Understanding Linguistic Clues from Fake Reviews

Md Mahin
Id: 1900421
University of Houston

Abstract—**This project tries to analyze linguistic clues from fake review datasets. It explores two types of linguistic clues, general linguistic clues, found using n-gram of words and n-gram of pos tagging, and several linguistic clues from deceptive language clues. The performance of general linguistic clues such as n-gram of words and tags are measured using the prediction accuracy of three classifiers: Logistic Regression, Support Vector Machine and Random Forrest classifier. The impact of deceptive language clues on fake reviews is explored using the r-squared value of a linear regression algorithm. The performance is compared against a Deep Recurrent Convolutional Classifier. Initial analysis shows using linguistic clues directly does not outperforms deep learning model. The main contribution of the project is the direction a new researcher can take while exploring linguistic features for text classifications and some deceptive language specific features.**

Keywords: Deceptive Language, N gram, Deep Recurrent Convolutional Classifier, Logistic Regression, Linear Regression.

I. Introduction

Online reviews are a popular identification method for quality and services. In the modern world, it impacts the decision of customers hugely. As a result, it also becomes a very common practice to pay and buy positive reviews from people to deceive mass customers. As a result, people often mistake products with lower quality for high-quality products and be often deceived.

Deceptive languages, especially when they are created hastily, demonstrate linguistic properties. Deep Learning Architectures are already very good at distinguishing normal-level deceptive languages. However, deceivers also adapt to the improvement of the system and try to generate more complex and camouflaging reviews as possible.

Based on the inspiration that deceivers always left some linguistic clues in their languages, this project tries to examine how generic and deceptive language-specific linguistic clues help identify deceptive languages and compares that against the performance of state-of-the-art deep learning architecture[1].

This project tries to identify two categories of linguistic clues. First category is, automatically identifiable general linguistic clues using n-gram of words and n-gram of POS tags. The second category is deceptive language specific linguistic clues proposed by [2].

This project explores if applying linguistic clues directly can outperform deep learning models. Though initial result analysis shows, linguistic clue based models needs more feature optimization techniques to provide comparable results with deep learning models. Number of analysis shows the models are suffering from curse of dimentionality after the feature creation and methods are needed to combine different types of features. The main contribution of the project is the direction a new researcher in the NLP domain can take for text classification using Linguistic Features and some deceptive language specific features.

II. Related Works

Twenty-five linguistic cues from nine broad categories related to the deceptive language are identified by Addawood et al. [2]. The categories are Uncertainty, Non-immediacy, Specificity, Information Complexity, Information Quantity, Persuasion, and Morality. The authors used these linguistic clues to detect deceptive political tweets.

Faranak et al. [3] have tried to find linguistic clues from a fake review dataset. The authors mainly focused on fifteen transcribed speech features defined by [2]. Ahmed et al. [4] have explored n-gram-based features to classify fake news using six different classifiers. N-gram-based features are also used in other classification tasks such as sentiment classification [5], malware identification [6], web page classification [7], and identifying duplicate bug reports [8].

III. Methodology

A. N-gram of Words Detection

First n-grams of words from each review are generated, and then all n-grams are treated as token input for the TFIDF. For the experiment purpose, a classifier is trained with (i to n) gram features and also with just n-gram feature. For example, for 3 gram, a classifier is trained with several versions of features such as 1 gram, 2 gram, 3 gram, combined 1,2 gram, combined 1,2,3 gram, and combined 2,3 gram. As a result, six different feature groups are generated. For n-gram, there is total n! Feature groups found. Three classifier models: Logistic Regression, Support Vector Machine and Random Forrest is trained for all n! groups, and accuracy is measured.

B. N-gram of POS Detection

First the POS tags of each word are generated, and n-grams are generated for the POS tags only. Similar to the previous section, for n-gram of tags, total n! numbers of features are generated. Three classifier models: Logistic Regression, Support Vector Machine and Random Forrest is trained for all n! groups, and accuracy is measured.

Positive Word Density: Positive word density represents the emotion of specificity related to deceptive language. Positive word density is found using MPQA. The density for each review is calculated using the following method:

$$Positive\ Word\ Density = \frac{NPW}{(NPW + NNW)} \quad (1)$$

where,
$NPW = Number\ of\ PositiveWords$
$NNW = Number\ of\ Negative\ Words$

Modifier Word Density: Modifiers represent the uncertainty of deceptive language. The density for each review is calculated using the following method:

$$Modifier\ Word\ Density = \frac{NMR}{TWR} \quad (2)$$

where,
$NMR = Number\ of\ Modifiers\ in\ the\ Review$
$TWR = Total\ Words\ in\ the\ Review$

Strong Subjectivity Word Density: Subjectivity determines the uncertainty of deceptive language. The strong subjectivity density for each review is calculated using the following method:

$$Strong\ Subjectivity\ Density = \frac{NSSWR}{NSSWR + NWSWR} \quad (3)$$

where,
$NSSWR = Number\ of\ Strong\ Subjective\ Words$
$NWSWR = Number\ of\ Weak\ Subjective\ Words$

Quotation Frequency: The presence of quotations also helps to identify the uncertainty of deceptive language. Quotation frequency is determined by the presence of the quotation ‘ ”” ’ symbol in the review. The number of quotation symbols is divided by two to find the frequency.

$$Quotation\ Frequency = NQR \quad (4)$$

where,
$NQR = Number\ of\ Quotation\ in\ the\ Review$

Question Frequency: The presence of questions is another form of uncertainty identifier of deceptive language. Question frequency is determined by the presence of the question ‘ ? ’ symbol in the review. The number of quotation symbols in each review represents the frequency.

$$Question\ Frequency = NQoR \quad (5)$$

where,
$NQoR = Number\ of\ Questions\ in\ the\ Review$

Number Density: The use of numbers determines the specificity of deceptive language. Number density identifies the percentage of words in a review that contains numbers. Number density is calculated using the following method:

$$Number\ Density = \frac{TNR}{TWR} \quad (6)$$

where,
$TNR = Total\ Numberss\ in\ the\ Review$
$TWR = Total\ Words\ in\ the\ Review$

Self-Reference Density: Self-reference identifies the non-immediacy nature of deceptive language. So, self-reference density is counted using the number of first-person pronouns present in a review. The density is defined as:

$$Self\ Reference\ Density = \frac{TNFP}{TWR} \quad (7)$$

where,
$TNFP = Total\ Numbers\ of\ First\ Person\ in\ the\ Review$
$TWR = Total\ Words\ in\ the\ Review$

Group Reference Density: Group reference identifies the non-immediacy nature of deceptive language. So group reference density is counted using the number of third-person pronouns present in a review. The density is defined as:

$$Group\ Reference\ Density = \frac{TNTP}{TWR} \quad (8)$$

where,
$TNTP = Total\ Numbers\ of\ Third\ Person\ in\ the\ Review$
$TWR = Total\ Words\ in\ the\ Review$

Article Density: The presence of articles also identifies the non-immediacy nature of deceptive language. So Article Density is counted using the number of the article present in a review. The density is defined as:

$$Article\ Density = \frac{NAR}{TWR} \quad (9)$$

where,
$NTP = Numbers\ of\ Articles\ in\ the\ Review$
$TWR = Total\ Words\ in\ the\ Review$

C. Deceptive Linguistic Features Generation

To analyze deceptive linguistic features, nine features are extracted manually. The features are inspired by the paper Addawood et. al. [2]. The features are Positive Word Density, Modifier Word Density, Strong Subjectivity Word Density, Quotation Frequency, Question Frequency, Number Density, Self-Reference Density, Group Reference Density, Article Density.

D. Deceptive Linguistic Features Evaluation

To evaluate deceptive linguistic features, the linear regression method is used. Ten linear regression models are fitted using individual features, and all of them combined and obtained r-squared value is used to interpret how much information the features provide explaining two different classes.

E. *Recurrent Convolutional Neural Network Architecture*

A state of art deep learning model, Recurrent Convolutional Neural Network Architecture, is used to evaluate the performance of linguistic clue-based classifiers. The main benefit of deep learning models are, feature creation are automatic within the models and thus it spares the researchers from the hectic process of creating features manually. The architecture of the network is following:

- Word embedding layer: 64-dimensional embedding vector.
- Embedding Matrix: 512 x 64 for each review.
- Convolutional layers: 3
- Convolutional filters: 128
- Kernel size: 5, followed by max-pool layer
- LSTM layers: 2, 128 neurons + 10% dropout
- Fully-connected layer: sigmoid activation

IV. Experiment Settings and Result Analysis

A. *Datasets Description*

Two datasets are used for the analysis. First, a computer-generated fake review dataset by Salminen et al. [9], which contains a total of 40432 reviews, among which 20216 are real and 20216 are fake. The reviews contain product categories, a rating score between 1 to 5, and labels if it is real or computer-generated. The second dataset is 110 amazon review datasets with 55 real and 55 fake[10]. The second dataset is used to create the models, and evaluation is done by combining the two datasets.

B. *Preprocessing*

Following preprocessing steps are followed for the general linguistic clues and classification using a Recurrent Convolutional Neural Network:

- Lower case conversion using Python String
- Number conversion into strings using Inflect
- Removal of punctuations, white spaces, and stop words removal using NLTK
- Named entity removal using spacy
- Stemming and Lemmatization using NLTK
- POS tagging using NLTK

For deceptive linguistic clues, only lower case conversion is applied, as many clues depend on unprocessed data.

C. *Word and Document Level Clustering*

To observe the word clusters within two types of reviews word level clustering is done and visualized using TSNE and seaborn. To do the clustering, first words are converted into word embeddings using word2vec.

Figure 1 and figure 2 shows the word level clustering for real and fake reviews. From the word level clustering, it can be observed fake reviews has way fewer linguistic variations than fake reviews.

Figure 3 and figure 4 shows the document level clustering for real and fake reviews. Though the differences are not easily visible, from the document level clustering, it can be observed

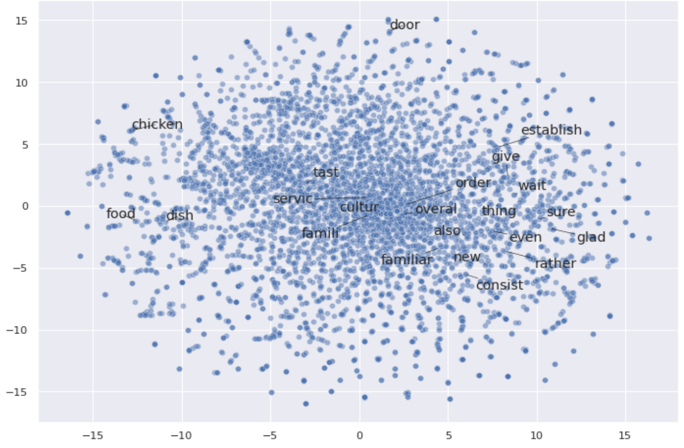


Fig. 1: Word Level Clustering for Fake Reviews

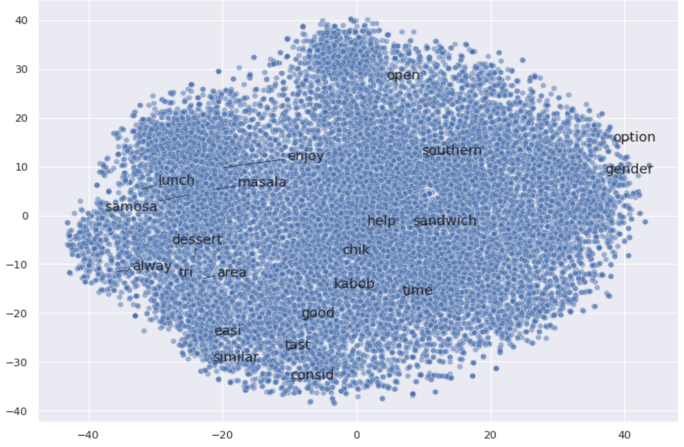


Fig. 2: Word Level Clustering for Real Reviews

fake reviews has few high dense zones where real reviews are more spread.

D. *Topic Modeling*

To see how topics are distribute from two types of reviews topic modeling for fake and real reviews are done using pyLDAvis. pyLDAvis uses Latent Dirichlet Allocation to do topic modeling.

From figure 5 and figure 6, it can be seen that the topics and more equally distributed, and frequent words for different types of topics are also visible.

E. *General Linguistic Model: N-gram of Words*

To observe the performance of the N-gram of words, three classifiers Logistic Regression, Support Vector Machine(SVM), and Random Forrest, are trained for different combinations of N-gram of words. Figure 7 and figure 8 shows the accuracy results where figure 7 are trained with named entities in the reviews and figure 8 are trained without named entities in the reviews. In these figures, each n-gram is represented by (i,j), where j goes till the nth gram. For example, (1,5) contains all the features from 1-gram, 2-gram,

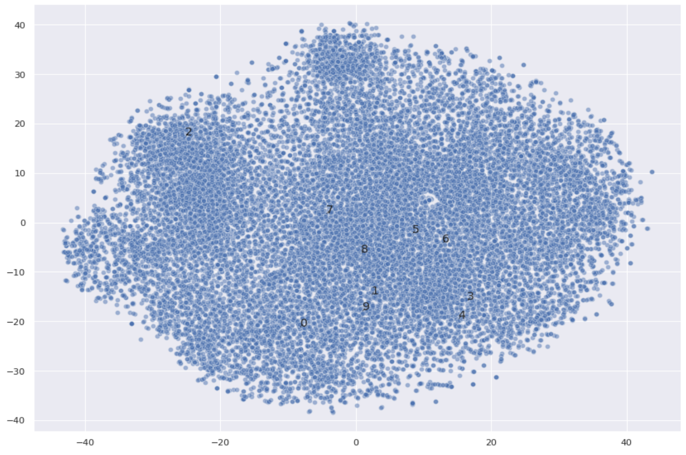


Fig. 3: Document Level Clustering for Fake Reviews

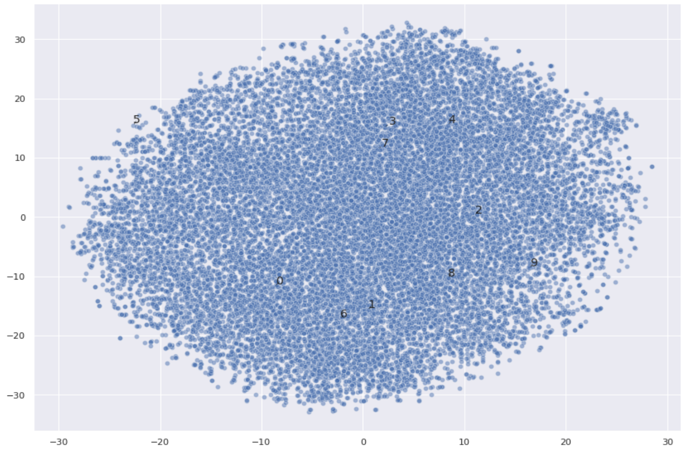
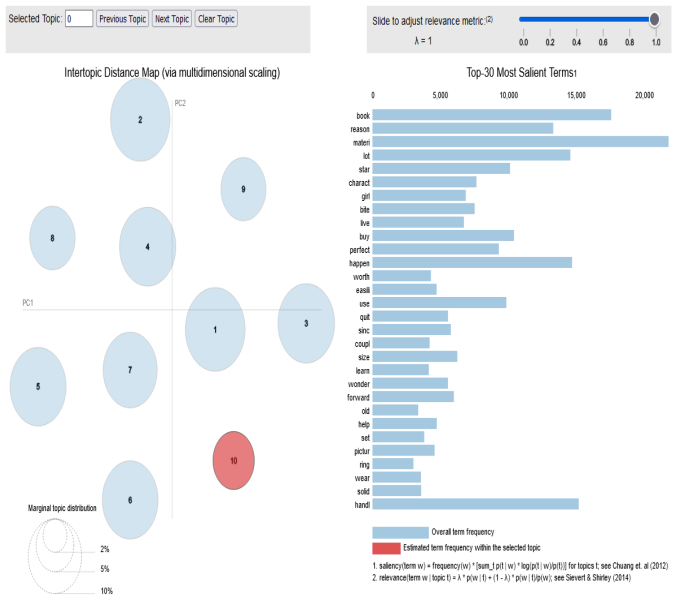
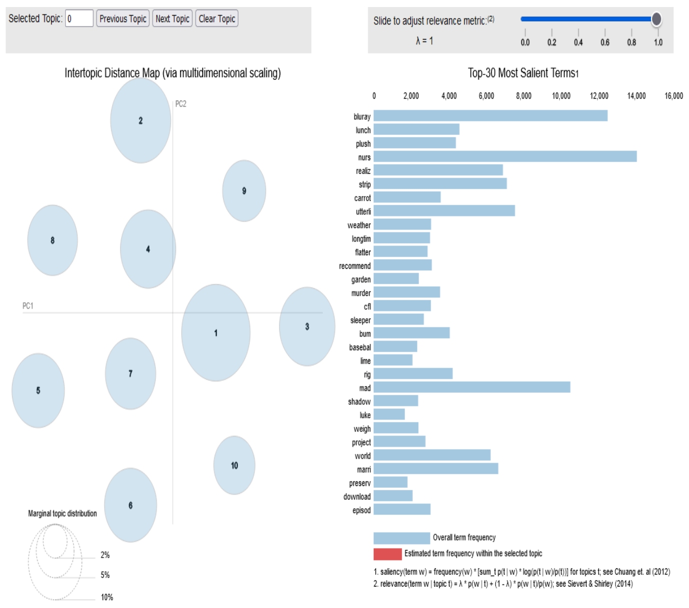


Fig. 4: Document Level Clustering for Real Reviews



Selected Topic: 0
Previous Topic
Next Topic
Clear Topic
Slide to adjust relevance metric:(2)
λ = 1
Intertopic Distance Map (via multidimensional scaling)
PC2
PC1
Marginal topic distribution
2%
5%
10%
Top-30 Most Salient Terms1
book
reason
materi
lot
star
charact
girl
bite
live
buy
perfect
happen
worth
easili
use
quit
sinc
coupl
size
learn
wonder
forward
old
help
set
pictur
ring
wear
solid
handl
Overall term frequency
Estimated term frequency within the selected topic
1. saliency(term w) = frequency(w) * [sum_t p(t | w) * log(p(t | w)/p(t))] for topics t; see Chuang et. al (2012)
2. relevance(term w | topic t) = λ * p(w | t) + (1 - λ) * p(w | t)/p(w); see Sievert & Shirley (2014)

Fig. 5: Topic Modeling for Fake Reviews



Selected Topic: 0
Previous Topic
Next Topic
Clear Topic
Slide to adjust relevance metric:(2)
λ = 1
Intertopic Distance Map (via multidimensional scaling)
PC2
PC1
Marginal topic distribution
2%
5%
10%
Top-30 Most Salient Terms1
bluray
lunch
plush
nurs
realiz
strip
carrot
utterli
weather
longtim
flatter
recommend
garden
murder
cfl
sleeper
bum
basebal
lime
rig
mad
shadow
luke
weigh
project
world
marri
preserv
download
episod
Overall term frequency
Estimated term frequency within the selected topic
1. saliency(term w) = frequency(w) * [sum_t p(t | w) * log(p(t | w)/p(t))] for topics t; see Chuang et. al (2012)
2. relevance(term w | topic t) = λ * p(w | t) + (1 - λ) * p(w | t)/p(w); see Sievert & Shirley (2014)

Fig. 6: Topic Modeling for Real Reviews

3-gram, 4-gram, and 5-gram, whereas (5,5) gram contains only 5-gram. All these grams are treated as a single token, and relative strength is calculated using TFIDF for any gram. From the result analysis, it is visible that the SVM gets the best results, but it is still around 50%. Also, named entity removal did not improve performance significantly.

F. General Linguistic Model: N-gram of PoS Tags

To observe the performance of the N-gram of PoS tags, the same three classifiers Logistic Regression, Support Vector Machine(SVM), and Random Forrest, are trained for different combinations of N-gram of PoS tags. Figure 9 and figure 10 shows the accuracy results where figure 9 are trained with named entities in the reviews and figure 10 are trained without named entities in the reviews. Here Random Forrest performed better, but accuracy dropped to 40%. Named entity removal did not significantly impact the accuracies here either.

G. Feature Importance

To observe the feature importance of N-gram of words for 1 to 9 grams, feature importance from the classifiers is extracted figure 11. A combination of 1 to 9 grams of reviews generated hundreds of thousands of features. As a result, feature importance is not observable. However, this gives insight that the classifiers are suffering from the curse of dimensionality due to the very high number of features.

H. Performance of Deceptive Linguistic Clues

To observe the performance of deceptive linguistic clues, a linear regression method is implemented for all deceptive features individually and combined. The r-square results are shown, as the r-square implies how many samples are explained by the features. Figure 12 and figure 13 shows the r-square results where figure 12 are trained with named entities

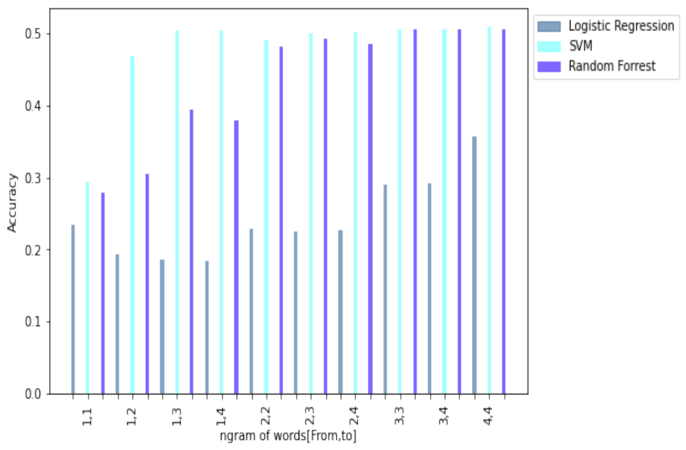


Fig. 7: Classification using N-gram of Words (With Named Entities)

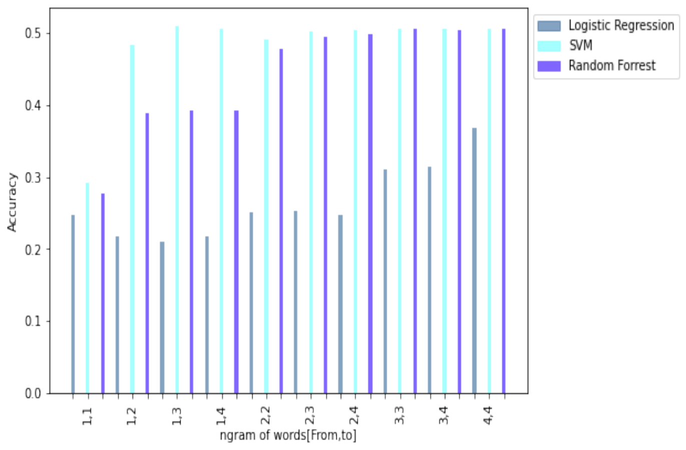


Fig. 8: Classification using N-gram of Words (Without Named Entities)

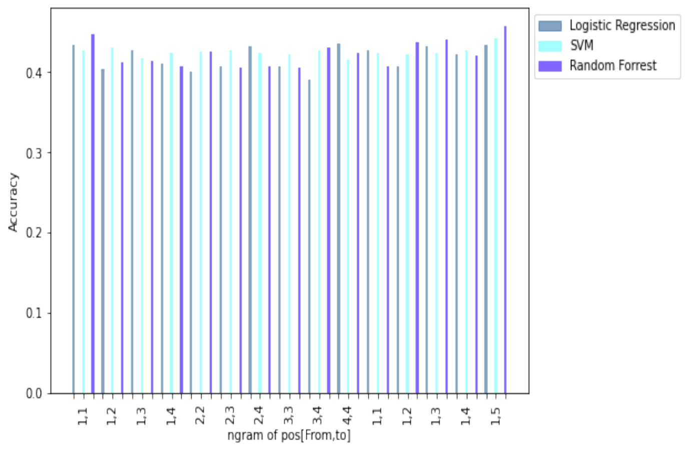


Fig. 9: Classification using N-gram of PoS Tags (With Named Entities)

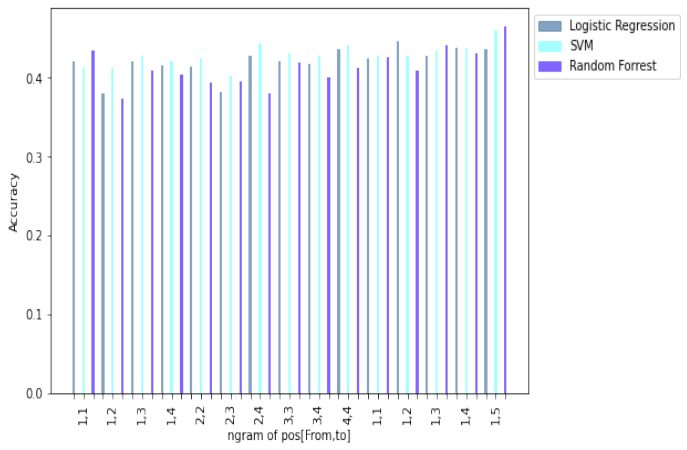


Fig. 10: Classification using N-gram of PoS Tags (Without Named Entities)

and figure 13 are trained without named entities. From the result, it can be seen, that named entity removal does not have much impact. Also, article density has the highest information among the features, but it explains only 15% of the samples. All features combined explain 20% of the samples.

I. Performance of Recurrent Convolutional Neural Network

The Recurrent Convolutional Neural Network is trained for two epochs as after two epochs the model starts overfitting and the following results are obtained:

- Accuracy: 94%
- Precision: 0.497
- Recall: 1
- F1-score: 0.664

V. Conclusions

From the analysis it can be seen Recurrent Convolutional Neural Network outperform any simple linguistic feature based models. The huge amount features may causes the curse of dimentionality problem for the classifiers. Number of techniques such as dimentionality reduction on the generated

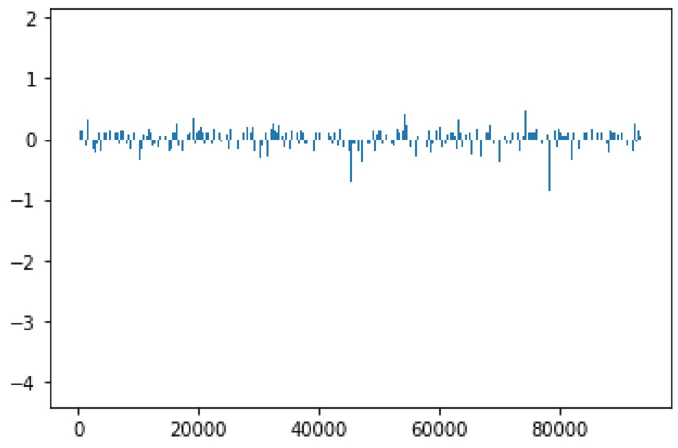


Fig. 11: Feature Importance for N-gram of Words (1-9 gram)

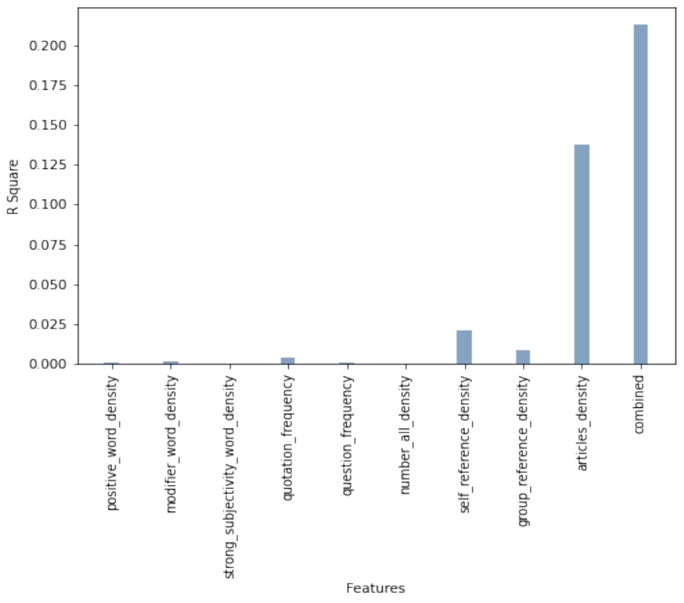


Fig. 12: R Square of Deceptive Linguistic Clues (With Named Entities)

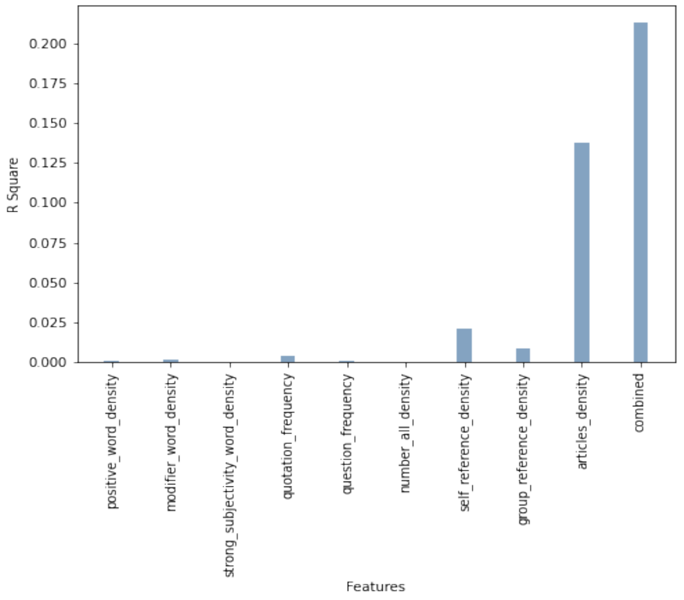


Fig. 13: R Square of Deceptive Linguistic Clues (Without Named Entities)

features might have improved the results. Also, combination of features could have been explored within the project. Another future direction is margin linguistic features with deep learning models and extending the model for more than two classes and observing the impacts. Due to time constraints all types of analysis could not be performed. However the project gives an insight about the problems researches may face while utilizing linguistic features for text classification.

References

[1] J. Y. Lee and F. Dernoncourt, "Sequential short-text classification with recurrent and convolutional neural networks," in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. San Diego, California: Association for Computational Linguistics, Jun. 2016, pp. 515–520. [Online]. Available: https://aclanthology.org/N16-1062

[2] A. Addawood, A. Badawy, K. Lerman, and E. Ferrara, "Linguistic cues to deception: Identifying political trolls on social media," in *Proceedings of the international AAAI conference on web and social media*, vol. 13, 2019, pp. 15–25.

[3] E. Y. Wang, L. H. N. Fong, and R. Law, "Detecting fake hospitality reviews through the interplay of emotional cues, cognitive cues and review valence," *International Journal of Contemporary Hospitality Management*, 2021.

[4] H. Ahmed, I. Traore, and S. Saad, "Detection of online fake news using n-gram analysis and machine learning techniques," in *International conference on intelligent, secure, and dependable systems in distributed and cloud environments*. Springer, 2017, pp. 127–138.

[5] A. Tripathy, A. Agrawal, and S. K. Rath, "Classification of sentiment reviews using n-gram machine learning approach," *Expert Systems with Applications*, vol. 57, pp. 117–126, 2016.

[6] M. Hassen, M. M. Carvalho, and P. K. Chan, "Malware classification using static analysis based features," in *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE, 2017, pp. 1–7.

[7] R. Rajalakshmi and C. Aravindan, "Web page classification using n-gram based url features," in *2013 fifth international conference on advanced computing (ICoAC)*. IEEE, 2013, pp. 15–21.

[8] A. Sureka and P. Jalote, "Detecting duplicate bug report using character n-gram-based features," in *2010 Asia Pacific Software Engineering Conference*. IEEE, 2010, pp. 366–374.

[9] J. Salminen, C. Kandpal, A. M. Kamel, S.-g. Jung, and B. J. Jansen, "Creating and detecting fake reviews of online products," *Journal of Retailing and Consumer Services*, vol. 64, p. 102771, 2022.

[10] E.-P. Lim, V.-A. Nguyen, N. Jindal, B. Liu, and H. W. Lauw, "Detecting product review spammers using rating behaviors," in *Proceedings of the 19th ACM international conference on Information and knowledge management*, 2010, pp. 939–948.