Parts of Speech (POS) Tagging - An Application of Natural Language Processing

Devashri Narendrabhai Chauhan
dchauhan1@stevens.edu
CWID: 20020187

Nikita Venkatdas Vanga
nvanga1@stevens.edu
CWID: 20016333

Vedant Sanjay Saraf
vsaraf1@stevens.edu
CWID: 20012288

I. Introduction

Words are the roots to form sentences in English language. A sentence in English is a combination of words in form of Parts of Speech such as Nouns, Pronouns, Verbs, Adjectives, Conjunctions, Prepositions, etc. A word can either be used as an adjective or a verb. Switching the part of speech for the words may have inappropriate meanings. It is of utmost importance to know in which context the word is being used. Some words can represent more than one part of speech at different times. For example, ”dogs” is usually thought of as a plural noun, it can also be used as a verb: ”The sailor dogs the hatch.” Thus, the main purpose of this paper is to propose a model that will efficiently train a Multi-Class Classifier that accurately predicts the part of speech for a given input word by training on features such as word length, prefixes-suffixes, punctuations, clauses, phrases and so on.

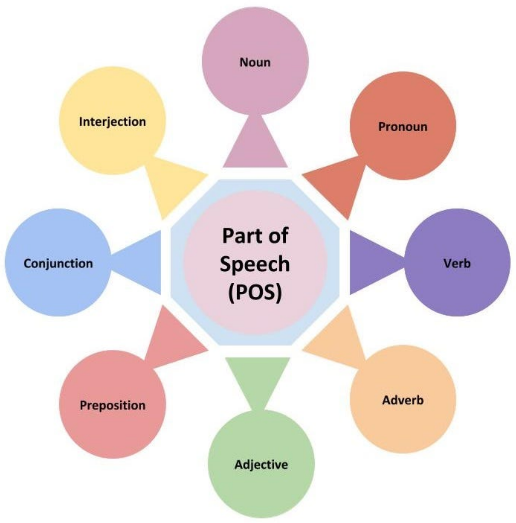


Fig. 1. Components of Parts of Speech (POS)

The upcoming sections will focus on a brief explanation about existing solutions to the above mentioned problem, their positive aspects and drawbacks, and an introduction and analysis of the dataset. Further sections will provide a detailed description of the Machine Learning Algorithms used for solving the problem with their implementation steps comprehensibly explained. The remaining part of the paper focuses on comparing the proposed models and their results taking into consideration certain factors, future study and conclusions.

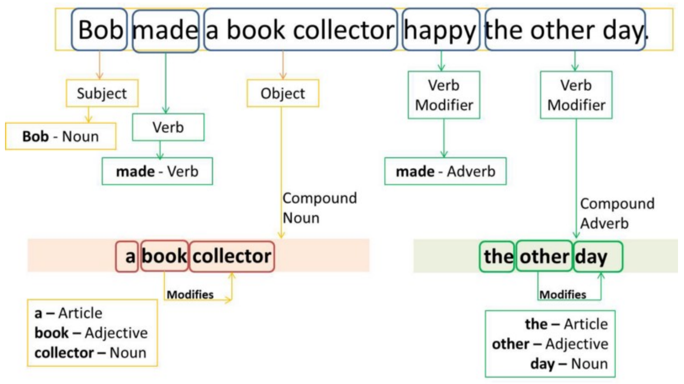


Fig. 2. POS Tagging

II. Related Work

POS Tagging is a common Natural Language Processing (NLP) task where words in text are assigned their corresponding part-of-speech labels. The existing solutions to this problem are:

1. **Rule-Based POS Tagging**: Rule-based methods use hand-crafted rules and linguistic patterns to assign POS tags to words based on their context and surrounding words. These methods are simple to implement but are unable to capture complex linguistic patterns.

2. **Statistical POS Tagging**: Statistical methods, such as Hidden Markov Models (HMMs) or Conditional Random Fields (CRFs), use probabilistic models to learn the likelihood of a word having a particular POS tag based on observed training data. They overcome the drawback of Rule-Based POS Tagging but demands annotated training data and performs poorly on rare or unseen data.

3. **Pre-Trained Language Models**: Pre-trained language models, such as BERT, GPT, and others, have shown significant success in POS tagging as they capture contextual information effectively. Fine-tuning on a specific POS tagging task can be performed. Eventhough, it leverages large-scale pre-training and state-of-art performance, it requires substantial computational resources which may not be suitable for every application.

4. **Machine Learning-Based POS Tagging**: Supervised machine learning algorithms, such as Support Vector Machines (SVM), Decision Trees, or more commonly, deep learning approaches like Recurrent Neural Networks (RNNs) and Long

Short-Term Memory networks (LSTMs), can be trained to predict POS tags based on contextual information. These algorithms can capture complex relationships but at the same there arises a huge requirement for high computing resources.

The existing solutions restrict to in terms of computational costs, resources, scaling, lack of data, etc. Thus, our solution has been proposed considering all the above mentioned factors. This paper will majorly focus on implementing Logistic Regression, Logistic Regression with PCA and Logistic Regression with Word Embedding techniques to solve the POS tagging problem.

III. Our Solution

The solution involves a general series of steps to be implemented. The steps are: Data Wrangling, Data Pre-Processing, Data Visualizations, Building Machine Learning Models, Training - Testing Models, Evaluating Models and Hyperparameter Tuning. These steps are divided into 3 major sections namely Description of Dataset, Machine Learning Algorithms and Implementation Details.

A. *Description of Dataset*

The dataset used in this paper has been taken from "*Kaggle - The world's largest repository for Data Scientists*". The dataset consists of 2617 rows and 8 features. The columns consists of different parts of speech such as Noun, Verb, Adjective, Adverb, Conjunction, Preposition, Interjection, Pronoun; filled with Boolean Values (TRUE or FALSE). i.e. They are categorical. The "Word" column contains the English words that are given as input to the classifer.

To gain more insights into data about its distribution, range of values, anomaly detection, unusual trends or patterns, missing values, etc. visualizations, feature engineering and analysis has been performed.

1. *Visualizations:* The visualizations representing certain features are:

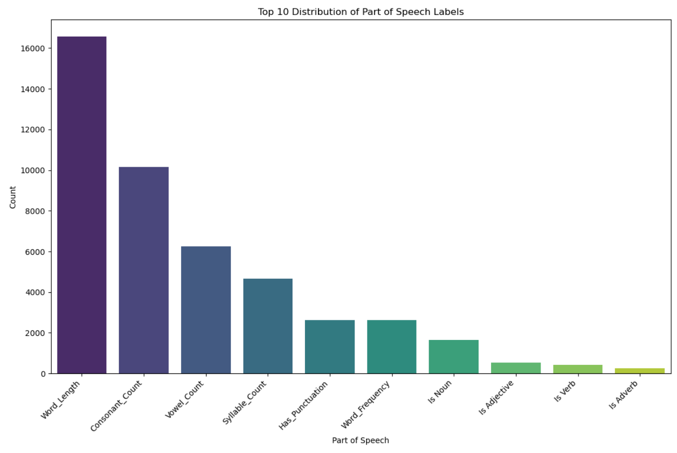


Fig. 3. Distribution of POS Labels

It can be clearly observed that the word length has highest count among other parts of speech. Whereas the words which are adverbs are very few in number. Also, there are approximately 3000 punctuated words in the dataset.

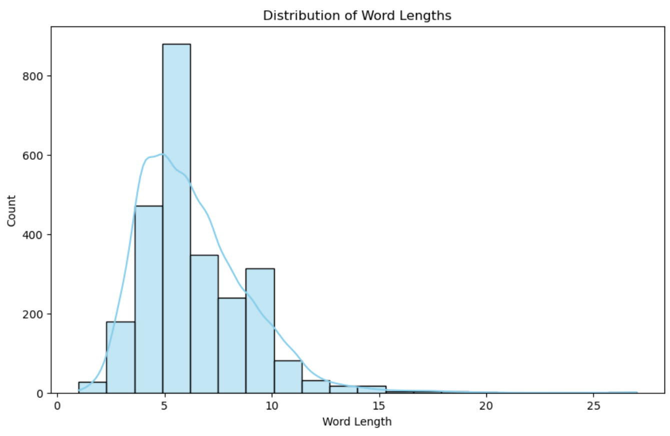


Fig. 4. Distribution of Word Lengths

It can be easily interpreted that maximum number of words have a length of about 6. There are very few words which are longer than 10 letters. Overall, the distribution of word lengths is not symmetric and skewed towards right.

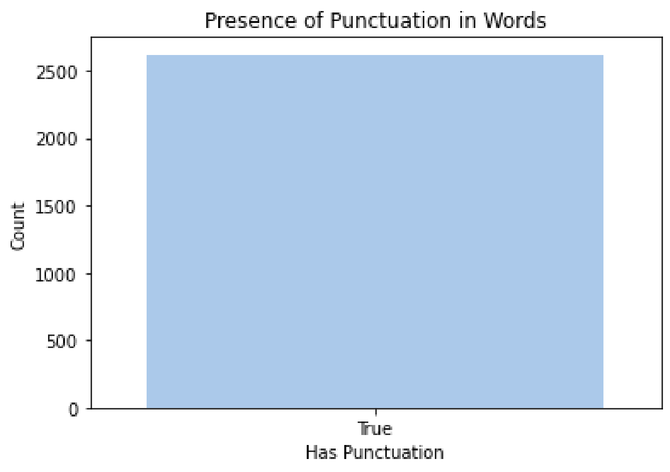


Fig. 5. Presence of Punctuation in Words

We find the count of words with or without punctuation. Here, the words in entire dataset has punctuation. There are no anomalies in the usage of punctuation.

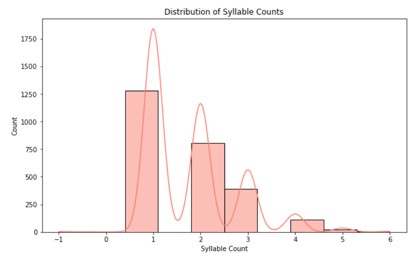
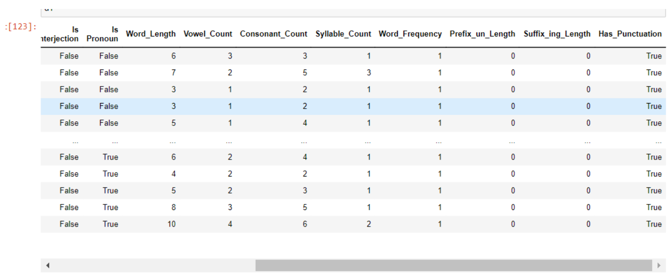


Fig. 6. Distribution of Syllable Counts

We examine patterns or outliers that might indicate discrepancies or errors in the syllable counting logic. It can be easily determined that count of syllables has a wavy decreasing pattern. i.e. a non-linear pattern having frequent ups and downs.

2. *Feature Engineering:* We modify some features by adding, removing or updating certain features.



:[123]:

Is nterjection	Is Pronoun	Word_Length	Vowel_Count	Consonant_Count	Syllable_Count	Word_Frequency	Prefix_un_Length	Suffix_ing_Length	Has_Punctuation
False	False	6	3	3	1	1	0	0	True
False	False	7	2	5	3	1	0	0	True
False	False	3	1	2	1	1	0	0	True
False	False	3	1	2	1	1	0	0	True
False	False	5	1	4	1	1	0	0	True
...	...	...	...	...	...	...	...	...	...
False	True	6	2	4	1	1	0	0	True
False	True	4	2	2	1	1	0	0	True
False	True	5	2	3	1	1	0	0	True
False	True	8	3	5	1	1	0	0	True
False	True	10	4	6	2	1	0	0	True

Fig. 7. Feature Engineering

We find the length of words, count of vowels and consonants, frequency of words, prefix-suffix lengths, etc.

3. *Analyzing and Handling Outliers:* We modify some features by adding, removing or updating certain features.

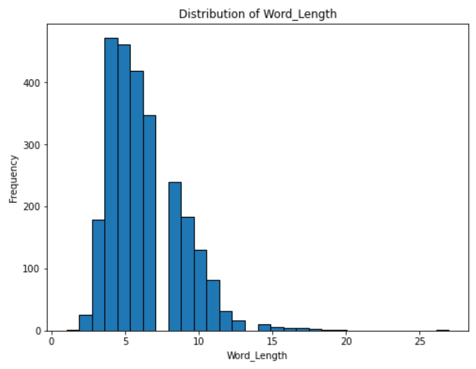


Fig. 8. Distribution of Word Length

The words with length between 2 and 12 are more frequent. However, there is one word having length greater than 25.

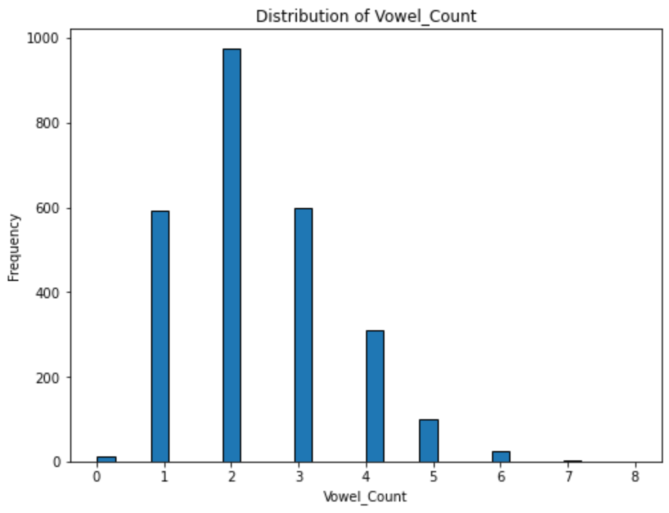


Fig. 9. Distribution of Vowel Count

We select 3 features: 'Word_length', 'Vowel_count' and 'Consonant_count' for analysis. By performing analysis we get outliers for Word_length and Vowel_count. We replace the value of outliers with the mean of individual features.

B. *Machine Learning Algorithms*

Further step in the series of steps of the proposed solution is implementing Machine Learning Algorithms. The justification as to why the proposed techniques are useful in solving the POS Tagging problem has been mentioned in a clear and precise manner.

$$\text{Class 1:} \quad p_1 = \frac{e^{\beta_1^T x_i}}{1 + \sum_{i=1}^{K-1} e^{\beta_i^T x_i}}$$

$$\text{Class 2:} \quad p_2 = \frac{e^{\beta_2^T x_i}}{1 + \sum_{i=1}^{K-1} e^{\beta_i^T x_i}}$$

$$\vdots$$

$$\text{Class K-1:} \quad p_{K-1} = \frac{e^{\beta_{K-1}^T x_i}}{1 + \sum_{i=1}^{K-1} e^{\beta_i^T x_i}}$$

$$\text{Class K:} \quad p_K = \frac{1}{1 + \sum_{i=1}^{K-1} e^{\beta_i^T x_i}}$$

Fig. 10. Multi-Class Logistic Regression

For a given input word, the Logistic Model predicts the class (i.e. POS) for that particular word. For example,

X: number of capital letters in target and surrounding words

Y: 1 if target is a part of a proper noun, 0 otherwise

1. **Logistic Regression**: For the given problem statement of predicting the part of speech (POS) for English words based on various features, using logistic regression can be justified for several reasons:

- *Multiclass Classification:* Logistic regression can be extended to handle multiclass classification problems through techniques such as one-vs-rest (OvR) or one-vs-one (OvO). In this case, where the goal is to predict the part of speech, which is likely to have multiple classes (nouns, verbs, adjectives, etc.), logistic regression can be adapted to handle this scenario effectively.
- *Interpretability:* Logistic regression provides a simple and interpretable model. The coefficients associated with each feature in logistic regression can be easily interpreted in terms of their impact on predicting a specific part of speech.
- *Efficiency with Linear Separation:* Logistic regression performs well when the decision boundaries between classes are approximately linear. Given that linguistic features (such as word length, prefixes, suffixes) are often linearly separable, logistic regression can efficiently capture these relationships.
- *Feature Importance:* Logistic regression allows the identification of important features. The coefficients associated with each feature indicate their contribution

to the prediction, helping to identify which linguistic features are more informative in determining the part of speech.

- *Efficiency for Small to Medium-Sized Datasets:* If the dataset is relatively small to medium-sized, logistic regression can be computationally efficient. It doesn't require as much computational power as more complex algorithms, making it suitable for datasets with a few thousand instances.
- *Regularization for Feature Selection:* Logistic regression supports regularization techniques (e.g., L1 regularization) that can be employed for feature selection. This can be beneficial in cases where there are many features, helping to focus on the most relevant ones.
- *Baseline Model for Comparison:* Logistic regression can serve as a baseline model for comparison with more complex algorithms. It provides a straightforward comparison point to evaluate the performance of more sophisticated models like decision trees, random forests, or neural networks.
- *Binary Relevance for Multilabel Classification:* If the problem involves predicting multiple parts of speech for a single word (multilabel classification), logistic regression can be used in a binary relevance approach where separate logistic regression models are trained for each class.

2. **Logistic Regression with Principal Component Analysis (PCA)**:

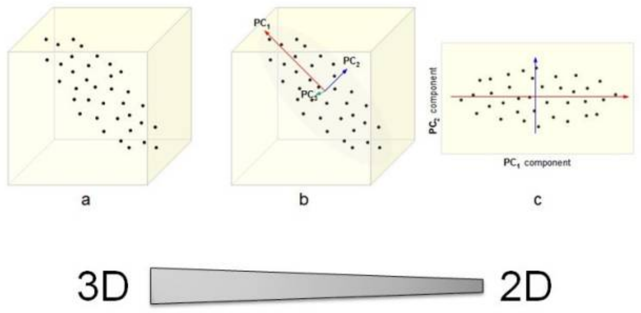


Fig. 11. Overview of Principal Component Analysis (PCA)

For the given problem statement of predicting the part of speech (POS) for English words based on various features, PCA is not directly used in solving the problem. But instead, it is largely employed for dimensionality reduction and feature extraction. As a result of this, it may lead to the increase in accuracy of the model considering fewer components. It can be justified for several reasons:

- *Dimensionality Reduction:* While solving POS Tagging problems, one major challenge that comes across is a huge dataset relying on a large number of features such as word embeddings, syntactic features, etc. PCA helps in reducing the number of features while retaining the most important information.
- *Feature Engineering:* PCA determines the features having largest variation among them and ultimately choosing the best features necessary for training the model.
- *Handling Multicollinearity:* Multicollinearity among features can be an issue in POS tagging models. PCA can help mitigate multicollinearity by transforming the original features into a set of linearly uncorrelated principal components. This can be particularly relevant when dealing with diverse linguistic features that may exhibit correlations.

3. **Logistic Regression with Word2Vec**:

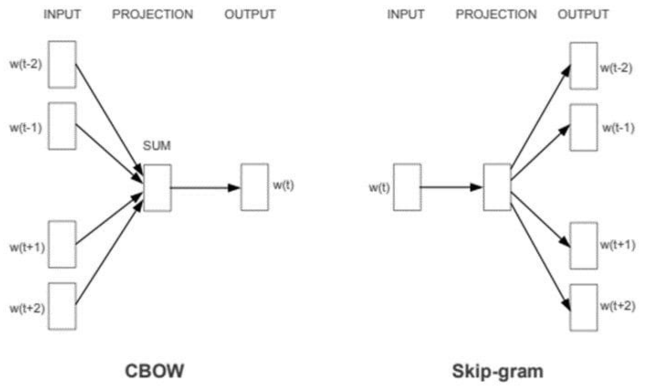


Fig. 12. Overview of Word2Vec

Word2vec is a group of related models that are used to produce word embeddings. These models are shallow, two-layer neural networks that are trained to reconstruct linguistic contexts of words. Word2Vec is not a singular algorithm, rather, it is a family of model architectures and optimizations that can be used to learn word embeddings from large datasets. It is basically an embedding technique that represents words in continuous vector spaces. It can iterate over a large corpus of text to learn associations or dependencies among words. It is widely used especially for text mining. The reasons for the same are:

- *Considering Contextual Information:* Word2Vec considers the context with respect to surrounding words during the training phase. It also captures meanings of words based on their usage in context.
- *Handling Ambiguity:* POS tagging often involves disambiguating between different possible parts of speech for a word (e.g., "lead" as a noun or verb). Word2Vec embeddings help capture the subtle differences in meaning and context, aiding in resolving such ambiguities.
- *Understanding and Representing Word Morphology:* Word2Vec embeddings encode information about word morphology and related linguistic features. This is valuable for POS tagging, where morphological cues often play a role in determining the grammatical category of a word.

C. Implementation Details

This section will describe the implementation steps in detail and results of the 3 algorithms discussed in above section. POS Tagging being a classification task, Confusion Matrix will majorly be used for model evaluation.

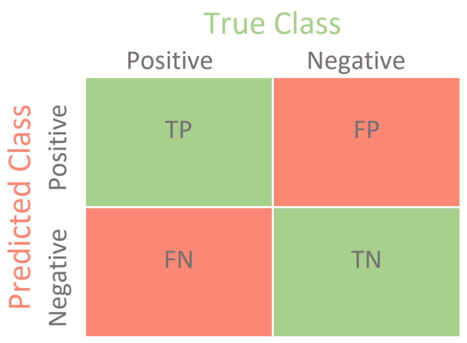


Fig. 13. Overview of Confusion Matrix

TP-Number positive examples classified accurately
FP-Number of actual negative examples classified as positive
FN-Number of actual positive examples classified as negative
TN-Number of negative examples classified accurately

Confusion Matrix is an evaluation metrics used to measure the performance of a classification model where the classes can be more than 2. It summarizes the performance of a machine learning model on a set of test data. The performance metrics used in this paper for model evaluation are mentioned below:
Accuracy = (TP + TN) / (TP + TN + FP + FN)
Precision = (TP) / (TP + FP)
Recall = (TP) / (TP + FN)
F1-Score = 2 * Precision * Recall / (Precision + Recall)

Logistic Regression Implementation Steps:

1. *Hyperparameter Definition* - The params dictionary specifies a grid of hyperparameter values to explore during the grid search. For example, it considers different values for regularization strength (C), penalty type (penalty), solver type (solver), maximum iterations (max_iter), and tolerance (tol).

2. *Logistic Regression Model Initialization* - Creates an instance of the Logistic Regression model. The logistic regression model is a popular choice for binary classification problems, and it's used as the base model for the grid search.

3. *MultiOutputClassifier Initialization* - Wraps the logistic regression model in a MultiOutputClassifier. MultiOutputClassifier is useful when dealing with multi-label classification problems where each sample can belong to more than one class.

4. *GridSearchCV Initialization* - Initializes a GridSearchCV object. multi_output_classifier is the base model, and it will be evaluated with different hyperparameter combinations. param_grid = params specifies the hyperparameter grid to search. scoring'accuracy' indicates that the evaluation metric is accuracy. cv5 sets up a 5fold cross-validation, which means the data is divided into 5 subsets, and the model is trained and evaluated 5 times, each time using a different subset as the test set.

5. *Grid Search Fit* - Fits the GridSearchCV object to the training data (X_train, y_train). Iterates through all possible combinations of hyperparameters and performs cross-validation to find the combination that maximizes accuracy.

6. *Best Model Extraction* - Extracts the best model found during the grid search. This model is the one with the hyperparameters that achieved the highest accuracy on the training data during cross-validation.

7. *Model Prediction on Test Data* - Uses the best model to make predictions on the test set (X_test). y_pred now contains the predicted labels for the test set.

8. *Evaluation Metrics Calculation* - Calculates various evaluation metrics (accuracy, precision, recall, F1 score) on the test set to assess the performance of the model. These metrics provide insights into different aspects of the model's performance, such as overall correctness (accuracy) and the balance between precision and recall.

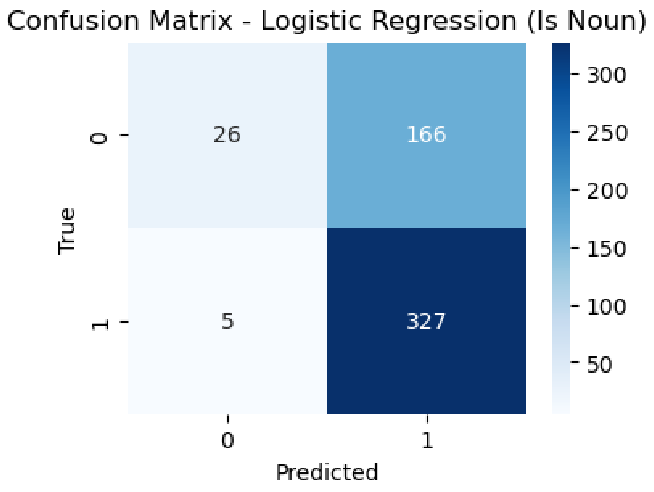


Fig. 14. Confusion Matrix - Logistic Regression (Is Noun)

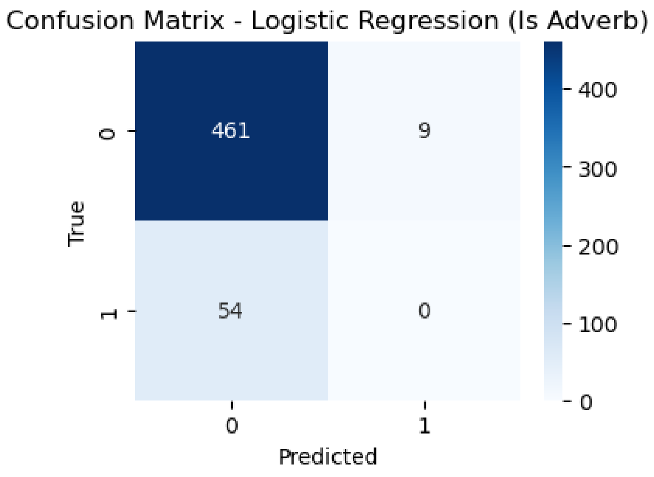


Fig. 15. Confusion Matrix - Logistic Regression (Is Adverb)

Preliminary Results obtained after implementing Logistic Regression using MultiOutput Classifier:

- Accuracy: **0.48091603053435117**
- Precision: **0.4073716102223907**
- Recall: **0.5263975155279503**
- F1 Score: **0.4366705036226193**

Logistic Regression with PCA Implementation Steps:

1. *Data Scaling with StandardScaler* - StandardScaler is used to standardize the features by removing the mean and scaling to unit variance. fit_transform is applied to the dataset X to scale it. This ensures that all features have a mean of 0 and a standard deviation of 1.

2. *Applying PCA for Dimensionality Reduction* - Creates an instance of the PCA (Principal Component Analysis) class with the parameter n_components=2, indicating that the goal is to reduce the dataset to two principal components. fit_transform is used to apply PCA to the scaled data (X_scaled).

3. *Creating a DataFrame for Principal Components* - Creates a new DataFrame (principal_df) to store the two principal components. The DataFrame has columns named 'PC1' and 'PC2', representing the first and second principal components, respectively.

4. *Printing Explained Variance Ratio* - Prints the explained variance ratio for each principal component. The explained variance ratio indicates the proportion of the dataset's variance that lies along each principal component. This information is useful for understanding how much information is retained by each principal component.

5. *Accessing Principal Components* - Prints the principal components themselves. Each row corresponds to a principal component, and each column corresponds to the original features of the dataset. The principal components are the directions in the feature space along which the data varies the most.

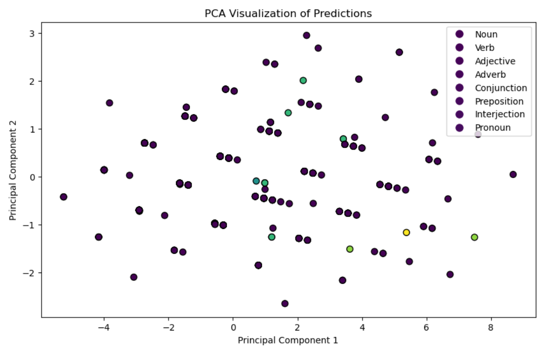


Fig. 16. Representation of Principal Components of Predictors

6. *Training and Evaluating Logistic Model with PCA* - Previously trained Logistic Regression Model is trained now on the 2-dimensional PCA model. Various evaluation metrics are used to evaluate the performance of the model.

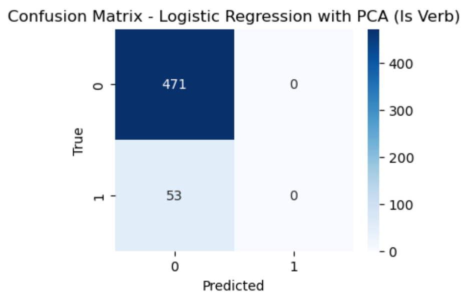


Fig. 17. Confusion Matrix - Logistic Regression with PCA (Is Verb)

Results obtained after implementing Logistic Regression with PCA Algorithm:

- Accuracy: **0.48854961832061067**
- Precision: **0.39026915113871635**
- Recall: **0.5201863354037267**
- F1 Score: **0.42421015837164905**

Logistic Regression with Word2Vec Implementation Steps:

1. *Preprocessing Data* - Tokenizing and preprocessing text data converts raw text into a format that can be easily fed into machine learning models. Here, NLTK's tokenize() function is used.

2. *Training Word2Vec Model* - Training a Word2Vec model captures the semantic relationships between words by representing them in a continuous vector space. Here, we use Continuous Bag of Words to train the Word2Vec Embedding.

3. *Feature Extraction* - Feature extraction using Word2Vec involves leveraging pre-trained Word2Vec models to convert words or phrases into continuous vector representations (embeddings).

4. *Feature Scaling* - Scales the extracted features so that they belong to a similar range of values for better training.

5. *Training and Evaluating Logistic Model with Word2Vec Embedding* - Previously trained Logistic Regression Model is trained now on the word embedded data and the features extracted from Word2Vec Embeddings. Various evaluation metrics are used to evaluate the performance of the model.

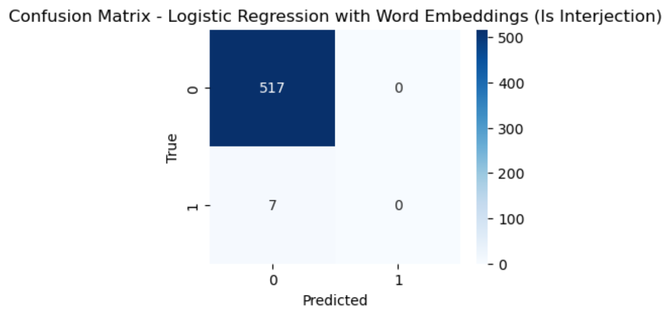


Fig. 18. Confusion Matrix - Logistic Regression with Word2Vec (Is Interjection)

Results obtained after implementing Logistic Regression with Word2Vec Embedding:

- Accuracy: **0.6335877862595419**
- Precision: **0.401433482897267**
- Recall: **0.6335877862595419**
- F1 Score: **0.4914746379396447**

IV. Comparison of Logistic Regression, Logistic Regression with PCA and Logistic Regression with Word2Vec

The comparison of the three models discussed in this paper can be made based on several factors such as Accuracy, Computational Resources, Efficiency, Interpretability, Information Loss, Task Complexity and Capturing Relationships. These factors can be justified as:

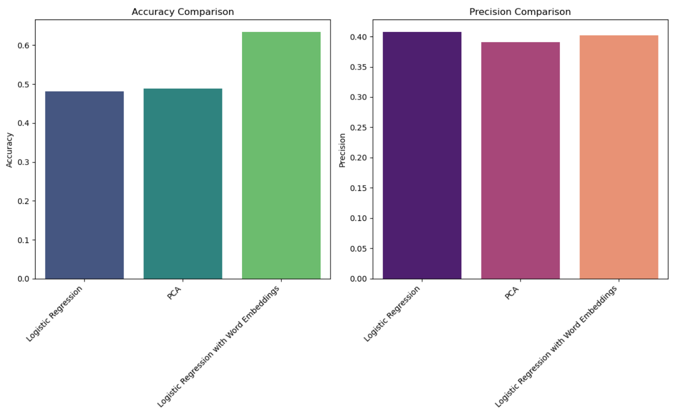


Fig. 19. Comparison of Accuracy and Precision of the three models

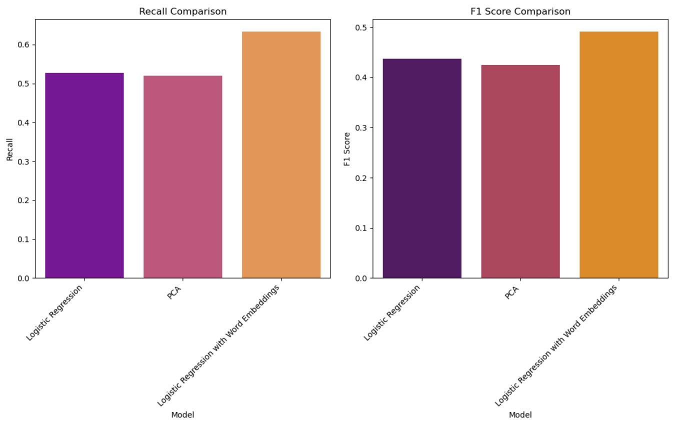


Fig. 20. Comparison of Recall and F1-Score of the three models

1. *Accuracy:* It can be clearly observed that Logistic Regression with Word2Vec outperforms the previously used 2 models. This can be justified for better semantic representation, improved feature representation and better understanding of contextual information provided by Logistic Regression with Word2Vec.

2. *Computational Resources:* PCA is less scalable for large datasets and hence demand high computing resources for computation. Considering Logistic Regression, it is scalable for large datasets but Word2Vec performs better when trained on large corpus.

3. *Efficiency:* From the observed results of the above mentioned 3 models, Word2Vec turns out to be the most efficient among others. The reason for this can be explained by its feature of handling non-linearity to a greater extent.

4. *Interpretability:* Logistic Regression Model is interpretable as the coefficients of the independent variables themselves show their individual contributions. Whereas, in Word2Vec each dimension in the embedding space can be associated with a semantic feature, providing insights into which aspects of the context are influencing the classification decisions. Hence, provides deep insights.

5. *Information Loss:* PCA performs dimensionality reduction by reducing the redundant features. This may lead to loss of some crucial information and the model may not be able to generalize in a better way. In contrast to this, Word2Vec performs effective dimensionality reduction retaining most meaningful information.

6. *Task Complexity:* Logistic Regression is well-suited for tasks with linear decision boundaries and where the relationship between features and the target variable is relatively simple. Logistic Regression with PCA is useful when dealing with high-dimensional datasets and multicollinearity among features. Whereas, Logistic Regression with Word2Vec is very useful especially for various NLP tasks.

7. *Capturing Relationships:* Logistic Regression assumes a linear relationship between the features and may not capture complex non-linear relationships. In contrast to this, Logistic Regression with PCA also assumes linear relationships and may not have a clear interpretation, making it challenging to understand the nature of the captured relationships. However, Word2Vec can easily capture non-linear relationships and can better understand the context in which the words are used.

V. Future Research Directions

This section provides the area which will require further improvement while using Word2Vec-based approach for POS Tagging.

1. *Efficient Word2Vec-based Model*: As previously mentioned, POS Tagging is very crucial while working with Natural Language Processing Algorithms having varied applications in Linguistic Analysis, Information Retrieval, Language Understanding and Machine Translation. Recent research works brings to notice that there lies a constraint in automatic tagging of ”Unknown” words for different languages. Thus, further research can be done in this area that will come up to build an efficient POS tagging model ultimately minimizing the number of False Positives.

2. *Moving towards using Complex Models*: Introducing complexity to our existing models involves considering huge dataset and using GPU-based high-performance computers, but they are computationally expensive. To reduce computational costs, we can migrate to cloud platforms for model training and evaluation.

VI. Conclusion

In this paper, the proposed solution to solve POS Tagging problem involves use of 3 algorithms: Simple Logistic Regression, Logistic Regression with PCA and Logistic Regression with Word2Vec. The main idea of the proposed approach is to predict the part of speech for a given English word. The same word can have more than one part of speech with different context in different sentences. Well, our proposed model with Word2Vec Embedding works out to be a good predictor for this problem at hand. However, further research work for improvement in efficiency can be done using another advanced techniques such as BERT, GloVe, etc.

VII. References

1. Part-of-Speech Tagging Article. Wikipedia

2. Chiche, A., Yitagesu, B. Part of speech tagging: a systematic review of deep learning and machine learning approaches. J Big Data 9, 10 (2022). https://doi.org/10.1186/s40537-022-00561-y

3. Logistic Regression and POS Tagging. Stony Brook University - Department of Computer Science

4. T. Pranckevičius and V. Marcinkevičius, "Application of Logistic Regression with part-of-the-speech tagging for multi-class text classification," 2016 IEEE 4th Workshop on Advances in Information, Electronic and Electrical Engineering (AIEEE), Vilnius, Lithuania, 2016, pp. 1-5, doi: 10.1109/AIEEE.2016.7821805.

5. Multi-Class Logistic Regression. Blog by Peter Oliver Caya

6. Fan JW, Prasad R, Yabut RM, Loomis RM, Zisook DS, Mattison JE, Huang Y. Part-of-speech tagging for clinical text: wall or bridge between institutions? AMIA Annu Symp Proc. 2011;2011:382-91. Epub 2011 Oct 22. PMID: 22195091; PMCID: PMC3243258.

7. D. Suleiman and A. A. Awajan, "Using Part of Speech Tagging for Improving Word2vec Model," 2019 2nd International Conference on new Trends in Computing Sciences (ICTCS), Amman, Jordan, 2019, pp. 1-7, doi: 10.1109/ICTCS.2019.8923081.

VIII. Abbreviations

BERT	Bidirectional Encoder Representations from Transformers
CRF	Conditional Random Fields
GPT	Generative Pre-trained Transformers
HMM	Hidden Markov Models
GloVe	Global Vectors for Word Representation
LSTMs	GLong Short-Term Memory Networks
NLP	Natural Language Processing
OvO	One-vs-One
OvR	One-vs-Rest
PCA	Principal Component Analysis
POS	Parts of Speech
RNN	Recurrent Neural Networks
SVM	Support Vector Machines