and stemmed sentences using word embeddings generated from the spaCy language model.

4. Results

The results of the experiments are presented in Table 2, which summarizes the performance of both logistic regression and random forest across the three stemming techniques

Table 2: Result of Experiment

Model	Stemmer	Accuracy	Precision	Recall	F1-Score	Vocabulary size	Semantic Preservation	Training Time (s)
Logistic Regression	Porter	0.79	0.79	0.71	0.75	17845	0.59	131.92 sec
Logistic Regression	Lancaster	0.78	0.77	0.71	0.74	16731	0.53	77.28 sec
Logistic Regression	Snowball	0.79	0.79	0.71	0.75	17695	0.58	77.47 sec
Random Forest	Porter	0.76	0.80	0.6	0.68	17845	0.59	158.72 sec
Random Forest	Lancaster	0.75	0.80	0.58	0.67	16731	0.53	128.76 sec
Random Forest	Snowball	0.75	0.79	0.59	0.68	17694	0.58	129.39 sec

4.1. Performance Metrics

The results demonstrate that across all stemming strategies, logistic regression and random forest both attained comparable performance measures. The accuracy, precision, and F1-scores of the Porter and Snowball stemmers were somewhat greater than those of the Lancaster stemmer. This is probably because the Lancaster stemmer was more aggressive and reduced words to extremely basic forms, losing crucial semantic information in the process. Compared to Random Forest, Logistic Regression has a slightly higher accuracy rate.

4.2. Semantic Preservation

The semantic preservation scores, measured through cosine similarity, revealed that the Porter and Snowball stemmers preserved the meaning of the text better than Lancaster. This was particularly important for sarcasm detection, where subtle word choices and phrasing play a crucial role in conveying sarcastic intent.

4.3. Computational Efficiency

In terms of computational efficiency, logistic regression significantly outperformed random forest in training time, completing in half of the time required by random forest. This makes logistic regression a more attractive option for real-time applications where speed is a critical factor.

4.4 Vocabulary Reduction

Vocabulary size was significantly reduced in all cases, with Lancaster showing the greatest reduction due to its aggressiveness. Snowball provided a balanced reduction, maintaining more meaningful stems than Lancaster while still achieving a lower vocabulary size than Porter.

5. Discussion

5.1. Impact of Stemming Techniques

The performance of sarcasm detection models was influenced by the choice of stemming technique. The ***Porter*** and ***Snowball*** stemmers performed similarly in terms of accuracy, while the ***Lancaster*** stemmer produced lower scores due to its aggressive nature. In applications like sarcasm detection, where nuances in word choice are critical, stemmers that preserve more semantic information, like Porter and Snowball, are preferable.

5.2. Model Comparison

Despite the theoretical advantages of random forest, such as its ability to capture complex, non-linear relationships in the data, the performance of logistic regression was comparable in terms of accuracy and other metrics. This suggests that the sarcastic instances in the dataset are linearly separable, allowing a simpler model like logistic regression to perform on par with random forest.

5.3. Trade-offs Between Models

While random forest offers greater flexibility and feature importance insights, its computational cost is significantly higher than logistic regression. For real-time sarcasm detection applications, the faster training time of logistic regression may outweigh the marginal performance gains offered by random forest.

6. Conclusion

This study compared the effectiveness of different stemming techniques in sarcasm detection using two machine learning models: logistic regression and random forest. Despite the differences in model complexity, the performance of both models was similar across all stemming techniques. The Porter and Snowball stemmers were found to preserve semantic meaning better than the Lancaster stemmer, contributing to higher sarcasm detection accuracy. While random forest performed slightly better in terms of accuracy, it required significantly more computational resources, making logistic regression a more practical choice for real-time applications.

Future work could explore more advanced preprocessing techniques, such as lemmatization or character-level embeddings, and evaluate the performance of deep learning models for sarcasm detection. Additionally, expanding the dataset to include more diverse sarcastic expressions could provide further insights into the generalizability of these findings.

REFERENCES

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

[2] Muralidaran, V., Palmer, G., Arman, L., O'Hare, K.E.Z.I.A.H., Knight, D. and Spasic, I., 2021. A practical implementation of a porter stemmer for Welsh. Language and Technology in Wales: Volume I, p.30.

[3] Jivani, A.G., 2011. A comparative study of stemming algorithms. Int. J. Comp. Tech. Appl, 2(6), pp.1930-1938.

[4] Lovins, J.B., 1968. Development of a stemming algorithm. Mech. Transl. Comput. Linguistics, 11(1-2), pp.22-31.

[5] Singh, J. and Gupta, V., 2017. A systematic review of text stemming techniques. Artificial Intelligence Review, 48, pp.157-217.

[6] https://www.kaggle.com/code/sqrl96/starter-news-headlines-dataset-for-e6eaa8df-5

[7] Abdullah Amer, A.Y. and Siddiqu, T., 2022. A novel algorithm for sarcasm detection using supervised machine learning approach. AIMS Electronics & Electrical Engineering, 6(4).