UNIVERSITY OF PASSAU

FACULTY OF COMPUTER SCIENCE AND MATHEMATICS

CHAIR FOR COMPUTATIONAL RHETORIC AND NATURAL LANGUAGE PROCESSING

Master Thesis in Informatics

Exploring Constructiveness in Online News Comments

submitted by

Ajay Kesarwani

1. Examiner:	Prof. Dr. Annette Hautli-Janisz
2. Examiner:	Prof. Dr. Florian Lemmerich
Date:	September 13, 2024

Contents

Abstract

The rise of online news platforms has significantly increased the volume of user-generated content, especially comments. While these comments can enrich the discourse by providing diverse perspectives and fostering community engagement, they can contribute to toxic and non-constructive conversations. This study investigates the linguistic, emotional, and psychological features that contribute to or detract from the constructive nature of online comments.

Using C3 (Constructive Comments Corpus) and YNACC (Yahoo News Annotated Comments Corpus) datasets, this study conducts an in-depth analysis to identify key features associated with constructive and non-constructive comments. The study evaluates the performance of various Machine Learning (ML) and Deep Learning (DL) models in classifying the constructiveness of comments, highlighting the impact of different datasets on model efficacy.

Key findings reveal that stylistic features like punctuation types, the presence of stopwords, and complexity features like higher word count and measures of textual lexical diversity (MTLD), are significantly associated with constructive comments. Positive emotions like joy and trust are associated with constructiveness, while negative emotions, like anger and disgust, correlate to non-constructive comments. Among all evaluated models, the Random Forest (RF) model consistently excelled across various feature types and datasets, highlighting its reliability. The Gated Recurrent Unit (GRU) model performed effectively with psychological and emotional features. Error analysis highlighted that models struggled most with abstract features like psychological and emotional features, which led to higher error rates. This highlights the need for more sophisticated feature engineering and model optimization.

This study concludes that structured and formal language combined with positive emotional expressions is essential for fostering constructive online discourse. It highlights the importance of selecting models based on specific dataset characteristics. Future research should explore diverse datasets and feature interactions to improve our understanding of constructive communication dynamics. These findings can guide the development of more effective content moderation strategies, contributing to healthier and more productive online communities.

Keywords: Constructive Comments, content moderation, machine learning, deep learning, linguistic features, emotional features, psychological features, C3 Dataset, YNACC Dataset

Acknowledgments

I would like to express my sincere gratitude to all those who have contributed to the completion of this master's thesis.

First and foremost, I want to express my heartfelt gratitude to Prof. Dr. Annette Hautli-Janisz for her time and effort, invaluable guidance, and insightful discussions throughout the research process. Her expertise and encouragement have been instrumental in shaping this work.

I would like to extend my sincere thanks to Prof. Dr. Florian Lemmerich for his invaluable support in my academic journey, His teaching has significantly enriched my work on this thesis.

My heartfelt appreciation goes to my family for their unwavering love, encouragement, and understanding during this academic journey. Their support has been my pillar of strength and determination.

I am also grateful to my friends and colleagues for their encouragement, helpful discussions, and moral support throughout this endeavour.

Lastly, I am grateful to all those who have contributed to this thesis. Your support and belief in my abilities have been crucial in completing this work.

Thank you all. . . .

List of Figures

List of Tables

List of Abbreviations

ML	Machine Learning
DL	Deep Learning
LR	Logistic Regression
RF	Random Forest
SVM	Support Vector Machine
KNN	K-Nearest Neighbors
LSTM	Long Short-Term Memory
BiLSTM	Bidirectional LSTM
CNN	Convolutional Neural Network
GRU	Gated Recurrent Units
C3	Constructive Comments Corpus
YNACC	Yahoo News Annotated Comments Corpus
POS	Parts-of-Speech
NLTK	Natural Language Toolkit
NLP	Natural Language Processing
AI	Artificial intelligence
HTML	HyperText Markup Language
TF-IDF	Term Frequency-Inverse Document Frequency
NER	Named Entity Recognition
RBF	Radial basis function
TTR	Type-Token Ratio
MTLD	Measure of Textual Lexical Diversity
FP	False Positive
FN	False Negative

1 Introduction

1.1 Introduction and Motivation

Today's era is a digital world where everyone is connected through social media platforms and stays updated with the latest information about their friends, family, and loved ones. In the past, people access information from print media, primarily from newspapers. Now, numerous online news platforms are available for example Yahoo News, BBC News, etc.

The best part of this social media platform is that people can share their opinions in the comment sections. Thousands of readers post millions of comments every day and express their views. These comments provide feedback to editors and insights into user's interests. They also encourage the editors to follow up with them to improve future articles and be reachable to a wider audience. Moreover, these platforms also facilitate interaction among users and provide productive conversations, this way, these users get to know the different perspectives and learn from other users while simultaneously expressing their opinions.

However, we also observe that sometimes users use abusive language, post toxic comments, and start fighting among themselves, creating a toxic environment in the comment section of these platforms. Therefore, handling these comments, maintaining a respectful atmosphere, and providing a positive and productive learning environment are crucial. To tackle these challenges, many newspaper platforms employ moderators to review and manage these toxic comments by removing, hiding, or some other means that violate community standards. For example, since 2007, a notable newspaper *The New York Times* has had a team of full-time moderators to review each comment posted on their website. [Eti17].

These news platforms receive large amounts of comments every hour from the users, therefore to handle these comments manually in real-time, even for a moderator team is significantly challenging. To address this problem, many platforms have adopted automated moderation software [Par+16]. This automated moderation software filters and organizes comments based on keywords and prevents posting toxic comments if users use such keywords in their posts.[See+19]. Software moderators have become common for news outlets and other social media platforms. This software provides several benefits such as speed and cost-effectiveness, but they also have limitations. For example, these software moderators struggle to distinguish between constructive and non-constructive comments.

Knowing insights into a user's comment and determining whether it provides meaningful and positive content and adds value to the other users is significantly challenging. It is difficult to find the pattern in such kinds of comments because, even utilizing the same bunch of words, but in different combinations, can completely change the meaning and intention of the entire comment. For instance, 'The project progress is not bad' can be interpreted positively as 'The project progress is good enough' and negatively as 'The project progress is unsatisfactory,' depending on the context and tone. One more such example could be "I saw the man with the telescope". Here we could have two possible senses from the same sentence, one can be interpreted as 'I saw the man who is holding the telescope' and another can be interpreted as 'I saw the man with the help of the telescope'.

Similarly, identifying the constructiveness of a comment is a complex and subjective task. It is challenging to pinpoint exactly which factors influence the constructiveness of comments or which properties of a comment contribute to its constructiveness. This research aims to explore and examine the pattern that impacts the constructiveness of comments and to understand and enhance the quality of discourse in digital news environments. Non-constructive comments not only degrade the quality of conversation but can also lead to misinformation, harassment, and a general decline in the credibility of online platforms. Addressing these issues is crucial for maintaining the integrity and value of online news discussions.

In this paper, we have leveraged linguistic features for detecting fake news as detailed in the study by Aich et al. (2022) [ABP22] and adapted them to recognize the constructiveness of online comments. The motivation behind adopting this paper is that fake news and constructive comments are similar in many cases. The constructive view is based on factual evidence and offers potential solutions. Similarly, fake news is based on false data and offers solutions only to advance a narrative or agenda. Constructive remarks are more formal and less emotional, whereas fake news manipulates the audience with emotions. Therefore, the features used in this paper i.e. stylistic, complexity, and psychological might impact the constructiveness of comments.

Stylistic features include punctuation, capitalization, and syntactic structures that focus on how to express language. Complexity features investigate language nuances such as sentence length and vocabulary diversity. Psychological characteristics reflect underlying emotional and cognitive qualities, providing insight into the intent and attitude of comments.

We also integrate Parts-of-speech (POS)[1] tagging to analyze grammatical structures and identify patterns associated with constructive language. Lastly, we have experimented with the eight basic emotion and bipolar sentiment analyses generated using the NRCLex library[2]. It helps us gauge the emotional tone of comments, distinguishing between positive and negative sentiments to assess their impact on comment constructiveness.

Our main goal is to investigate how these various features influence the constructive nature of comments. By applying linguistic and emotional features to the C3 and YNACC datasets, we aim to develop ML and DL models for classifying constructive comments. Our approach focuses on making classification decisions based on the text content of the comment, independent of the user's previously posted comment. This approach is valuable when such historical data is limited or when evaluating comments solely on their intrinsic merit.

Furthermore, this thesis seeks to bridge the gap between technological advancements in machine learning and the practical needs of content moderation in online news platforms. This study enhances the constructiveness of online news comments to contribute to a more informed, engaged, and respectful digital society. Through rigorous analysis and innovative model development, this thesis endeavors to make a meaningful contribution to the ongoing efforts to improve online communication and maintain the integrity of digital news environments.

[1] For more details, see the Penn Treebank POS Tags.

[2] `https://pypi.org/project/NRCLex/`

1.2 Research Questions

In this study, we have addressed the following research questions:

a) **What features contribute to or detract from the constructiveness of a comment?**

We were inspired by Varda et al.'s method [Kol+20] but chose features (stylistic, complexity, and psychological features) from studies on detecting fake news [ABP22]. We investigate what encourages people to write constructive comments. We also explore emotional features encompassing eight basic emotions and POS tags. This study aims to uncover the features that promote constructive comments and suggest ways to create productive online conversations.

Selected features for study:

- Stylistic, Complexity, and Psychological features
- Eight basic Emotion features
- POS Tag features

b) **What impact do different features have on the constructiveness of comments, as measured by correlation and mean percentage scores?**

This question investigates the specific impact of different features on the constructiveness of comments. By utilizing correlation and mean percentage scores of construction and non-constructive comments, the study aims to provide a detailed analysis of how individual features, such as punctuation, sentiment, and length of sentence, contribute to or detract from the constructiveness of comments.

c) **Which ML and DL models most effectively identify constructive comments?**

We conducted experiments with various ML and DL models to address this research question and determine their efficacy in identifying constructive comments. This involves evaluating Logistic Regression (LR), Support Vector Machine (SVM), Random Forest (RF), K-Nearest Neighbors (KNN), Long Short-Term Memory (LSTM), Bidirectional LSTM (BiLSTM), Convolutional Neural Network (CNN), and Gated Recurrent Unit (GRU) models. Each model will be assessed based on its performance metrics, such as accuracy and F1 score, to identify which techniques best capture the characteristics of constructiveness in online comments. This comparative research seeks to determine the most successful model for increasing online debate by encouraging constructive feedback.

d) **How does the choice of dataset affect model performance in identifying constructive comments?**

This question evaluates the influence of different datasets, such as the C3 and the YNACC, on the performance of ML and DL models. It aims to understand how different datasets impact model effectiveness and identify the constructiveness of comments.

By refining these research questions, we aim to provide a clear and systematic approach to understanding and enhancing the constructiveness of online comments.

1.3 Structure of the Thesis

Chapter 2, Background, delves into the definition and characteristics of constructiveness in the context of online news comments. It explains the properties associated with constructive comments, such as well-reasoned arguments, relevance to the topic, and respectful language. In addition, this chapter discusses state-of-the-art ML and DL algorithms relevant to our experiments for identifying constructive comments.

Chapter 3, Related Work, explores the relevant papers that have focused on identifying the constructiveness of comments and provides a comprehensive overview of previous works, highlighting key studies and methodologies that have contributed to the understanding and analysis of constructive discourse in online environments.

Chapter 4, Data, focuses on the datasets used in the study. This chapter explains the data collection, cleaning, and pre-processing techniques employed before analysis.

Chapter 5, Methods, describes the feature extraction techniques for constructive comment evaluation. Each feature category is explored with its relevance and impact on distinguishing between constructive and non-constructive comments.

Chapter 6, Results, presents the findings of the experiments. It starts with an in-depth feature analysis, examining how a feature influences the constructiveness of a comment. The chapter showcases tables and figures to visualize results, making it easier to interpret the findings. It discusses the performance of ML and DL models with each feature set, highlighting their effectiveness in classifying comments based on their constructiveness. An error analysis is also provided, detailing the models' challenges with abstract features and the resulting higher false positive (FP) and false negative (FN) rates. This analysis offers insights into model performance and areas for improvement. The chapter aims to answer the research questions posed earlier in the thesis, providing empirical evidence to support conclusions.

Chapter 7, Discussion, critically evaluates the results obtained in Chapter 6. It interprets the implications of the findings. The chapter discusses the limitations of this study, and potential biases in the data or methodology, and offers recommendations for future research directions. Through a comprehensive discussion, this chapter aims to deepen understanding of the factors influencing comment quality and constructiveness, incorporating insights from the error analysis.

Chapter 8, Conclusion and Future Work summarizes the main findings of the thesis. It revisits the research questions and objectives, highlighting key insights from the study. The conclusion discusses the broader implications of the research for online community management and digital interaction. Finally, future research directions are outlined, suggesting ways to enhance the current methodology, explore new features, address challenges identified in the error analysis, or apply the findings in different contexts.

2 Technical Background

2.1 Constructiveness

This chapter delves into the definition and characteristics of constructiveness in the context of online news comments. Constructive comments promote meaningful and productive discussions, while non-constructive comments often derail conversations and foster negativity. Additionally, this chapter explores state-of-the-art ML and DL algorithms relevant to NLP. Each algorithm's suitability for processing textual data is examined, emphasising their application in identifying constructive comments.

Characteristics of Constructive Comment in Online News Discourse

Constructive comments in online news articles are essential for fostering a healthy and productive dialogue. Well-reasoned arguments, relevance to the news topic, and respectful language characterize them. These comments contribute positively to the discussion by providing insightful perspectives, fostering healthy debate, and promoting understanding among readers.

- **Well-Reasoned Arguments**: Constructive comments often present logical and coherent arguments supported by evidence or sound reasoning. Such comments go beyond opinions by providing substantiated viewpoints, contributing to a deeper understanding of the topic.
- **Relevance to the Topic**: Another key feature of constructive comments is their relevance to the subject matter of the news article. Comments that stray off-topic or introduce unrelated issues can detract from the discussion and dilute the focus. By maintaining relevance, constructive comments help keep the conversation centred on the core issues, facilitating a more coherent and focused dialogue.
- **Respectful Language**: Respectful language is a cornerstone of constructive comments. It involves expressing viewpoints and disagreements in a manner that is polite and considerate of others' opinions. Respectful discourse encourages participation and reduces the likelihood of conflicts and hostile exchanges, creating a more welcoming environment for diverse perspectives.
- **Solution Oriented**: Constructive comments frequently provide potential solutions to existing problems supported by factual information. It also gives several perspectives to demonstrate the possible outcomes of following such a route, whether they are even harsh truths.

In Table 2.1, we have two comments taken from the C3 dataset, the first comment is non-constructive and the second is constructive.

Label	**Comment**
Non-Constructive	What planet are you from ? This how it always was , how it now is , and how it will forever be . Its not hypocritical . Its the way society works . If you are still not sure how things work in a the real world , do your next column on Sociology PhDs and underemployment .
Constructive	I love these free market pundits that selectively dish out obvious falsehoods disguised as market theory . As I understand the Free Market Theory , scarcity drives up prices and abundance drives them down , no ? If what the fast food industry says is true about labour shortages , the wages at these outlets should be outrageously high but strangely they are not . Obviously if Canadians wont accept $ 10 an hour jobs at Tims , then Tims should have to offer better wages to attract employees from McDonalds and vice versa . This is a fundamental free market tenet . It appears that its the crappy pay that is the problem for these companies , not a lack of people to take the jobs . Why is it that these jobs must be kept at minimum wage at all costs ? Is it so that we can pay a buck less for a cheeseburger ? Why has the invisible hand of the market not driven wages up ? Well its because the temporary worker policies are designed to ensure that they stay low . This is not a case of an industry not being able to compete in the so called global market , a fast food company competes with the outlet across the street . Funny how even public companies (CBC) justify staggering executive salaries and bonuses by saying that they are necessary to attract the best talent but this principle never seems to apply to ordinary workers . Why does scarcity in a profession lead to fabulous wages but scarcity in unskilled labour results in FTW programs ? What we are seeing is a deliberate ongoing program to drive down wages of the working class and maintain a permanent underclass of workers to be exploited at will . Nothing new here . Why do we not import FTWs at near starvation wages to work as policemen , firemen , notaries , engineers , lawyers , radiologists or doctors for that matter , where the savings to public expenditures would really be significant ? The obvious result would be a country full of FTWs , very high unemployment and eventually the pay for these jobs would bottom out at the minimum wage (less 15 %) . Anyone who contends that the FTW program for unskilled labour is necessary or even good for the economy rather than just a plan to keep wages low for blue collar workers is either badly informed or a liar . I dont believe Amanda is badly informed .

Table 2.1: Example comment

The first comment is an example of a non-constructive comment. The comment is not relevant to the topic of the article and does not contribute any meaningful insight to the discussion. It primarily attacks a political party without providing substantial arguments or solutions. The tone is negative and accusatory, focusing on discrediting the party rather than discussing the issues or offering constructive suggestions.

The second comment is an example of a constructive comment. The comment provides a well-reasoned argument by explaining the potential impacts of labour market dynamics and supporting it with logical points. Despite its critical tone, it gives a detailed analysis of the situation, describing the commenter's understanding of free market theory and why they believe the current practices are flawed. It includes evidence and reasoning, discussing the implications of temporary worker policies and comparing different sectors to highlight inconsistencies.

2.2 Natural Language Processing Project Workflow

In NLP tasks, the concept of constructiveness is not entirely new. Many social media websites, such as X (formerly known as Twitter), Reddit, and Facebook, utilize moderation tools like Profanity Checkers[1] to maintain constructive discourse. However, the relevance of constructiveness has increased with the growing volume of comments posted under each comment section.

NLP comes under computer science and artificial intelligence which interacts between computers and human language and enables tasks such as sentiment analysis, chatbot interactions, etc. It primarily deals with text data and processes it through ML and DL algorithms to achieve these tasks. Before applying these models, text data undergoes several processing steps to prepare for analysis. The following sections provide an overview of common text-processing tasks in NLP.

Text Processing

- **Cleaning**: The raw text is cleaned by removing irrelevant information such as extra spaces, special symbols, URLs, email addresses, and sometimes punctuation, depending on the project's requirements. This step ensures that we are working with only relevant data. Additionally, this phase may involve the correction of common spelling errors, expanding contractions (e.g., "can't" to "cannot"), and removing HTML tags if the text comes from web sources.
- **Normalization**: Normalization involves converting text to a standard format. This typically includes converting all text to uppercase or lowercase to ensure consistency, and handling various spellings or forms of words (e.g., British vs. American English spellings). Normalization also encompasses handling of accented characters and transforming them into their unaccented forms.
- **Tokenization**: Tokenization is a crucial step in preparing text data for NLP tasks. In this process, the text is broken into smaller units called tokens. First, sentence

[1]`https://pypi.org/project/profanity-check/`

tokenization splits the text into individual sentences based on delimiters. Next, word tokenization breaks down each sentence into words, including punctuation marks as separate tokens. This step is fundamental for subsequent text analysis processes.

- **Stemming and Lemmatization**
  - **Stemming**: This process reduces words to their base or root form by removing prefixes and suffixes according to predefined rules. For example, "running", "runner", and "ran" may all be reduced to the root "run". Stemming is a rule-based approach and can sometimes result in non-dictionary forms of words.
  - **Lemmatization**: Unlike stemming, lemmatization involves reducing words to their canonical or dictionary form using vocabulary and morphological analysis. It considers the context and part of speech, making it more accurate. For example, "better" is lemmatized to "good" and "is" to "be".
- **Stop Words Removal** Stop words are common words such as "the", "and", "is", which are often removed from the text as they do not contribute significantly to the meaning of the content. Removing stop words helps reduce the text data's dimensionality and remove noise, making the analysis more efficient.
- **TF-IDF (Term Frequency-Inverse Document Frequency)** It is a statistical measure for evaluating the importance of a word in a document relative to a collection of documents (corpus). It balances the frequency of a term in a document with its inverse frequency across the entire corpus.

  a) **Term Frequency (TF)**: Measures how frequently a term appears in a document.

$$TF = \frac{\text{Number of times term appears in a document}}{\text{Total number of terms in the document}} \tag{2.1}$$

  b) **Inverse Document Frequency (IDF)**: Measures how common or rare a term is across all documents in the corpus.

$$IDF = \log\left(\frac{\text{Total number of documents}}{\text{Number of documents with term in it}}\right) \tag{2.2}$$

  Hence, **TF-IDF** is computed as:

$$\text{TF-IDF} = \text{TF} \times \text{IDF}$$

Feature Engineering

In traditional ML, handcrafted features are extracted from the text data to represent the characteristics relevant to the NLP task. Examples include:

- **Word n-grams**: Word n-grams are contiguous sequences of n-words. They capture co-occurrence patterns and are useful for understanding context. For instance, bigrams are pairs of words, and trigrams are triplets of words. These features help capture local dependencies and context within the text.

- **Parts-of-Speech (POS) Tagging**: It labels each word in the text with its corresponding parts of speech, such as nouns, verbs, adjectives, etc. This syntactic information helps us understand the grammatical structure and relationships between words in the text.
- **Named Entity Recognition (NER)**: NER identifies and categorises key elements in the text, including names of individuals, organizations, locations, dates, and other proper nouns. Retrieving structured information from unstructured text is essential for NLP applications.

Word Embeddings (For ML & DL)

Words are represented as numerical vectors in a high-dimensional space to capture semantic relationships such as similarity and analogy. Techniques like Word2Vec and GloVe analyze large text corpora to learn these relationships.

- **Word2Vec**: Word2Vec creates word embeddings using two primary models:
  - Continuous Bag of Words (CBOW): Makes predictions about a target word by examining the context words that surround it.
  - Skip-gram: Predicts the surrounding context words from a given target word.
- **Global Vectors for Word Representation (GloVe)**: GloVe constructs word embeddings by analyzing the co-occurrence of words in a global context. It captures semantic similarities based on a broader context, producing more comprehensive word representations.

Model Training

The preprocessed text data trains machine learning or deep learning models. The model learns patterns and relationships within the data to perform specific NLP tasks, such as sentiment analysis, topic modeling, or machine translation.

- **Model Selection**: Selecting the appropriate model is essential for the success of an NLP project. Various factors, such as the size and nature of the dataset, computational resources, and the specific task requirements, influence this choice.
- **Evaluation and Error Analysis**: After training, the models are evaluated using various metrics such as accuracy, precision, recall, F1 score, and area under the ROC curve (AUC-ROC). False Positive (FP) - an instance where the model incorrectly classifies a non-constructive comment as constructive. Conversely, the False Negative (FN) - where the model classifies the correct constructive comment as non-constructive. Cross-validation and grid search techniques are often employed to fine-tune the models, optimize hyperparameters and ensure their robustness and generalizability.

These foundational processes in NLP set the stage for building sophisticated models capable of performing a wide range of language-related tasks. By systematically cleaning, normalizing, and transforming text data and applying powerful ML and DL models, we can derive meaningful insights and perform complex analyses of textual information.

2.3 Baseline Systems

In this section, we provide a comprehensive overview of the machine learning models[2] and deep learning models[3] utilized in our study as part of the technology stack. These models serve as the backbone of our analysis and play a crucial role in extracting insights from the data.

2.3.1 Logistic Regression

Logistic Regression (LR) is a statistical model designed for binary classification tasks [Bis06]. It makes predictions of the probability of a binary outcome using one or more predictor variables. The model uses a logistic function to ensure the output is between 0 and 1, representing probabilities. The logistic function is defined as:

$$\sigma(z) = \frac{1}{1+e^{-z}} \tag{2.3}$$

where z is the linear combination of input features X and weights β:

$$z = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_n X_n$$

The logistic regression model is expressed as:

$$\log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_n X_n \tag{2.4}$$

where:

- p represents the probability of the event occurring.
- $X_1, X_2, \ldots, X_n$ are the independent variables.
- $\beta_0, \beta_1, \ldots, \beta_n$ are the coefficients representing each independent variable's effect on the dependent variable's log odds.

LR uses maximum likelihood estimation to find the best-fitting model. This involves iteratively adjusting the coefficients to maximize the likelihood of observing the given data.

Hyperparameter

In LR, important hyperparameters include the regularization parameter, the type of regularization, the solver algorithm, and the maximum number of iterations. Proper tuning of these hyperparameters are crucial as they directly influence the model's ability to generalize from the training data to unseen data.

- **Regularization Parameter (C)**: This parameter controls the inverse of the regularization strength.

[2] `https://scikit-learn.org/stable/supervised_learning.html`
[3] `https://www.tensorflow.org/api_docs/python/tf/keras/layers/`

  - A smaller value of C implies stronger regularization, which helps to prevent overfitting by penalizing large coefficients.
  - Conversely, a larger value of C might lead to underfitting if the model is too constrained.
- **Regularization Types**: The choice of regularization affects the model's complexity and performance:
  - L1 Regularization (Lasso): This can lead to sparse models by driving some coefficients to zero, which is useful for feature selection.
  - L2 Regularization (Ridge): It tends to distribute error among all the terms, leading to more balanced models.
  - Elastic Net: It combines L1 and L2 regularization, allowing for feature selection and balanced error distribution.
  - None: Here, no regularization is applied, which might be useful for datasets with little noise.
- **Solver Algorithm**: The choice of the solver (e.g., `liblinear`, `newton-cg`, `lbfgs`, `sag`, `saga`) affects the optimization process of the LR model. Solvers have different convergence properties and computational efficiency, impacting model training time and performance.
- **Maximum Number of Iterations**: This parameter specifies the maximum number of iterations the solver takes to converge. A higher value might be necessary for more complex models or larger datasets to ensure convergence.

LR is simple interpretable and effective in binary classification tasks. It is ideal for understanding the impact of each feature on the likelihood of a comment being constructive or non-constructive. Its efficiency and straightforward implementation make it a practical choice for initial analysis. Additionally, LR provides probabilistic predictions, which can be useful for understanding the confidence of the model's classification.

2.3.2 Support Vector Machine

Support Vector Machines (SVMs) are supervised learning models that find the optimal hyperplane that separates data points of different classes with the maximum margin [CV95]. SVMs can handle linear and non-linear classification through kernel functions such as linear, polynomial, and radial basis functions.

The decision function for SVM is defined as:

$$f(X) = \text{sign}(w \cdot X + b) \tag{2.5}$$

where:

- w is the weight vector.
- X is the input feature vector.
- b is the bias term.

For non-linear classification, the kernel transforms the input data into a higher-dimensional space where a linear hyperplane separates the classes.

Hyperparameter

In SVM, important hyperparameters include the regularization parameter (C), the kernel type, kernel-specific parameters (gamma, degree, coef0), and the maximum number of iterations.

- **Kernel Type**: The kernel function determines the shape of the decision boundary. The main types of kernels include:
  - Linear Kernel: It is suitable for linearly separable data and is often used when the number of features is large.
  - Polynomial Kernel: It can model complex, nonlinear relationships by mapping the data into a higher-dimensional space.
  - Radial Basis Function Kernel: It is a popular choice for many problems due to its ability to handle nonlinearity by mapping data into an infinite-dimensional space.
  - Sigmoid Kernel: It is typically used in neural networks, it can behave like a two-layer neural network.
- **Gamma (γ)**: This parameter defines the influence of a single training example in RBF and polynomial kernels.
  - Scale: $\gamma = \frac{1}{n \cdot \sigma^2}$ where n is the number of features and σ is the variance of the features. This is the default setting and generally works well.
  - Auto: $\gamma = \frac{1}{n}$ where n is the number of features. This might be less effective compared to 'scale' in some cases.
- **Degree**: This parameter is specific to the polynomial kernel and defines the degree of the polynomial.
  - Higher degrees increase the model complexity and can capture more complex relationships but might also lead to overfitting.
- **Coef0**: This parameter is used in polynomial and sigmoid kernels and controls the influence of higher-order versus lower-order terms.
  - A larger value increases the influence of higher-order terms.
  - Typically, fine-tuning this parameter is essential for achieving the best model performance when using polynomial or sigmoid kernels.

SVMs can handle high-dimensional feature spaces and robustness in binary classification. They are well-suited for text classification tasks, especially with feature extraction methods like TF-IDF. The capacity of SVMs to manage non-linear boundaries using kernel tricks makes them versatile for complex data distributions. SVMs also have strong theoretical foundations, providing guarantees on the generalization error through margin maximization.

2.3.3 K-Nearest Neighbor

K-Nearest Neighbors (KNNs) is an instance-based learning algorithm that classifies a data point based on the majority class of its k-nearest neighbors in the feature space [HTF09]. It is non-parametric, making no assumptions about the underlying data distribution.

The KNN algorithm works as follows:

- Calculate the distance between the input data point and all points in the training set using a distance metric such as Euclidean, Manhattan, or cosine similarity.
- Select the k training data points closest to the input data point.
- Assign the input data point to the class that appears most frequently among its k closest neighbors.

The distance metric commonly used is the Euclidean distance, defined as:

$$d(X_i, X_j) = \sqrt{\sum_{m=1}^{n} (X_{im} - X_{jm})^2} \tag{2.6}$$

where:

- X_i and X_j are two points in the feature space.
- n is the number of features.
- X_{im} and X_{jm} are the m-th features of X_i and X_j respectively.

Hyperparameter

In KNN, important hyperparameters include the number of neighbors, distance metric, weight function, and algorithms for computing nearest neighbors.

- **Number of Neighbors (k)**: This parameter determines how many neighbors are considered when making predictions.
  - A smaller value of k can lead to a model that captures more noise from the training data, potentially causing overfitting.
  - A larger value of k results in a smoother decision boundary, which can improve generalization but may also cause underfitting.
- **Distance Metric**: This parameter defines how the distance between data points is calculated. Common metrics include:
  - Euclidean Distance: It is the most commonly used distance metric, suitable for many problems.
  - Manhattan Distance: It measures distance as the sum of absolute differences, which can be useful for high-dimensional data.
  - Minkowski Distance: It is a generalization of Euclidean and Manhattan distances, controlled by a parameter p.

  - Hamming Distance: It is used for categorical data and measuring the proportion of differing features.
- **Weight Function**: This parameter determines how each neighbor contributes to the final prediction.
  - Uniform: Here, all neighbors contribute equally.
  - Distance: Here, neighbors contribution is weighted by the inverse of their distance, so closer neighbors have a greater influence.
- **Algorithm**: This parameter specifies the algorithm to compute the nearest neighbors. Different algorithms have different computational efficiencies and performance characteristics.
  - Brute Force: It is a straightforward approach that computes the distance between all pairs, suitable for small datasets.
  - Ball Tree: It is efficient for larger datasets with few dimensions.
  - KD Tree: It is efficient for moderate-sized datasets with a moderate number of dimensions.
  - Auto: It automatically selects the best algorithm based on the input data.
- **Leaf Size**: This parameter is used in Ball Tree and KD Tree algorithms and affects the tree construction and query speed.
  - A smaller leaf size can lead to faster query times but slower tree construction times.
  - A larger leaf size can lead to faster tree construction but slower query times.
- **Metric Parameters**: Additional parameters for the distance metric, such as the power parameter p for Minkowski distance.
  - Tuning these parameters can help optimize the model for specific data or problem domains.

KNN is simple, easy to implement, and effective in capturing local patterns in the data. It is particularly suitable for tasks where local similarities are important, such as distinguishing between constructive and non-constructive comments. KNN's instance-based nature allows it to adapt naturally to complex decision boundaries. However, the computational cost during prediction can be high, especially for large datasets, as distances to all training points must be calculated.

2.3.4 Random Forest

Random Forests (RF) is an ensemble learning method that constructs multiple decision trees during training and outputs the mode of the classes (classification) or mean prediction (regression) of the individual trees [Bre01]. Each tree in the forest is trained using a random selection of the data and features, a technique known as bootstrap aggregating or bagging.

A decision tree splits the data into subsets based on the value of input features, aiming to improve the purity of the subsets at each split. The final decision tree is built by recursively splitting the data until a stopping criterion is met, such as a maximum depth or a minimum number of samples per leaf.

RF works as follows:

- Randomly select n samples with replacements from the training set to create a bootstrap sample.
- For each bootstrap sample, grow an unpruned decision tree by recursively splitting the data on the best features.
- Aggregate the predictions of all individual trees to make the final prediction.

Hyperparameter

In an RF model, important hyperparameters include the number of trees in the forest, the minimum number of samples required to divide a node, the maximum depth of the trees, the minimum number of samples needed for a leaf node, the maximum number of features considered for splitting a node, and the bootstrap method.

- **Number of Trees in the Forest**: This parameter determines how many trees are grown in the forest.
  - A larger number of trees generally leads to better performance and stability but increases computational cost.
  - A smaller number of trees may result in a faster model but can reduce accuracy.
- **Maximum Depth of the Trees**: This parameter limits how deep each tree in the forest can grow.
  - A larger maximum depth allows the model to capture more complexity but can lead to overfitting.
  - A smaller maximum depth reduces the risk of overfitting but may underfit the data.
- **Minimum Number of Samples Required to Split a Node**: This parameter specifies the minimum number of samples needed to split an internal node.
  - A larger value can prevent the model from learning overly specific patterns, reducing overfitting.
  - A smaller value allows the model to capture more details but increases the risk of overfitting.
- **Minimum Number of Samples Required at a Leaf Node**: This parameter determines the minimum number of samples a leaf node must have.
  - A larger value can help smooth the model and prevent overfitting.
  - A smaller value allows for capturing finer details but may lead to overfitting.
- **Maximum Number of Features Considered for Splitting a Node**: This parameter specifies the maximum number of features considered for splitting a node.

  - Auto: It takes $\sqrt{n\_features}$ features which is the default setting for classification tasks.
  - Log2: It takes $\log_2 n\_features$ features.
  - None: It takes all features, which can lead to better performance but increases the risk of overfitting and computational cost.
  - Other Values: It Specifies a fraction (e.g., 0.5) or an absolute number of features (e.g., 10).
- **Bootstrap Method**: This parameter specifies whether bootstrap samples are used when building trees.
  - True: It uses bootstrap samples by default. This method allows for different trees to be built from different subsets of the data, leading to more diverse models.
  - False: It uses the entire dataset to build each tree. This can be useful for smaller datasets.

RFs are robust, able to handle high-dimensional data, and interpretable. They reduce overfitting by averaging multiple trees, improving predictive accuracy and robustness. RFs can capture complex patterns and interactions between features, making them suitable for identifying constructiveness in comments. Additionally, feature importance scores provided by RFs offer insights into which features are most influential in the classification process, aiding in the interpretability of the model.

2.3.5 Long Short-Term Memory

Long short-term memory (LSTM) networks are a type of recurrent neural network (RNN) designed to capture long-term dependencies in sequential data [HS97]. LSTM use memory cells with input, output, and forget gates, to regulate the flow of information and mitigate the vanishing gradient problem commonly found in traditional RNNs.

An LSTM cell's operations can be summarized as follows:

- **Forget gate** f_t: Decides what information to discard from the cell state.
- **Input gate** i_t: Determines which values from the input to update the cell state.
- **Cell state update** $\tilde{C}_t$: Creates a new candidate vector.
- **Output gate** o_t: Decides what part of the cell state to output.

The equations for an LSTM cell are:

$$\begin{aligned}
f_t &= \sigma(W_f \cdot [h_{t-1}, X_t] + b_f) \\
i_t &= \sigma(W_i \cdot [h_{t-1}, X_t] + b_i) \\
\tilde{C}_t &= \tanh(W_C \cdot [h_{t-1}, X_t] + b_C) \\
C_t &= f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t \\
o_t &= \sigma(W_o \cdot [h_{t-1}, X_t] + b_o) \\
h_t &= o_t \cdot \tanh(C_t)
\end{aligned} \tag{2.7}$$

where:

- h_t is the hidden state at time step t.
- C_t is the cell state at time step t.

Hyperparameter

The LSTM model has important hyperparameters like the number of units in each LSTM layer, the number of layers, the dropout rate, the batch size, the learning rate, the sequence length, the optimization algorithm, gradient clipping, and the number of epochs. It is essential to tune these hyperparameters as they directly impact the model's capacity to generalize from the training data to new, unseen data.

- **Number of Units**: This parameter determines the number of memory units (neurons) in each LSTM layer.
  - More units can capture more complex patterns but increase the risk of overfitting and computational cost.
  - Fewer units may not capture sufficient patterns, leading to underfitting.
- **Number of Layers**: This parameter defines the number of LSTM layers in the network.
  - More layers can model more complex sequences but may increase the risk of overfitting and longer training times.
  - Fewer layers reduce the model complexity and training time but may not capture enough sequential patterns.
- **Dropout Rate**: This parameter specifies the fraction of the units to drop for the linear transformation of the inputs.
  - A higher dropout rate can help prevent overfitting by making the model less sensitive to specific neurons.
  - A lower dropout rate may lead to overfitting as the model relies too much on specific neurons.
- **Batch Size**: This parameter defines the number of samples processed before the model is updated.
  - A larger batch size leads to faster training but might reduce the model's generalisation ability.
  - A smaller batch size provides a more accurate estimate of the gradient but increases training time.
- **Learning Rate**: This parameter controls the step size taken at each iteration while moving toward a minimum of the loss function.
  - A higher learning rate can accelerate the training but might miss the optimal solution.
  - A lower learning rate ensures convergence to a minimum but increases training time.

- **Sequence Length**: This parameter specifies the length of the input sequences used by the LSTM model.
  - A longer sequence length provides more context for the model but increases computational complexity and risk of overfitting.
  - A shorter sequence length reduces the complexity but may not capture sufficient context.
- **Optimization Algorithm**: This parameter specifies the algorithm to optimize the loss function.
  - Common choices consist of Adam, RMSprop, and SGD.
  - Adam is popular for its adaptive learning rate properties.
- **Gradient Clipping**: This parameter prevents the gradients from exploding by capping them to a maximum value.
  - Useful for maintaining stable training in RNNs like LSTM.
- **Epochs**: This parameter determines the number of times the learning algorithm works through the entire training dataset.
  - More epochs can improve the model performance but increase the overfitting risk and training time.

LSTMs can model long-term dependencies and capture contextual information in text, which is crucial for understanding the constructiveness of comments. Their capacity to learn from entire sequences of words allows them to understand the context and nuances in text data. LSTMs are particularly useful for tasks where the order of words significantly impacts the meaning.

2.3.6 Bidirectional LSTM

Bidirectional LSTMs (BiLSTMs) extend traditional LSTMs by processing input sequences in both forward and backward directions [HXY15]. This allows the model to capture context from past and future states, providing a more comprehensive understanding of the sequence.

BiLSTMs use two LSTM layers running in parallel; one processes the sequence from start to end, while the other processes it starting from the end. The outputs from both layers are then combined.

Hyperparameter

Similar to LSTMs, the BiLSTM model has hyperparameters which include the number of units in each LSTM layer, the number of layers, the dropout rate, the batch size, the learning rate, the sequence length, the optimization algorithm, gradient clipping, and the number of epochs.

BiLSTMs can capture context from both directions, enhancing the model's understanding of text sequences. This bidirectional processing improves the identification of constructive comments by considering surrounding words and phrases from past and future contexts. BiLSTMs are particularly effective for tasks where understanding the full context of a sequence is important.

2.3.7 Gated Recurrent Unit

Gated recurrent units (GRUs) are a type of recurrent neural network similar to LSTMs but with a simplified architecture [Cho+14]. GRUs combine the forget and input gates to form a single update gate, simplifying the model. GRUs maintain the ability to capture long-term dependencies while being computationally more efficient than LSTMs.

The equations for a GRU cell are:

$$\begin{aligned} z_t &= \sigma(W_z \cdot [h_{t-1}, X_t] + b_z) \\ r_t &= \sigma(W_r \cdot [h_{t-1}, X_t] + b_r) \\ \tilde{h}_t &= \tanh(W_h \cdot [r_t \cdot h_{t-1}, X_t] + b_h) \\ h_t &= (1 - z_t) \cdot h_{t-1} + z_t \cdot \tilde{h}_t \end{aligned} \tag{2.8}$$

where:

- z_t is the update gate.
- r_t is the reset gate.
- $\tilde{h}_t$ is the candidate activation.
- h_t is the new hidden state.

Hyperparameter

Similar to LSTMs and BiLSTMs, the GRU model has hyperparameters which include the number of units in each GRU layer, the number of layers, the dropout rate, the batch size, the learning rate, the sequence length, the optimization algorithm, gradient clipping, and the number of epochs.

GRUs can model long-term dependencies in text data while being more computationally efficient than LSTMs. This makes them suitable for tasks requiring contextual understanding, such as identifying constructive comments, with lower computational overhead. GRUs' simpler architecture allows for faster training and inference while still maintaining the ability to capture important sequence information.

2.3.8 Convolutional Neural Network

Convolutional Neural Networks (CNNs) [KSH12] are deep learning models predominantly used in computer vision tasks but have also been adapted for NLP. CNNs are effective at capturing local patterns in data through the use of convolutional layers.

A CNN consists of several key components:

- **Convolutional Layers**: These layers utilize convolutional filters on the input data to detect local patterns. Each filter slides over the input data, conducting multiplications on each element and summing the results to produce a feature map.
- **Activation Functions**: To provide non-linearity to the model, non-linear functions, such as ReLU (Rectified Linear Unit), are applied to the feature maps.
- **Pooling Layers**: These layers reduce the spatial dimensions of the feature maps, typically using max pooling or average pooling. This process helps to reduce the computational complexity and extract dominant features.
- **Fully Connected Layers**: The feature maps are flattened and passed through fully connected layers after several convolutional and pooling layers to produce the final output.

The convolution operation can be described mathematically as:

$$f(x,y) = (I * K)(x,y) = \sum_{i=-m}^{m} \sum_{j=-n}^{n} I(x+i, y+j) \cdot K(i,j) \tag{2.9}$$

where:

- I is the input image (or data).
- K is the convolutional kernel (filter).
- (x, y) are the coordinates of the output feature map.
- m and n define the size of the kernel.

Hyperparameter

The CNN model has hyperparameters which include filters, filter size, stride, padding, convolutional layers, fully connected layers, dropout rate, batch size, learning rate, and optimization algorithm.

- **Number of Filters (kernels)**: This parameter determines the number of filters in each convolutional layer.
  - More filters can capture more complex features but increase computational cost.
  - Fewer filters reduce computational complexity but may not capture sufficient features.
- **Filter Size**: This parameter defines the size of the filters (kernels) used in the convolutional layers.
  - Common choices include 3x3, 5x5, and 7x7.
  - Larger filters capture more context but increase computational cost.
  - Smaller filters capture finer details but may miss larger patterns.
- **Stride**: This parameter specifies the step size with which the filter moves across the input.

  - A larger stride reduces the spatial dimensions more quickly but may lose information.
  - A smaller stride retains more spatial information but raises the computational cost.

- **Padding**: This parameter determines whether and how much padding is added to the input.
  - Valid: No padding is added, resulting in smaller output dimensions.
  - Same: Padding is added to keep the output dimensions the same as the input dimensions.
- **Number of Fully Connected Layers**: This parameter defines the number of fully connected (dense) layers in the network.
  - More layers can capture more complex interactions but increase the risk of overfitting and computational cost.
  - Fewer layers reduce computational complexity but may limit the model's capacity to learn complex relationships.
- **Pooling Layers**: This parameter determines the type and size of pooling layers used to reduce spatial dimensions.
  - Max Pooling: It selects the maximum value from the input window.
  - Average Pooling: It computes the average of the values in the input window. Common sizes include 2x2 and 3x3.

CNNs are capable of detecting specific patterns and complex features within textual information. When applied to text classification, CNNs detect n-grams and semantic patterns within the text by applying convolutional filters to word embeddings or character-level representations. This capability is useful for identifying the constructiveness of comments, as local linguistic patterns and context can significantly influence the classification. CNNs are also computationally efficient, making them suitable for large-scale text-processing tasks.

3 Related Work

3.1 Related Work

The analysis and identification of constructive comments in online discussions have garnered significant attention in recent years, particularly with the increasing prevalence of social media platforms. This section explores the relevant papers focusing on identifying constructiveness and detecting misinformation, providing a comprehensive overview of existing research.

Niculae et al. (2016) [ND16] introduced a framework for evaluating group discussions by analyzing conversational patterns and linguistic cues from the first 20 seconds. Reitter and Moore (2007) [RM07] discovered that successful dialogues are marked by long-term adaptation and alignment of linguistic structures at syntactic, lexical, and character levels. They attempted to predict the success of a dialogue using only the first five minutes of conversation.

Pitler and Nenkova [PN08] combined lexical, syntactic, and discourse features to develop a predictive model for human judgments of text readability. They empirically showed that discourse relations significantly impact perceived text quality, demonstrating their robustness across tasks like predicting and ranking readability.

Nguyen et al. (2021) [NVN21] presented the UIT-ViCTSD dataset, which includes 10,000 annotated Vietnamese comments for detecting constructive and toxic speech. Utilizing the PhoBERT model, the authors achieve F1 scores of 78.59%, and 59.40% for detecting constructive and toxic comments, respectively.

Saleem et al. [Sal+17] used critique keyword-based methods for detecting hateful speech and proposed a novel approach using data from self-identifying hateful communities. This method avoids costly annotations and significantly outperforms current state-of-the-art techniques across multiple platforms.

Warner et al. [WH12] define hate speech as abusive content targeting specific groups and introduce a method for detection using high-frequency stereotypical words. They created a hate speech corpus and achieved 94% accuracy in classifying anti-Semitic speech.

Wulczyn et al. [WTD16] presented a method combining crowdsourcing and machine learning to analyze and classify personal attacks at scale, addressing online personal attacks. They generated a large corpus from English Wikipedia and showed that personal attacks are widespread, not just caused by a few malicious users.

Mayer et al. [MC14] explored the impact of online comments on community journalism, based on surveys of editors and internet users. Findings indicate that high comment volumes may deter journalists from engaging with audiences, while journalist encouragement and perceived virtual community significantly boost comment frequency. Active journalist involvement is crucial for fostering online communities.

Several studies have explored the automatic detection and moderation of comments to maintain constructive discourse on online platforms. For example, Chen et al. [CXW11] studied the impact of various moderation techniques on the quality of online discussions, highlighting the importance of automated tools in managing large volumes of user-generated content.

Sentiment analysis and toxicity detection are closely related to identifying constructive comments. Sentiment analysis involves classifying text based on expressed sentiment, such as positive, negative, or neutral [HMM14]. Toxicity detection focuses on identifying harmful or abusive language in online comments. Wulczyn et al. [WTD16] developed a model for detecting personal attacks in online discussions using linguistic features and machine learning algorithms, highlighting the challenges and effectiveness of automated toxicity detection systems.

Kobayashi et al. (2021) [Kob+21] investigated the ranking of user comments on news articles to enhance user experience by focusing on constructiveness as a key metric. They conducted a case study involving an in-house competition to improve the ranking performance of constructive comments and demonstrated the effectiveness of the top model for commercial applications.

Napoles et al. (2017) [Nap+17] introduced a dataset and annotation scheme for identifying good online conversations, ERICs: Engaging, Respectful, and Informative Conversations. They developed a taxonomy for thread and comment features, annotated a large dataset comprising Yahoo News comment threads and the Internet Argument Corpus, and analyzed features characteristic of ERICs, making it one of the largest and most detailed annotated corpora of online dialogues. Kolhatkar et al. [Kol+20] extended this line of research by examining the use of deep learning models for the same task. They applied a variety of neural network architectures, including CNNs and LSTMs, to identify constructive comments, demonstrating the superior performance of deep learning approaches in capturing nuanced language patterns.

Despite the progress made in identifying constructive comments, several challenges remain. The complexity of natural language, the presence of sarcasm, and the context-dependent nature of constructiveness pose significant hurdles. Moreover, the need for large annotated datasets for training robust models continues to be a bottleneck.

Future research directions include the development of more sophisticated models that can better understand context and user intent, integrating multimodal data (e.g., combining text with images or videos), and creating more comprehensive annotated datasets. Exploring transfer learning and domain adaptation techniques could also help improve model performance across different platforms and languages.

This related work section has reviewed key studies and contributions in the field of NLP related to the identification and analysis of constructive comments. The evolution from traditional ML models to advanced DL architectures highlights the progress and ongoing challenges in this area. By building on these foundational works, our study aims to further advance the understanding and identification of constructiveness in online comments.

4 Data

4.1 Datasets

Finding suitable datasets for processing and evaluating the constructiveness of comments is a challenge due to the scarcity of annotated datasets labeled with constructiveness scores. In this experiment, we utilized the C3 and YNACC datasets, where comments are labeled with constructiveness scores ranging from 0 to 1. A score of 1 indicates a fully constructive comment, while 0 indicates a non-constructive comment. The description of the datasets is as follows:

- **Constructive Comments Corpus (C3)**: The C3 dataset[1] comprises 12,000 comments on news articles from the SFU Opinion and Comments Corpus (SOCC). The total number of constructive labels is 6516 and the number of non-constructive labels is 5484. The SOCC contains opinion articles and their associated comments posted on the Canadian newspaper *The Globe and Mail* between 2012 and 2016. Crowdsourced workers annotate each comment in the C3 dataset for constructiveness, toxicity, and various constructiveness and non-constructiveness characteristics. In this study, we have utilized only the constructiveness values, calculated as the average ratings from ten crowd workers.
- **Yahoo News Annotated Comments Corpus (YNACC)**: The YNACC dataset[2] consists of 23,383 annotated comments, of which 22,795 are labelled "Constructive" or "Non-Constructive". The total number of constructive labels is 11812 and the number of non-constructive labels is 10983. It includes 9,200 comments and 2,400 threads capturing various characteristics such as persuasiveness, sentiment, and tone. The dataset provides additional information on whether a comment agrees or disagrees with the article and categorizes comments into positive or respectful, argumentative, snarky or humorous, etc. These comments, posted on Yahoo News articles in 2016, are used to evaluate our models' generalizability across different datasets.

We developed various supervised, statistical, and deep learning models including logistic regression, LSTM, biLSTM, and CNN. These models were trained and tested using the datasets and the extracted features from the comments.

4.2 Data Pre-processing

The collected datasets contained various pieces of information that were not required for our work. We only needed the comments and their respective constructiveness scores for our

[1]`https://github.com/sfu-discourse-lab/SOCC`
[2]`https://github.com/cnap/ynacc`

experiment. However, we found many irrelevant pieces of information in the datasets, which were not considered for our experiment. Before extracting features from the comments, we cleaned them using the following steps:

- **Removing hyperlinks:** We removed URLs i.e. HTTP and WWW text from the comments as this information does not help to identify the constructiveness of comments.
- **Condensing double spaces:** We observed that users often included multiple double spaces in their comments. Therefore, we removed these extra spaces.
- **Normalizing contractions:** We replaced contractions with their expanded forms. For example, the word "can't" is replaced with "cannot," and "we'll" is replaced with "we will."

5 Methods

5.1 Feature Extraction

In this section, we have explored the feature extraction process to find the influence of various features on the constructiveness of comments. We worked on multiple feature sets, to unveil various facets of linguistic expression and emotional tones within comments. Our feature extraction starts with stylistic, complexity, and psychological features taken from Aich et al.'s [ABP22] work. These features help to understand the various characteristics influencing the constructiveness of comments. We also worked on emotional features with the help of the NRCLex library [MT13] that provide eight fundamental emotions like anger, anticipation, trust, etc., and sentiment polarity. Lastly, we also worked on 36 unique parts-of-speech (POS) tags as per the Penn TreeBank that help to understand the grammatical structures, and linguistic patterns that can explain the constructive discourse. Through these various feature extraction approaches, we aim to find the relationship between linguistic expression, emotional resonance, and grammatical structure with the constructiveness of comments.

Stylistic Features

In our study, we examined stylistic features that affect the constructiveness of comments. These features include different aspects of writing style and linguistic patterns. Here, the focus is on sixteen sub-features that capture various stylistic elements, ranging from punctuation usage to the frequency of specific word types as per Aich et al.'s [ABP22] work. These sub-features are as follows:

- **Frequency of quotation marks:** It measures the average number of quotations per sentence.
- **Frequency of punctuation:** It measures the average number of punctuation marks used per sentence.
- **Number of unique forms of punctuation**: It counts the unique punctuation types in a comment.
- **Frequency of exclamations**: It measures the average number of exclamation marks ('!') used per sentence.
- **Frequency of stopwords**: It measures the average number of stopwords using NLTK's stopwords list per sentence.
- **Frequency of camel-case words**: It measures the average number of words beginning with an uppercase character followed by ≥ 1 lowercase character in a sentence.
- **Frequency of negations**: It measures the average number of occurrences of *no*, *never*, or *not* in a sentence.

- **Frequency of proper nouns**: It measures the average number of singular and plural proper noun (NNP and NNPS) POS tags per sentence.
- **Frequency of user mentions**: It measures the average number of mentions per sentence.
- **Frequency of hashtags**: It measures the average number of '#' hashtags per sentence.
- **Frequency of misspelled words**: It measures the average number of words not considered valid by the PyEnchant[1] spell checkering library in a sentence.
- **Frequency of out-of-vocabulary words**: It measures the average number of words not in SentiWordNet (a lexical resource that provides the sentiment score of a word using WordNet) [BES10a] in a sentence.
- **Frequency of nouns**: It measures the average number of NNP, NNPS, NN, and NNS POS tags in a sentence.
- **Frequency of past tense words**: It measures the average number of past tense verbs identified by POS tags VBD and VBN in a sentence.
- **Frequency of verbs**: It measures the average number of verbs identified by POS tags VB, VBD, VBG, VBN, VBP, and VBZ in a sentence.
- **Frequency of interrogative words**: It measures the average number of interrogative words identified by POS tags WRB, WDT, and WP in a sentence.

An example of the stylistic features of the C3 dataset is shown in Table 5.1. From the comment, we observe three unique punctuation marks: one question mark (?), three commas (,), and four periods (.). Therefore, the Number of Unique Forms of Punctuation value is 3. Additionally, the comment consists of 5 sentences. Thus, the Frequency of Punctuation per Sentence is calculated as (1 + 3 + 4) / 5 = 1.6.

Comment	**Frequency of punctuation**	**Number of unique forms of punctuation**	**Type-Token Ratio (TTR)**	**Sentiment Score**
What planet are you from ? This how it always was , how it now is , and how it will forever be . Its not hypocritical . Its the way society works . If you are still not sure how things work in a the real world , do your next column on Sociology PhDs and underemployment .	1.6	3	0.7241	-1.375

Table 5.1: Stylistic features example

[1] `https://pyenchant.github.io/pyenchant/`

Complexity Features

We also analyzed several complexity features to gain insight into the structural complexity of the comments. Following the approach of Aich et al.'s [ABP22] work, we focused on four key sub-features that quantify different aspects of textual complexity which are as follows:

- **Word Count:** The total number of words in the comment.
- **Mean Word Length:** It is the average number of characters per word.
- **Type-Token Ratio (TTR)**: It measures how varied the words in a text are. It's calculated by dividing the number of unique words by the total number of words [MJ10]. A higher TTR indicates more vocabulary diversity. The formula for calculating the TTR is as follows:

  $$TTR = \frac{N_{\text{unique}}}{N_{\text{total}}} \tag{5.1}$$

  where:

  N_{unique} is the number of unique words (types) in the text,

  N_{total} is the total number of words (tokens) in the text.

- **Measure of Textual Lexical Diversity (MTLD)**: Measures how many different words are used in a text. It breaks the text into parts, looks at how many unique words are in each part, and calculates an average. (McCarthy and Jarvis, 2010) [MJ10]. The formula for calculating MTLD is as follows:

  $$MTLD = \frac{N}{\sum_{i=1}^{n} \frac{N_i}{T_i}} \tag{5.2}$$

  where:

  N is the total number of words in the text,

  n is the total number of segments into which the text is divided,

  N_i is the number of words in segment i,

  T_i is the type-token ratio (TTR) of segment i.

Table 5.1 shows an example of the complexity feature. From the comments, we observe that we have 58 words, and the unique words are 42. Thus, the TTR is calculated as 42/58 = 0.7241.

Psychological Features

For the Psychological feature calculation, we have utilized the sentiment scores derived from word-level sentiment analysis using SentiWordNet (Baccianella et al., 2010) [BES10b].

- **Sentiment Score:** The average sentiment scores are calculated using the SentiWordNet for all available vocabulary words in a comment. We got a sentiment score of -1.375 for the comment as provided in Table 5.1.

Emotion Features

One of the key aspects of identifying constructive and non-constructive comments is understanding the emotional content embedded in the text. To explore this, we experimented with multiple emotion detection methods, including pre-trained models from Hugging Face and the NRCLex library.

Initially, we experimented with two BERT-based models, namely bert-base-uncased-goemotions-original-finetuned and bert-base-uncased, designed for general-purpose emotion detection. These models were fine-tuned on emotion-related tasks and leveraged the transformer architecture to predict multiple emotions in text. The goal was to determine whether these emotional features could effectively distinguish between constructive and non-constructive comments. However, during experimentation, we found that the emotions extracted by these models did not show a clear correlation with the constructiveness of the comments. Therefore, we shifted to the NRCLex library [MT13], which allowed us to derive emotional and sentiment aspects of the comments more effectively.

NRCLex provides emotional scores for fundamental emotions such as anger, anticipation, disgust, fear, joy, sadness, surprise, and trust, in addition to sentiment polarity scores that reveal the overall positive or negative sentiment expressed within the text. By using NRCLex, we obtained valuable emotional nuances and sentiment orientations.

These emotional features played a crucial role in our research, offering deeper insights into user attitudes, sentiments, and emotional responses that significantly influence the constructiveness of comments.

We present emotional scores attributed in Table 5.2 for the comment mentioned in Table 5.1. Here, we can observe that the emotions of surprise and joy are assigned a value of 0.25, while the remaining emotions have a value of 0. Additionally, the positive and negative polarity scores of 0.25, signify the consistent sentiment polarity within the comment.

fear	**trust**	**surprise**	**sadness**	**disgust**	**joy**	**anticipation**	**anger**	**positive**	**negative**
0	0	0.25	0	0	0.25	0	0	0.25	0.25

Table 5.2: Emotion features example

Part-of-speech Tagging

In Part-of-speech (POS) tagging, we extracted the POS tag of each comment using the Natural Language Toolkit (NLTK) library as per the Penn Treebank tag set, which comprises 36 unique tags. Afterwards, we calculated the average occurrence of these tags per sentence to find the relationship with constructive value. Table 5.4 provides a detailed explanation of these tags. This tagging helps us to identify the grammatical structures and linguistic patterns within the comments to understand the constructiveness of the comment.

An example of POS tag values for the comment mentioned in Table 5.1 is shown in Table 5.3, which contains the Coordinating conjunction (CC), Cardinal number (CD), and Determiner (DT) values as 0.4, 0, and 0.8, respectively.

CC	CD	DT
0.4	0	0.8

Table 5.3: POS Tag features example

Number	Tag	Description
1	CC	Coordinating conjunction
2	CD	Cardinal number
3	DT	Determiner
4	EX	Existential there
5	FW	Foreign word
6	IN	Preposition or subordinating conjunction
7	JJ	Adjective
8	JJR	Adjective, comparative
9	JJS	Adjective, superlative
10	LS	List item marker
11	MD	Modal
12	NN	Noun, singular or mass
13	NNS	Noun, plural
14	NNP	Proper noun, singular
15	NNPS	Proper noun, plural
16	PDT	Predeterminer
17	POS	Possessive ending
18	PRP	Personal pronoun
19	PRP$	Possessive pronoun
20	RB	Adverb
21	RBR	Adverb, comparative
22	RBS	Adverb, superlative
23	RP	Particle
24	SYM	Symbol
25	TO	to
26	UH	Interjection
27	VB	Verb, base form
28	VBD	Verb, past tense
29	VBG	Verb, gerund or present participle
30	VBN	Verb, past participle
31	VBP	Verb, non-3rd person singular present
32	VBZ	Verb, 3rd person singular present
33	WDT	Wh-determiner
34	WP	Wh-pronoun
35	WP$	Possessive wh-pronoun
36	WRB	Wh-adverb

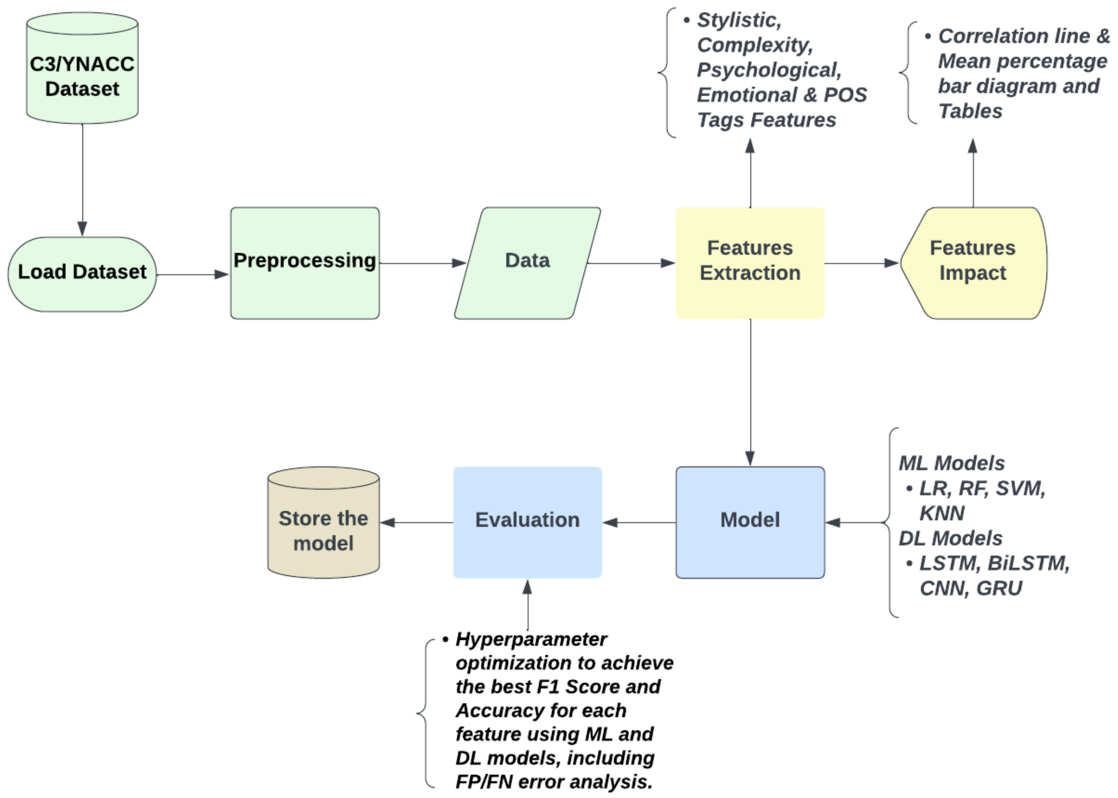
Table 5.4: POS Tag description

6 Experiments and Results

In this section, we have explained how to analyze the constructiveness of comments on various features. Subsequently, we present experimental results obtained through statistical and deep learning models, offering insights into their impact on comment constructiveness.

6.1 Experiment Setup

This section describes the entire architecture of our experiment from the data collection to the final result shown in Figure 6.1



C3/YNACC Dataset
Load Dataset
Preprocessing
Data
Features Extraction
• Stylistic, Complexity, Psychological, Emotional & POS Tags Features
Features Impact
• Correlation line & Mean percentage bar diagram and Tables
Model
ML Models
• LR, RF, SVM, KNN
DL Models
• LSTM, BiLSTM, CNN, GRU
Evaluation
• Hyperparameter optimization to achieve the best F1 Score and Accuracy for each feature using ML and DL models, including FP/FN error analysis.
Store the model

Figure 6.1: Experimental Flowchart

For our experiments, we utilized two datasets: C3 and YNACC. These datasets were loaded and prepared for analysis by undergoing a series of preprocessing steps to ensure data quality and consistency.

In the preprocessing phase, we removed hyperlinks, expanded contractions (e.g., converting "can't" to "cannot"), etc. Post-preprocessing, we extracted a comprehensive set of features from the comments, including Stylistic, Complexity, Psychological, Emotional, and POS Tags features. To understand the impact of these features on our classification task, we utilized correlation diagrams, mean percentage bar diagrams, and tables for analysis.

We trained several models using the extracted features to classify the comments. Our modeling approach encompassed baseline ML models (e.g., LR, SVM, KNN, and RF) and DL models (e.g., LSTM, BiLSTM, CNN, and GRU). To optimize the performance of each model, we performed hyperparameter tuning using grid search and randomized search to identify the best combination of hyperparameters.

The models' performance was evaluated using F1 score and accuracy and we did the error analysis using false positive and false negatives. Finally, we stored the model that can be later deployed to classify new, unseen comments provided by users. This will allow for real-time classification, validating the effectiveness of our models in practical applications.

6.2 Feature Analysis

In this section, we have explored the analysis of how various linguistic and emotional features contribute to the constructiveness of comments. With the help of the tables and figures, we provided a comprehensive examination utilizing correlation, mean values and mean percentages of constructive and non-constructive comments for both the C3 and YNACC datasets. By analyzing these datasets, we aimed to uncover which features play pivotal roles in determining the constructiveness of comments.

6.2.1 Stylistic Features

1. Correlations between Stylistic Features and Constructiveness

Table 6.1 provides insights into the correlations between various stylistic features and the constructiveness of comments in the C3 and YNACC datasets. The correlations highlight how different linguistic elements contribute differently to the constructiveness of online discourse. Notably, *Punctuation Types* shows a strong positive correlation with constructiveness in the C3 dataset (0.4770) and a moderate positive correlation in YNACC (0.0810). This suggests that comments with diverse punctuation marks are more likely to be perceived as constructive, indicating a structured and coherent expression of ideas. Similarly, features like *Stopwords* exhibit positive correlations in both datasets (0.3255 in C3 and 0.0727 in YNACC), indicating their role in facilitating grammatical flow and clarity in constructive comments.

Conversely, features such as *Exclamation Marks*, *Hashtags*, and *Interrogative Words* show negative correlations with constructiveness, particularly in the C3 dataset (-0.1404, -0.0135, and -0.0501, respectively). This suggests that these linguistic elements are more prevalent in non-constructive comments, potentially indicating emotional expression, lack of focus, or confrontational language rather than substantive contributions to the discussion. Overall, the correlations underscore the importance of stylistic features in influencing how comments are perceived in terms of their constructiveness across different datasets. These findings contribute to understanding the linguistic markers that contribute to constructive discourse online, informing moderation strategies and content analysis in digital environments.

2. Comparison of Stylistic Features Means between Constructive and Non-Constructive Comments in C3 and YNACC Datasets

The mean percentages of stylistic features for constructive and non-constructive comments provide further information on the constructiveness of comments. Table 6.2 compares the mean values of various stylistic features between non-constructive and constructive comments in the C3 and YNACC datasets. In both datasets, constructive comments generally show higher values for features related to punctuation, and lower values for exclamation.

In the C3 dataset, constructive comments exhibit notable differences in stylistic features compared to non-constructive ones. Specifically, constructive comments show higher mean values for *Punctuation Frequency* (2.9487 vs. 2.2946) and *Punctuation Type* (3.7851 vs. 2.0788). This suggests that constructive interactions tend to utilize a more varied and complex use of punctuation marks, potentially indicating a structured and thought-out expression of ideas. Conversely, features like *Exclamation Marks* are significantly lower in constructive comments (0.0320 vs. 0.0776), indicating a reduced emphasis on emotive language and a preference for a more formal tone.

Similarly, in the YNACC dataset, constructive comments display higher means for *Punctuation Type* (3.1207 vs. 2.8144) and *Stopwords* (9.0555 vs. 8.1592), albeit with smaller differences compared to the C3 dataset. This suggests that while constructive comments in YNACC also tend to punctuation and conversational elements, the distinctions between constructive and non-constructive discourse are more subtle.

We analyzed the constructiveness of comments in the C3 dataset through Figure 6.2, which provides a detailed visual representation of stylistic features and their correlation with constructive discourse. The bar plot contrasts the mean percentages of constructive comments against non-constructive comments. Notably, constructive comments exhibit higher percentages in features indicative of linguistic complexity and formality, such as *Punctuation Type* (64.5498%) and *Stopwords* (60.8677%), compared to non-constructive comments. Conversely, non-constructive comments display elevated percentages of *Exclamation Marks* and *Usermention.*

Additionally, a correlation line plot (in black) overlays the bar plot, aligning with data from Table 6.1. This plot underscores the strong positive correlations of features like *Punctuation Type* (correlation coefficient: 0.4770) and *Stopwords* (correlation coefficient: 0.3255) with constructiveness. These findings suggest that constructive comments utilize diverse punctuation marks and contain more conversational elements, contributing to a structured and engaging discourse. Conversely, features like *Exclamation Marks* exhibit a negative correlation

Stylistic Feature	Correlation (C3)	Correlation (YNACC)
Quotes	0.0524	-0.0132
Punctuation Frequency (Punc_Freq)	0.0989	-0.0050
Punctuation Types (Punc_Type)	0.4770	0.0810
Exclamation Marks	-0.1404	-0.0363
Stopwords	0.3255	0.0727
Camelcase	-0.0037	-0.0084
Negation	-0.0893	0.0115
Proper Nouns (Propn)	0.0024	-0.0359
Usermention	-0.0107	-0.0065
Hashtags	-0.0135	-0.0291
Misspelled Words (Misspel)	0.0897	-0.0255
Out of Vocabulary (OutofVocab)	0.2646	0.0471
Noun	0.2295	0.0400
Past Tense Words (Past_Tense)	0.0822	0.0334
Verbs	0.2668	0.0626
Interrogative Words	-0.0501	-0.0109

Table 6.1: Stylistic Features and Correlations with constructiveness in C3 and YNACC datasets

Dataset	C3			YNACC		
Stylistic Feature	Non-Con Mean	Con Mean	Con Mean %	Non-Con Mean	Con Mean	Con Mean %
Quotes	0.0018	0.0067	79.0289	0.1962	0.1833	48.2972
Punctuation Frequency	2.2946	2.9487	56.2372	2.4822	2.4554	49.7285
Punctuation Type	2.0788	3.7851	64.5497	2.8144	3.1207	52.5803
Exclamation Marks	0.0776	0.0320	4.8732	0.0804	0.0542	40.2605
Stopwords	5.8323	9.4758	60.8677	8.1592	9.0555	52.6035
Camelcase	0.0272	0.0340	47.1634	0.0055	0.0042	43.6571
Negation	0.1512	0.2255	30.7073	0.0812	0.0887	52.2087
Proper Nouns	1.2425	1.4449	50.1866	0.7237	0.6296	46.5251
Usermention	0.0012	0.0008	21.9080	0.0140	0.0125	47.1903
Hashtags	0.0036	0.0019	29.6176	0.0150	0.0084	35.8106
Misspelled Words	1.3450	1.9789	57.7360	1.0631	0.9931	48.2990
OutofVocab	5.3914	8.2362	60.0019	5.5637	5.9535	51.6924
Noun	3.8448	5.6854	60.1613	3.1483	3.3584	51.6141
Past Tense Words	0.4524	0.8129	58.0017	0.3899	0.4451	53.3045
Verbs	2.2732	3.5511	60.2658	2.3671	2.6334	52.6628
Interrogation Words	0.2552	0.3935	42.5088	0.1554	0.1461	48.4580

Table 6.2: Stylistic Features, Non-Constructive Mean, Constructive Mean, and Constructive Mean % in C3 and YNACC datasets

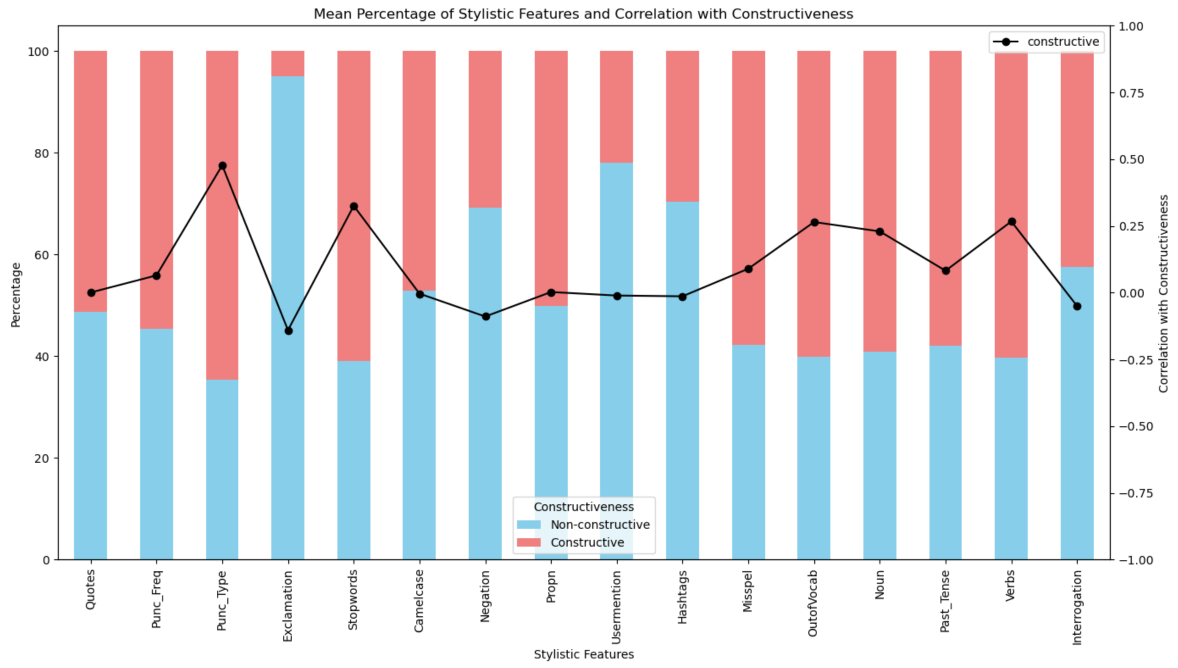


Figure 6.2: Mean percentage of stylistic features and their correlation with constructiveness for the C3 dataset

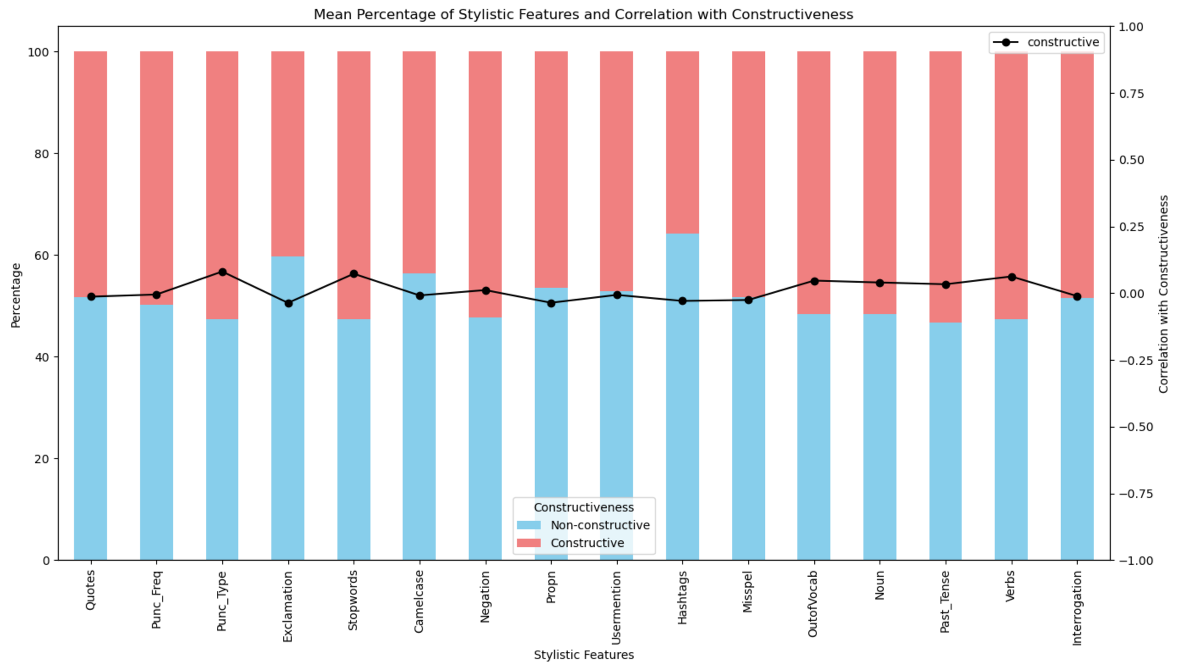


Figure 6.3: Mean percentage of stylistic features and their correlation with constructiveness for the YNACC dataset

(-0.1404), indicating their association with non-constructive expressions.

We also explored the stylistic features and their implications for constructiveness within the YNACC dataset using Figure 6.3. This visual representation features a bar plot contrasting mean percentages of constructive comments versus non-constructive comments. Constructive comments in the YNACC dataset exhibit higher percentages in certain stylistic features associated with structured and formal discourse. Specifically, they show elevated percentages in *Punctuation Type* (52.5803%) and *Stopwords* (52.6035%), suggesting a deliberate use of punctuation variety and conversational elements to foster constructive interactions online.

The correlation line plot overlaid on the bar plot aligns with the data from Table 6.1, highlighting key correlations between stylistic features and constructiveness. Features like *Punctuation Type* demonstrate a positive correlation (correlation coefficient: 0.0810), indicating their association with constructive comments, although not as noticeable as in the C3 dataset. Conversely, features such as *Exclamation Marks* and *Usermention* exhibit lower percentages in constructive comments, emphasizing their prevalence in non-constructive discourse.

In summary, these analyses underscore the nuanced distinctions in stylistic features contributing to constructive communication in both the C3 and YNACC datasets. The findings emphasize the importance of punctuation diversity and language formality in fostering positive interactions and meaningful discussions online, offering valuable insights for promoting constructive dialogue across digital platforms.

6.2.2 Complexity Features

1. Correlations between Complexity Features and Constructiveness in C3 and YNACC Datasets

Table 6.3 provides insights into the correlations between various complexity features and the constructiveness of comments in the C3 and YNACC datasets. These correlations shed light on how different linguistic complexities contribute to the perceived quality of online discourse.

Among the complexity features, *word count* shows a strong positive correlation with constructiveness in the C3 dataset (0.6005) and a weaker positive correlation in YNACC (0.0834). This suggests that longer comments tend to be perceived as more constructive, potentially indicating thorough explanations or detailed contributions to discussions.

About *mean word length*, correlations are positive but relatively modest, with values of 0.0533 in C3 and 0.0329 in YNACC. This indicates that comments with longer average word lengths may contribute slightly more to constructive discussions, possibly reflecting clarity or depth in expression.

On the other hand, *TTR* shows a negative correlation with constructiveness in both datasets (-0.3655 in C3 and -0.1083 in YNACC). A lower *TTR* suggests greater lexical diversity, which could imply that comments with more varied vocabulary are perceived as less constructive, possibly due to complexity or difficulty in comprehension.

Lastly, *MTLD* (Measure of Textual Lexical Diversity) exhibits a positive correlation with constructiveness, notably better in the C3 dataset (0.4843) and weaker in YNACC (0.0903). A higher *MTLD* value indicates greater lexical diversity and sophistication in language use, potentially contributing positively to the perceived quality of constructive comments.

These correlations underscore how different linguistic complexities influence the constructiveness of online comments across varied datasets. Understanding these nuances can inform strategies for fostering constructive discourse and content moderation in digital environments.

2. Comparison of Complexity Features Means between Constructive and Non-Constructive Comments in C3 and YNACC Datasets

The mean percentages of complexity features shed light on the nuanced differences between constructive and non-constructive comments across the C3 and YNACC datasets. Table 6.4 compares the mean values of key complexity features, highlighting their association with comment constructiveness.

In the C3 dataset, constructive comments demonstrate significant differences in complexity features compared to non-constructive ones. For instance, constructive comments exhibit notably higher mean values for *word count* (124.8883 vs. 27.2284) and *MTLD* (52.9992 vs. 9.2425). This indicates that constructive interactions tend to involve longer and more lexically diverse comments, reflecting a deeper engagement and elaboration of ideas. Conversely, features like *TTR* (0.0017 vs. 0.2257) show lower mean values in constructive comments, suggesting less lexical diversity and potentially more focused discussions.

Similarly, in the YNACC dataset, constructive comments also display higher mean values for *word count* (58.2053 vs. 44.4595) and *MTLD* (25.0510 vs. 18.0852), albeit with smaller differences compared to C3. This indicates that constructive comments in YNACC tend to be longer and more lexically diverse than non-constructive comments, fostering richer content and deeper engagement.

Complexity Feature	**Correlation (C3)**	**Correlation (YNACC)**
word_count	0.6005	0.0834
mean_word_length	0.0533	0.0329
ttr	-0.3655	-0.1083
mtld	0.4843	0.0903

Table 6.3: Complexity Features and Correlations with constructiveness in C3 and YNACC datasets

Dataset	**C3**			**YNACC**		
Complexity Feature	**Non-Con Mean**	**Con Mean**	**Con Mean %**	**Non-Con Mean**	**Con Mean**	**Con Mean %**
word_count	27.2284	124.8883	82.1003	44.4595	58.2053	56.6945
mean_word_length	3.8846	4.3400	50.6914	3.3537	3.3976	50.3253
ttr	0.2257	0.0017	0.7423	0.2732	0.1825	40.0482
mtld	9.2425	52.9992	85.1506	18.0852	25.0510	58.0741

Table 6.4: Complexity Features, Non-Constructive Mean, Constructive Mean, and Constructive Mean % in C3 and YNACC datasets

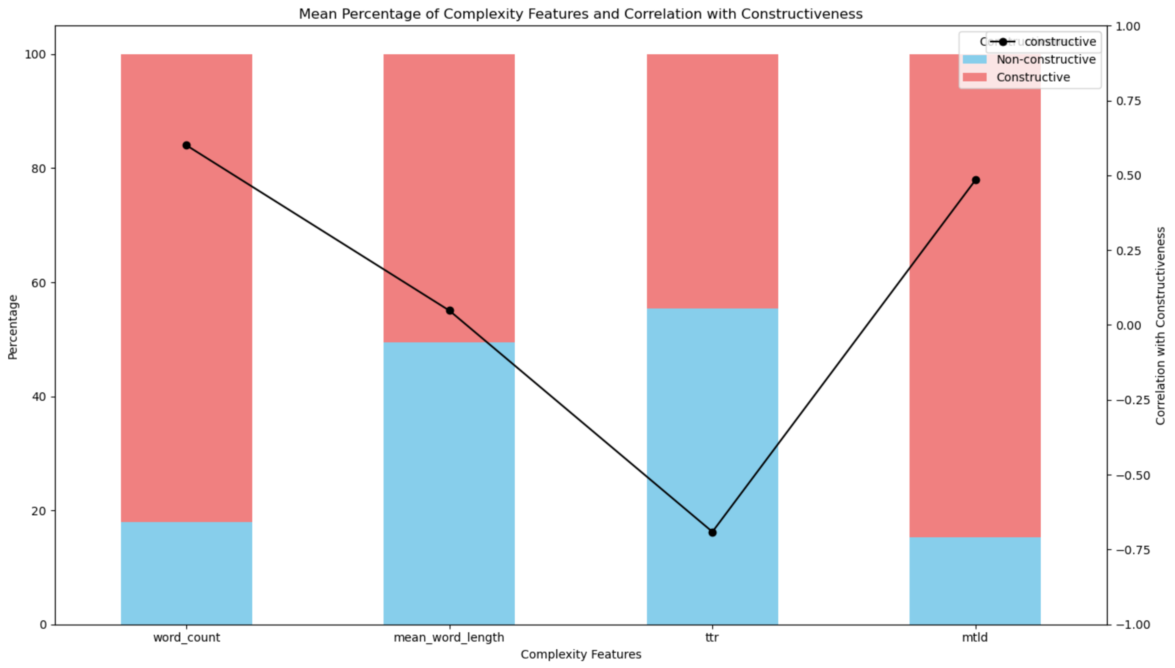


Figure 6.4: Mean percentage of complexity features and their correlation with constructiveness for the C3 dataset

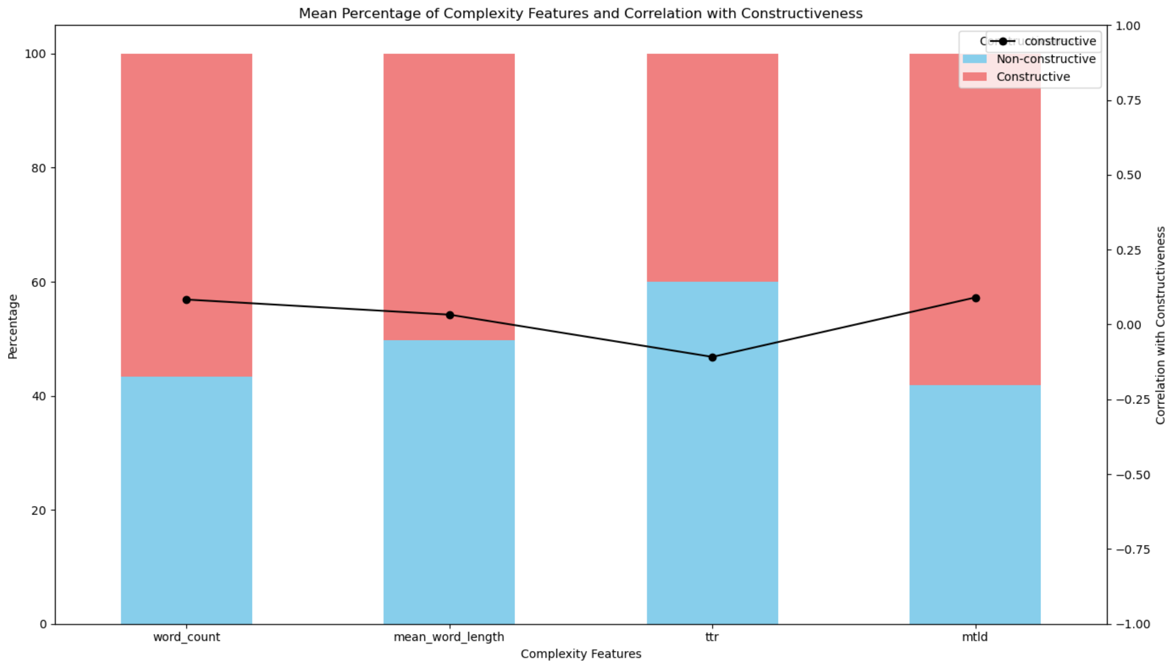


Figure 6.5: Mean percentage of complexity features and their correlation with constructiveness for the YNACC dataset

Figures 6.4 and 6.5 visually represent these insights. The bar plots contrast mean percentages of constructive comments versus non-constructive comments, illustrating how constructive comments consistently exhibit higher percentages in complexity features across both datasets. These features are critical indicators of constructive discourse, reflecting the depth and complexity of content that contributes positively to online discussions.

Moreover, the correlation line plots overlaid on the bar plots align with the data from Table 6.3, emphasizing key correlations between complexity features and constructiveness. Features like *word_count* and *MTLD* demonstrate positive correlations with constructiveness, underscoring their role in fostering detailed and insightful contributions. Conversely, features such as *TTR* show negative correlations, indicating their association with non-constructive discourse patterns.

In summary, these analyses highlight how complexity features influence the perceived constructiveness of comments in digital environments. Understanding these dynamics provides valuable insights for fostering meaningful interactions and developing effective moderation strategies across online platforms.

6.2.3 Psychological Features

1. Correlations between Psychological Features and Constructiveness in C3 and YNACC Datasets

Table 6.5 provides insights into the correlations between various psychological features and the constructiveness of comments in the C3 and YNACC datasets. These correlations shed light on how different psychological aspects contribute to the perception of constructiveness in online discourse.

The analysis reveals that *sentiment score* shows a positive correlation with constructiveness, albeit relatively small, in both the C3 dataset (0.0611) and the YNACC dataset (0.0108). This suggests that comments with more positive sentiments tend to be associated with constructive contributions. Positive sentiment may indicate a supportive or respectful tone, fostering a conducive environment for meaningful dialogue and engagement.

These findings indicate that psychological features, such as sentiment analysis, play a nuanced role in shaping how comments are perceived in terms of their constructiveness across different datasets. While the correlations for sentiment are modest compared to other stylistic or content-based features, they underscore the significance of emotional tone in online interactions. Understanding these psychological markers can inform moderation strategies and content analysis efforts to promote constructive discourse and mitigate potential conflicts in digital environments.

Psychological Feature	**Correlation (C3)**	**Correlation (YNACC)**
sentiment_score	0.0611	0.0108

Table 6.5: Psychological Features and Correlations with constructiveness in C3 and YNACC datasets

2. Comparison of Psychological Features Means between Constructive and Non-Constructive Comments in C3 and YNACC Datasets

The mean percentages of psychological features for constructive and non-constructive comments provide further insights into the constructiveness of comments. Table 6.6 compares the mean values of various psychological features between non-constructive and constructive comments in the C3 and YNACC datasets. In both datasets, constructive comments generally show higher values for sentiment score, indicating a more positive sentiment.

In the C3 dataset, constructive comments exhibit notable differences in psychological features compared to non-constructive ones. Specifically, constructive comments show higher mean values for *Sentiment Score* (0.2769 vs. 0.0838). This suggests that constructive interactions tend to carry a more positive sentiment, potentially indicating a more respectful and supportive tone. Conversely, non-constructive comments exhibit lower sentiment scores, which may reflect a more negative tone.

Similarly, in the YNACC dataset, constructive comments display higher means for *Sentiment Score* (0.0770 vs. 0.0533), although the differences are smaller in the C3 dataset. This indicates that while constructive comments in YNACC exhibit a more positive sentiment, the distinctions between constructive and non-constructive discourse are more subtle.

We analyzed the constructiveness of comments in the C3 dataset through Figure 6.6, which provides a detailed visual representation of psychological features and their correlation with constructive discourse. The bar plot contrasts the mean percentages of constructive comments against non-constructive comments. Notably, constructive comments exhibit higher percentages in features indicative of positive sentiments, such as *Sentiment Score* (76.7610%), compared to non-constructive comments.

Dataset	**C3**			**YNACC**		
Psychological Feature	**Non-Con Mean**	**Con Mean**	**Con Mean %**	**Non-Con Mean**	**Con Mean**	**Con Mean %**
sentiment_score	0.0838	0.2769	76.7610	0.0533	0.0770	59.1233

Table 6.6: Psychological Features, Non-Constructive Mean, Constructive Mean, and Constructive Mean % in C3 and YNACC datasets

Additionally, a correlation line plot (in black) overlays the bar plot, aligning with data from Table 6.5. This plot underscores the positive correlations of features like *Sentiment Score* (correlation coefficient: 0.0611) with constructiveness. These findings suggest that constructive comments tend to carry a more positive sentiment, contributing to a supportive and respectful discourse. Conversely, features with lower sentiment scores are associated with non-constructive expressions.

We also explored the psychological features and their implications for constructiveness within the YNACC dataset using Figure 6.7. This visual representation features a bar plot contrasting mean percentages of constructive comments versus non-constructive comments. Construc-

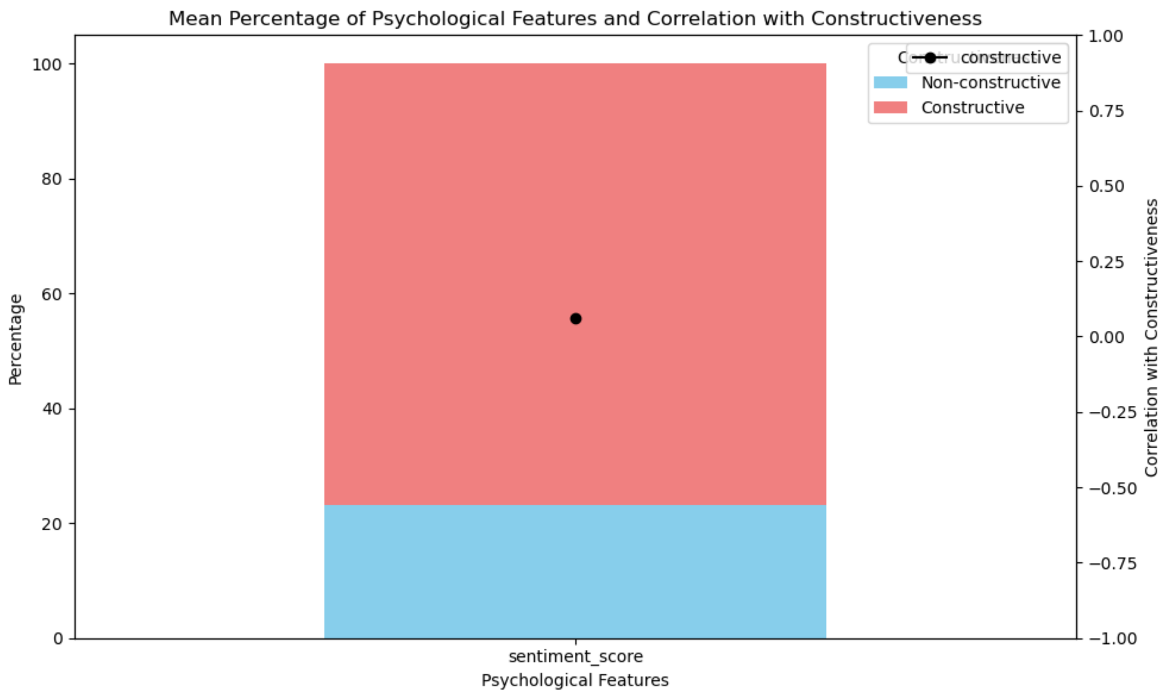


Figure 6.6: Mean percentage of psychological features and their correlation with constructiveness for the C3 dataset

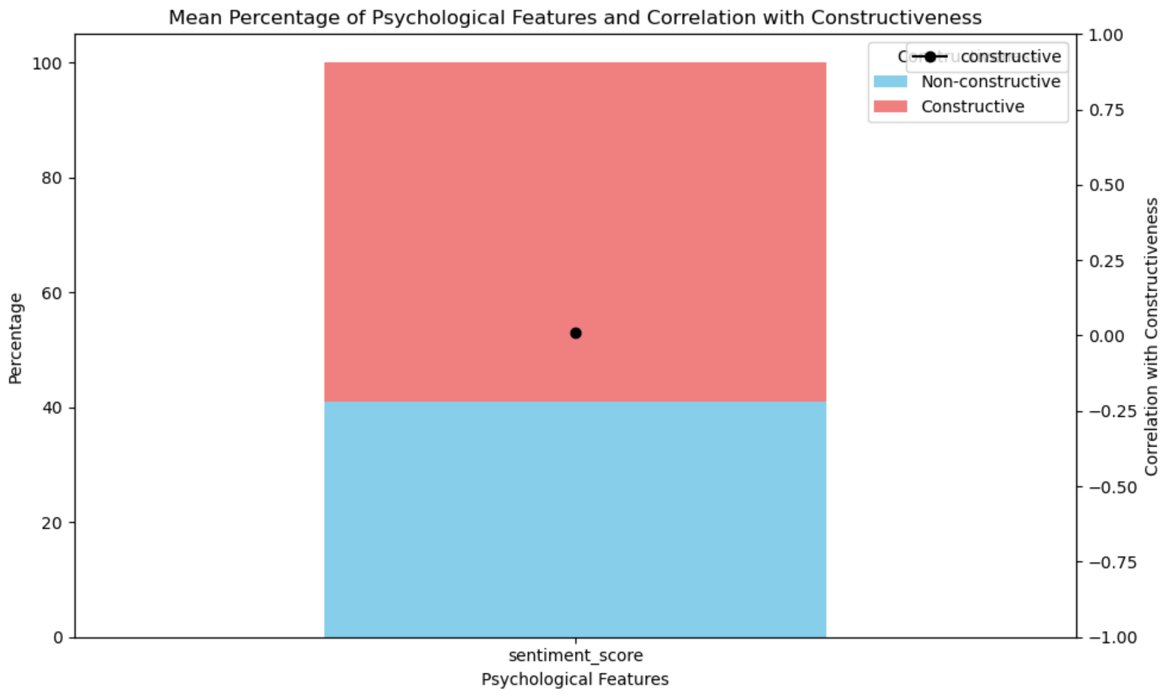


Figure 6.7: Mean percentage of psychological features and their correlation with constructiveness for the YNACC dataset

tive comments in the YNACC dataset exhibit higher percentages in psychological features associated with positive sentiment. Specifically, they show elevated percentages in the *Sentiment Score* (59.1233%), suggesting a positive tone to foster constructive interactions online.

The correlation line plot overlaid on the bar plot aligns with the data from Table 6.5, highlighting key correlations between psychological features and constructiveness. Features like *Sentiment Score* demonstrate a positive correlation (correlation coefficient: 0.0108), indicating their association with constructive comments, albeit less pronounced in the C3 dataset.

In summary, these analyses underscore the distinctions in psychological features contributing to constructive communication in both the C3 and YNACC datasets. The findings emphasize the importance of positive sentiment in fostering respectful and supportive interactions and meaningful discussions online, offering valuable insights for promoting constructive dialogue across digital platforms.

6.2.4 Emotional Features

1. Correlations between Emotional Features and Constructiveness in C3 and YNACC Datasets

Table 6.7 provides insights into the correlations between various emotional features and the constructiveness of comments in the C3 and YNACC datasets. These correlations highlight how different emotional elements contribute differently to the constructiveness of online discourse. Notably, *joy* shows a strong positive correlation with constructiveness in the C3 dataset (0.1475), indicating that comments expressing joy are more likely to be perceived as constructive. Similarly, *trust* also exhibits a positive correlation in the C3 dataset (0.1230), suggesting that comments that convey trustworthiness are likely to be constructive.

In the YNACC dataset, however, the correlations for emotional features are generally weaker and sometimes negative. For instance, *anger* shows a slight negative correlation with constructiveness (-0.0056), implying that comments expressing anger are less likely to be constructive. Similarly, *disgust* also has a negative correlation (-0.0103), further indicating that negative emotions are associated with non-constructive comments in the YNACC dataset.

Feature	Correlation (C3)	Correlation (YNACC)
fear	0.1072	0.0019
anger	0.0977	-0.0056
trust	0.1230	-0.0037
surprise	0.0451	-0.0029
positive	0.0766	-0.0011
negative	0.0233	0.0017
sadness	0.0846	0.0088
disgust	0.0422	-0.0103
joy	0.1475	-0.0051
anticipation	-0.3613	-0.0225

Table 6.7: Emotion Features and Correlations with constructiveness in C3 and YNACC Dataset

Conversely, features such as *anticipation* exhibit a negative correlation with constructiveness in both datasets (-0.3613 in C3 and -0.0225 in YNACC). This suggests that comments expressing anticipation may lack the immediacy or relevance required for constructiveness.

Overall, these correlations underscore the importance of emotional features in influencing how comments are perceived in terms of their constructiveness across different datasets. Positive emotions like joy and trust are associated with constructive comments, whereas negative emotions like anger and disgust are linked to non-constructive comments. These findings contribute to understanding the emotional markers that contribute to constructive discourse online, informing moderation strategies and content analysis in digital environments.

2. Comparison of Emotion Features Means between Constructive and Non-Constructive Comments in C3 and YNACC datasets

The mean percentages of emotional features for constructive and non-constructive comments provide further information on the constructiveness of comments. Table 6.8 compares the mean values of various emotional features between non-constructive and constructive comments in the C3 and YNACC datasets. In both datasets, constructive comments generally show higher values for features related to positive emotions and lower values for negative emotions.

In the C3 dataset, constructive comments exhibit notable differences in emotional features compared to non-constructive ones. Specifically, constructive comments show higher mean values for *trust* (0.1424 vs. 0.1088) and *joy* (0.0629 vs. 0.0427). This suggests constructive interactions express more trust and joy, potentially indicating a positive and supportive tone. Conversely, features like *anger* (0.0686 vs. 0.0532) and *sadness* (0.0690 vs. 0.0553) are significantly higher in non-constructive comments, pointing to a more negative emotional expression.

Dataset	**C3**			**YNACC**		
Feature	**Non-Con Mean**	**Con Mean**	**Con Mean %**	**Non-Con Mean**	**Con Mean**	**Con Mean %**
fear	0.0655	0.0868	57.0036	0.0656	0.0661	50.1566
anger	0.0532	0.0686	56.3220	0.0558	0.0548	49.5391
trust	0.1088	0.1424	56.6917	0.1061	0.1050	49.7238
surprise	0.0320	0.0382	54.3977	0.0327	0.0322	49.6002
positive	0.1940	0.2235	53.5232	0.1730	0.1725	49.9320
negative	0.1622	0.1695	51.1122	0.1590	0.1597	50.1019
sadness	0.0553	0.0690	55.5364	0.0596	0.0613	50.6835
disgust	0.0371	0.0428	53.5568	0.0460	0.0441	48.9633
joy	0.0427	0.0629	59.5696	0.0509	0.0500	49.5802
anticipation	0.2215	0.1151	34.1977	0.0761	0.0700	47.9060

Table 6.8: Emotion Features, Non-Constructive Mean, Constructive Mean, and Constructive Mean % in C3 and YNACC datasets

Similarly, in the YNACC dataset, constructive comments display higher means for *positive* (0.1725 vs. 0.1730) and *joy* (0.0500 vs. 0.0509), albeit with smaller differences compared to the C3 dataset. This suggests that while constructive comments in YNACC also exhibit a preference for positive emotions, the distinctions between constructive and non-constructive discourse are more subtle. Non-constructive comments in the YNACC dataset show slightly higher values for *anger* (0.0558 vs. 0.0548) and *sadness* (0.0596 vs. 0.0613), indicating a prevalence of negative emotions.

We analyzed the constructiveness of comments in the C3 dataset through Figure 6.8, which provides a detailed visual representation of emotional features and their correlation with constructive discourse. The bar plot contrasts the mean percentages of constructive comments against non-constructive comments. Notably, constructive comments exhibit higher percentages in features indicative of positive emotions, such as *joy* (59.5696%) and *trust* (56.6917%), compared to non-constructive comments. Conversely, non-constructive comments display elevated percentages of negative emotions like *anger* and *sadness.*

Additionally, a correlation line plot (in black) overlays the bar plot, aligning with data from Table 6.7. This plot underscores the strong positive correlations of features like *joy* (correlation coefficient: 0.1475) and *trust* (correlation coefficient: 0.1230) with constructiveness. These findings suggest that constructive comments express more positive emotions, contributing to a supportive and engaging discourse. Conversely, features like *anger* exhibit a negative correlation (-0.0056), indicating their association with non-constructive expressions.

We also explored the emotional features and their implications for constructiveness within the YNACC dataset using Figure 6.9. This visual representation features a bar plot contrasting mean percentages of constructive comments versus non-constructive comments. Constructive comments in the YNACC dataset exhibit higher percentages in certain emotional features associated with positive and supportive discourse. Specifically, they show elevated percentages in *joy* (49.5802%) and *trust* (49.7238%), suggesting a deliberate use of positive emotions to foster constructive interactions online.

The correlation line plot overlaid on the bar plot aligns with the data from Table 6.7, highlighting key correlations between emotional features and constructiveness. Features like *joy* demonstrate a positive correlation (correlation coefficient: 0.1475), indicating their association with constructive comments, albeit less pronounced in the C3 dataset. Conversely, features such as *anger* and *sadness* exhibit lower percentages in constructive comments, emphasizing their prevalence in non-constructive discourse.

In summary, these analyses underscored the nuanced distinctions in emotional features contributing to constructive communication in both the C3 and YNACC datasets. The findings emphasize the importance of positive emotional expressions in fostering positive interactions and meaningful discussions online, offering valuable insights for promoting constructive dialogue across digital platforms.

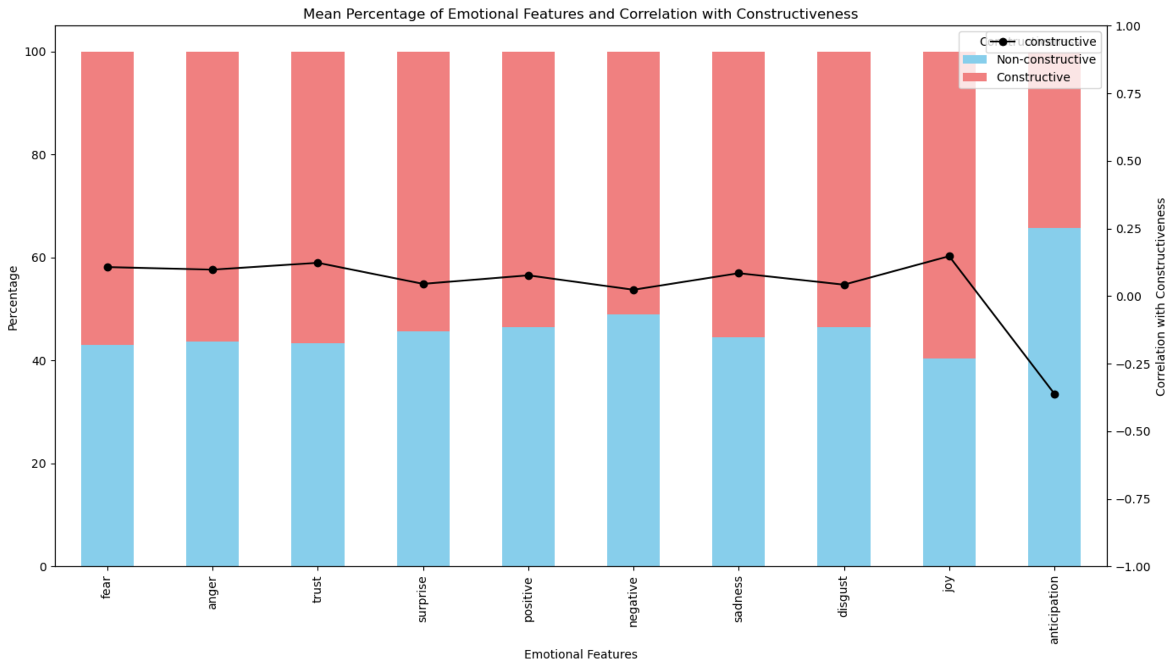


Figure 6.8: Mean percentage of emotional features and their correlation with constructiveness for the C3 dataset

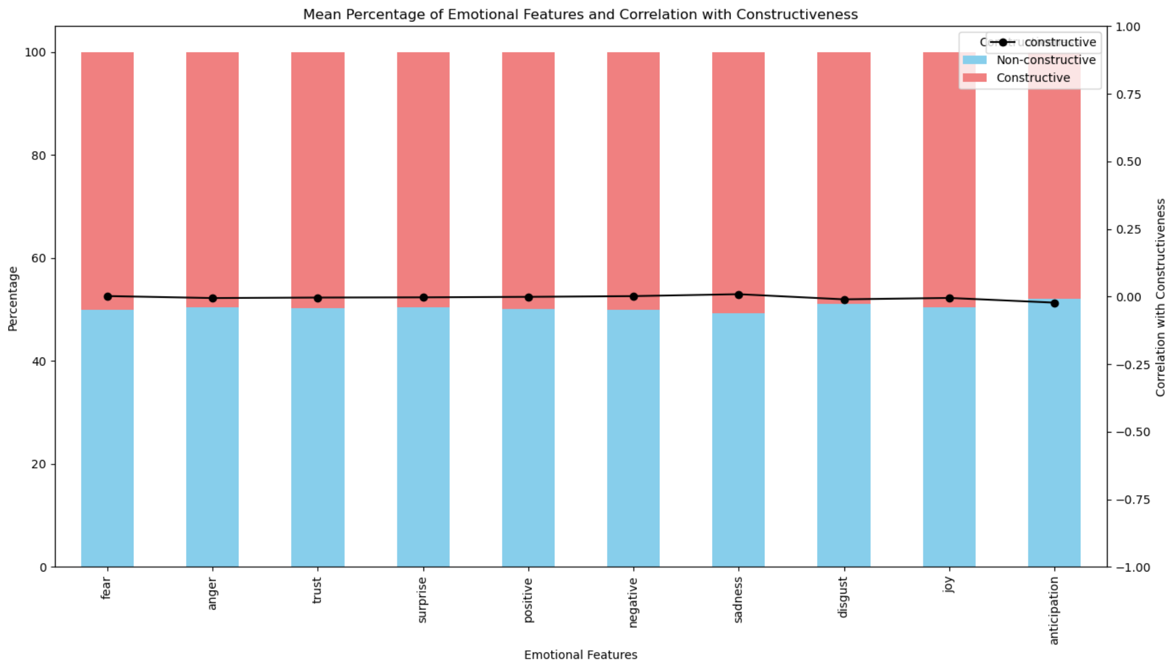


Figure 6.9: Mean percentage of emotional features and their correlation with constructiveness for the YNACC dataset

6.2.5 POS Tag Features

1. Correlations between Emotional Features and Constructiveness in C3 and YNACC Datasets

Table 6.9 provides insights into the correlations between various POS tag features and the constructiveness of comments in the C3 and YNACC datasets. The correlations highlight how different grammatical elements contribute differently to the constructiveness of online discourse. Notably, prepositions and subordinating conjunctions (*IN*) show a strong positive correlation with constructiveness in the C3 dataset (0.3248) and a moderate positive correlation in YNACC (0.0881). This suggests that comments rich in prepositions and conjunctions are more likely to be perceived as constructive, indicating a structured and coherent expression of ideas. Similarly, features like determiners (*DT*) exhibit positive correlations in both datasets (0.2863 in C3 and 0.0767 in YNACC), indicating their role in clarifying and specifying ideas in constructive comments.

Conversely, features such as interjections (*UH*), possessive ending (*POS*), and symbols (*SYM*) show negligible or negative correlations with constructiveness, particularly in the YNACC dataset (-0.0061, 0.0115, and -0.0088, respectively). This suggests that these grammatical elements are less prevalent in constructive comments, potentially indicating that highly emotional or informal expressions (marked by interjections) and symbolic notations are not conducive to substantive contributions to the discussion.

Overall, the correlations underscore the importance of POS tag features in influencing how comments are perceived in terms of their constructiveness across different datasets. These findings contribute to understanding the grammatical markers that contribute to constructive discourse online, informing moderation strategies and content analysis in digital environments. Specifically, the positive correlations of tags like *IN*, *DT*, and nouns (*NN*) with constructiveness suggest that comments using more structured and informative language are seen as more constructive. Conversely, the negative or negligible correlations of less formal tags like *UH* and *SYM* highlight their association with non-constructive discourse.

In summary, analyzing POS tag features offers valuable insights into the linguistic components that foster constructive online communication, aiding in the development of better moderation tools and strategies to enhance the quality of online discourse.

2. Comparison of Stylistic Features Means between Constructive and Non-Constructive Comments in C3 and YNACC Datasets

The mean percentages of POS tag features for constructive and non-constructive comments provide further information on the constructiveness of comments. Table 6.10 compares the mean values of various POS tag features between non-constructive and constructive comments in the C3 and YNACC datasets. In both datasets, constructive comments generally show higher values for features related to grammatical structure, such as conjunctions, determiners, and nouns, and lower values for features like symbols and interjections.

In the C3 dataset, constructive comments exhibit notable differences in POS tag features compared to non-constructive ones. Specifically, constructive comments show higher mean

POS Tag	Correlation (C3)	Correlation (YNACC)
CC	0.2849	0.0629
CD	0.0830	0.0483
DT	0.2863	0.0767
EX	0.0699	0.0354
FW	0.0238	0.0045
IN	0.3248	0.0881
JJ	0.2622	0.0413
JJR	0.1036	0.0323
JJS	0.0611	0.0360
LS	0.0169	0.0024
MD	0.1201	0.0464
NN	0.2691	0.0640
NNS	0.2893	0.0675
NNP	0.0525	-0.0276
NNPS	0.0699	0.0067
PDT	0.0291	0.0166
POS	-0.0026	0.0115
PRP	0.1876	0.0356
PRP$	0.1522	0.0122
RB	0.2291	0.0603
RBR	0.0723	0.0145
RBS	0.0351	0.0001
RP	0.0587	0.0136
SYM	0.0068	-0.0088
TO	0.2242	0.0398
UH	-0.0024	-0.0061
VB	0.2021	0.0396
VBD	0.1568	0.0417
VBG	0.1678	0.0369
VBN	0.2199	0.0607
VBP	0.1870	0.0346
VBZ	0.1644	0.0416
WDT	0.1536	0.0494
WP	0.0930	-0.0031
WP$	0.0290	0.0094
WRB	0.0646	0.0170

Table 6.9: POS tag Features and Correlations with constructiveness in C3 and YNACC datasets

Dataset	**C3**			**YNACC**		
POS Tag	**Non-Con Mean**	**Con Mean**	**Con Mean %**	**Non-Con Mean**	**Con Mean**	**Con Mean %**
CC	0.3947	0.7662	65.9988	0.4049	0.4862	54.5661
CD	0.1179	0.2316	66.2555	0.1319	0.1714	56.5100
DT	1.3647	2.1250	60.8929	1.1522	1.3303	53.5875
EX	0.0260	0.0443	62.9939	0.0309	0.0414	57.2222
FW	0.0048	0.0082	63.2119	0.0048	0.0054	52.9938
IN	1.3586	2.4345	64.1827	1.2257	1.4593	54.3493
JJ	0.9665	1.6565	63.1526	0.8662	0.9474	52.2369
JJR	0.0432	0.0812	65.2838	0.0404	0.0515	55.9913
JJS	0.0369	0.0560	60.2907	0.0247	0.0347	58.4136
LS	0.0001	0.0003	83.4314	0.0002	0.0002	54.9931
MD	0.2212	0.3151	58.7562	0.2220	0.2589	53.8295
NN	1.9538	3.0770	61.1634	1.8068	2.0239	52.8332
NNS	0.6675	1.2481	65.1556	0.6463	0.7688	54.3282
NNP	1.1907	1.3609	53.3356	0.8632	0.7927	47.8708
NNPS	0.0428	0.0699	62.0284	0.0201	0.0216	51.8737
PDT	0.0158	0.0214	57.5680	0.0130	0.0160	55.2262
POS	0.0001	0.0001	41.3758	0.0462	0.0502	52.1032
PRP	0.6018	0.8799	59.3827	1.0004	1.0711	51.7061
PRP$	0.2277	0.3595	61.2218	0.2494	0.2596	51.0057
RB	0.7028	1.0893	60.7833	0.8480	0.9677	53.2963
RBR	0.0241	0.0428	64.0214	0.0196	0.0230	54.0362
RBS	0.0077	0.0126	62.1988	0.0059	0.0059	50.0514
RP	0.0619	0.0845	57.7114	0.0724	0.0782	51.9469
SYM	0.0012	0.0018	59.2432	0.0008	0.0005	38.1314
TO	0.3449	0.6017	63.5675	0.3249	0.3658	52.9579
UH	0.0125	0.0121	49.2434	0.0176	0.0163	48.1287
VB	0.6180	0.9345	60.1928	0.6961	0.7609	52.2266
VBD	0.2292	0.3911	63.0493	0.3167	0.3676	53.7152
VBG	0.2866	0.4588	61.5482	0.2715	0.3071	53.0739
VBN	0.2271	0.4295	65.4143	0.2242	0.2780	55.3532
VBP	0.4324	0.6656	60.6209	0.5888	0.6396	52.0683
WDT	0.1536	0.0478	69.5649	0.0515	0.0713	58.0717
WP	0.0930	0.1056	59.2693	0.1153	0.1136	49.6401
WP$	0.0016	0.0042	72.5160	0.0006	0.0011	62.9824
WRB	0.104002	0.136124	56.688516	0.1081	0.1172	50.0151

Table 6.10: POS tag Features, Non-Constructive Mean, Constructive Mean, and Constructive Mean % in C3 and YNACC datasets

values for *IN* (2.4345 vs. 1.3586) and *DT* (determiners) (2.1250 vs. 1.3647). This suggests that constructive interactions utilize more structured and informative language, potentially indicating a more detailed and coherent expression of ideas. Conversely, features like *UH* are significantly lower in constructive comments (0.0121 vs. 0.0125), pointing to a reduced emphasis on emotive and informal language.

Similarly, in the YNACC dataset, constructive comments display higher means for *IN* (1.4593 vs. 1.2257) and *NN* (2.0239 vs. 1.8068), albeit with smaller differences compared to the C3 dataset. This suggests that while constructive comments in YNACC prefer structured grammatical elements and informative content, the distinctions between constructive and non-constructive discourse are more subtle.

We analyzed the constructiveness of comments in the C3 dataset through Figure 6.10, which provides a detailed visual representation of POS tag features and their correlation with constructive discourse. The bar plot contrasts the mean percentages of constructive comments against non-constructive comments. Notably, constructive comments exhibit higher percentages in features indicative of grammatical complexity and formality, such as *IN* (64.1827%) and *DT* (60.8929%), compared to non-constructive comments. Conversely, non-constructive comments display elevated percentages of *UH* and *SYM*.

Additionally, a correlation line plot (in black) overlays the bar plot, aligning with data from Table 6.9. This plot underscores the strong positive correlations of features like *IN* (correlation coefficient: 0.3248) and *DT* (correlation coefficient: 0.2863) with constructiveness. These findings suggest that constructive comments employ more structured grammatical elements, contributing to a detailed and engaging discourse. Conversely, features like *UH* exhibit a negligible correlation (-0.0024), indicating their association with non-constructive expressions.

We also explored the POS tag features and their implications for constructiveness within the YNACC dataset using Figure 6.11. This visual representation features a bar plot contrasting mean percentages of constructive comments versus non-constructive comments. Constructive comments in the YNACC dataset exhibit higher percentages in certain POS tag features associated with structured and formal discourse. Specifically, they show elevated percentages in *IN* (54.3493%) and *NN* (52.8332%), suggesting a deliberate use of prepositions, conjunctions, and nouns to foster constructive interactions online.

The correlation line plot overlaid on the bar plot aligns with the data from Table 6.9, highlighting key correlations between POS tag features and constructiveness. Features like *IN* demonstrate a positive correlation (correlation coefficient: 0.0881), indicating their association with constructive comments, albeit less pronounced in the C3 dataset. Conversely, features such as *UH* and *SYM* exhibit lower percentages in constructive comments, emphasizing their prevalence in non-constructive discourse.

In summary, these analyses underscored the nuanced distinctions in POS tag features that contribute to constructive communication in both the C3 and YNACC datasets. The findings emphasize the importance of grammatical complexity and language formality in fostering positive interactions and meaningful discussions online, offering valuable insights for promoting constructive dialogue across digital platforms.

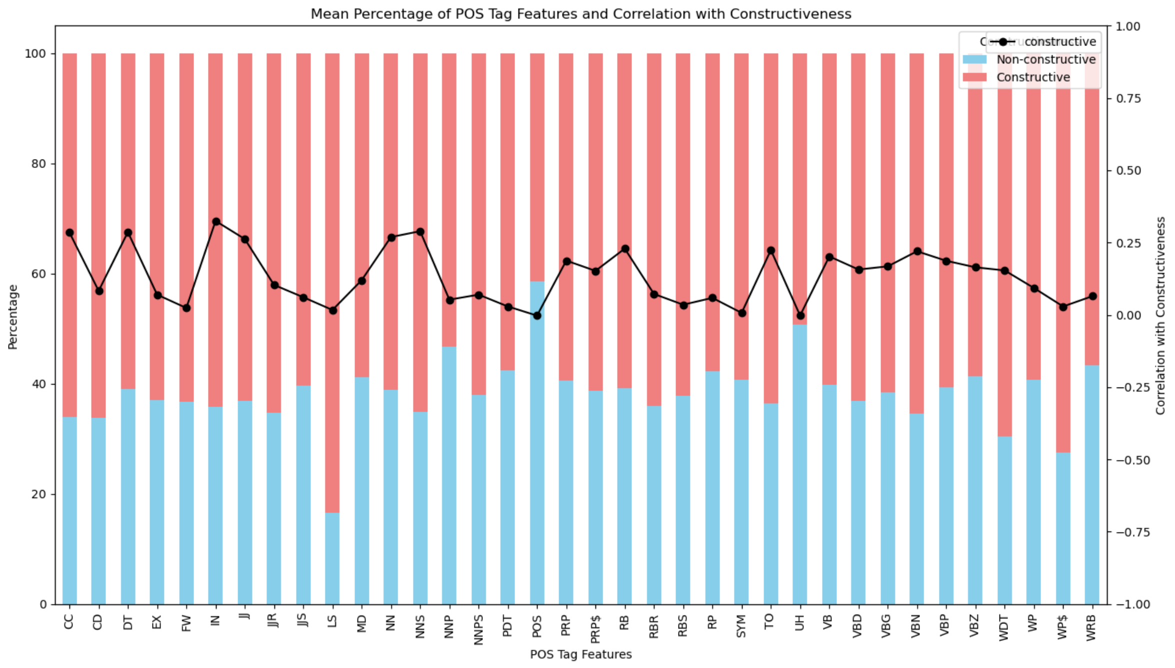


Figure 6.10: Mean percentage of POS tag features and their correlation with constructiveness for the C3 dataset

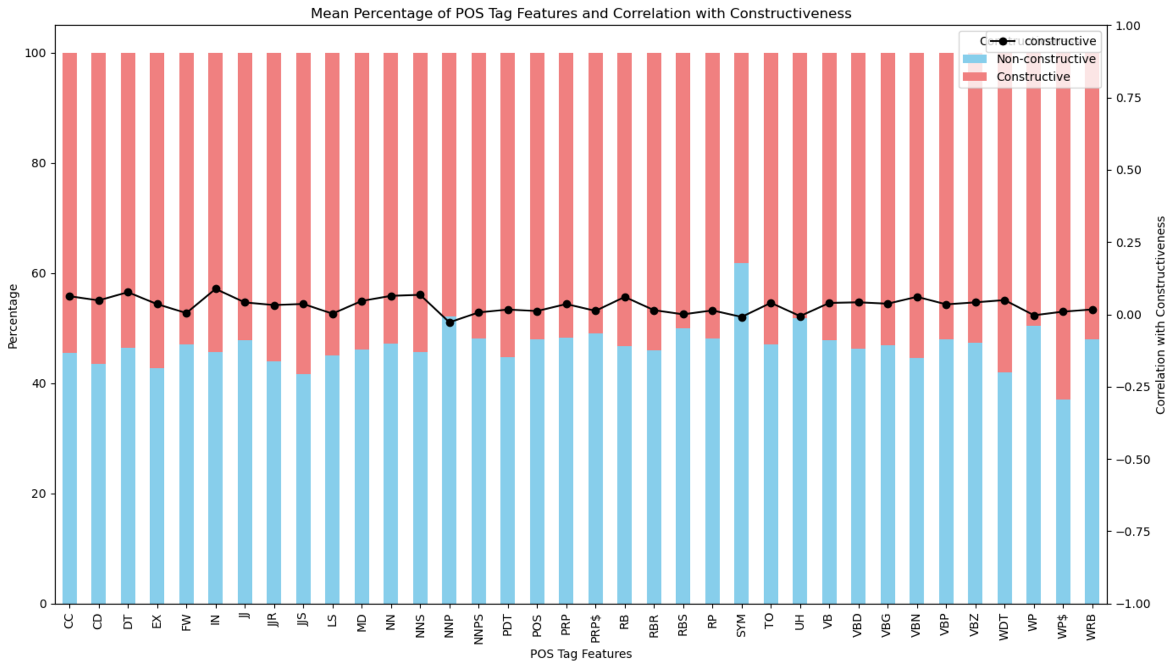


Figure 6.11: Mean percentage of POS tag features and their correlation with constructiveness for the YNACC dataset

6.3 Models and Hyperparameters Settings

We used several ML and DL models and evaluated them by tuning their hyperparameters to achieve optimal performance in predicting constructiveness. The hyperparameters for each model were systematically tested across a range of values to determine the best configuration. Table 6.11 summarizes the hyperparameter settings used for each model.

For ML models, we conducted experiments on LR, RF, SVM and KNN models. The LR model's hyperparameters are tuned with regularization parameters, penalty type, solver, maximum iterations, and class weights. The RF model's hyperparameters are tuned with the number of estimators, maximum features, maximum depth, minimum samples split, minimum samples leaf, and bootstrap method. The SVM model's hyperparameters are tuned with regularization parameters, kernel types, gamma values, degrees for the polynomial kernel, and independent terms in the kernel function. The KNN model's hyperparameters are tuned with the number of neighbors, weight function, distance metric, algorithm, leaf size, and power parameter for Minkowski distance.

For DL models, we conducted experiments on LSTM, BiLSTM, CNN, and GRU, we tuned various hyperparameters such as the number of units, dropout rates, batch size, and number of epochs with early stopping. All deep learning models used binary cross-entropy as the loss function and Adam as the optimizer.

The LSTM model consisted of two LSTM layers, where the first layer had a specified number of units and the second layer had half the number of units of the first layer, followed by a dropout layer after each LSTM layer. The final layer was a dense layer with a sigmoid activation function.

The BiLSTM model included two bidirectional LSTM layers, each followed by a dropout layer. Like the LSTM model, the second bidirectional LSTM layer had half the number of units as the first layer, and the final layer was a dense layer with a sigmoid activation function.

The CNN model started with a reshaping layer to ensure compatibility with Conv1D layers, followed by a Conv1D layer, a global max pooling layer, and two dense layers with dropout layers between them. The final layer was a dense layer with a sigmoid activation function.

The GRU model included two GRU layers, where the first GRU layer had a specified number of units and the second GRU layer had the same number of units, followed by a dropout layer after each GRU layer. The final layer was a dense layer with a sigmoid activation function.

The detailed settings for each model are provided in Table 6.11, offering a comprehensive overview of the hyperparameters tested. This systematic approach allows for a thorough evaluation of each model's performance under different configurations.

6.4 Evaluation

This section aimed to determine the most effective models by comparing their performances on two datasets, C3 and YNACC, using stylistic, complexity, psychological, emotional, POS Tag and all features. For LR, RF, and KNN models, we performed a grid search, while for

Model	Hyperparameters	Tuning Values
LR	Regularization (C)	0.1, 1, 10, 100
	Penalty	l1, l2, elasticnet, none
	Solver	newton-cg, lbfgs, liblinear, sag, saga
	Max Iterations	100, 200, 300
	Class Weights	balanced, None
RF	Number of Estimators	100, 200, 300
	Max. Features	sqrt, log2
	Max. Depth	None, 10, 20, 30
	Min. Samples Split	2, 5, 10
	Min. Samples Leaf	1, 2, 4
	Bootstrap	True, False
SVM	Regularization (C)	C 0.1, 1, 10
	Kernel	linear, rbf, poly
	Gamma	scale, auto
	Degree	2, 3, 4
	coef0	0.0, 0.1, 0.5
KNN	Number of Neighbors	3, 5, 7, 9, 11
	Weight Function	uniform, distance
	Distance Metric	euclidean, manhattan
	Algorithm	auto, ball tree, kd tree, brute force
	Leaf Size	10, 20, 30, 40, 50
	Minkowski Power Parameter	1, 2
LSTM	Units	32, 64, 128
	Dropout Rate	0.2, 0.3
	Batch Size	16, 32, 64
	Epochs	30
	Loss Function	binary cross-entropy
	Optimizer	Adam
BiLSTM	Units	32, 64, 128
	Dropout Rate	0.2, 0.3
	Batch Size	16, 32, 64
	Epochs	30
	Loss Function	binary cross-entropy
	Optimizer	Adam
CNN	Filters	32, 64, 128
	Kernel Size	3, 5
	Pool Size	2, 3
	Dropout Rate	0.2, 0.3
	Batch Size	16, 32, 64
	Epochs	30
	Loss Function	binary cross-entropy
	Optimizer	Adam
GRU	Units	32, 64, 128
	Dropout Rate	0.2, 0.3
	Batch Size	16, 32, 64
	Epochs	30
	Loss Function	binary cross-entropy
	Optimizer	Adam

Table 6.11: Hyperparameters for Various Models

the other models, we used randomized search with a 5-fold cross-validation to identify the optimal hyperparameters. We then compared the models' performance based on accuracy and F1 score.

6.4.1 Performance of Stylistic Features

Table 6.12 presents a comparative analysis of various models applied to stylistic features extracted from two datasets, C3 and YNACC, using different hyperparameters. Each model's performance is evaluated based on the F1 score and accuracy metrics.

For the C3 dataset: The LR model, with hyperparameters *C=0.1, max_iter=100, penalty='l2', solver='lbfgs'*, achieved an F1 score of 0.8044 and an accuracy of 0.7875. The RF model, with hyperparameters *bootstrap=True, max_depth=None, max_features='sqrt', min_samples_leaf=1, min_samples_split=5, n_estimators=300*, demonstrated superior performance with an F1 score of 0.8942 and an accuracy of 0.8833, making it the best-performing model for this dataset. The SVM model, with hyperparameters *kernel='poly', gamma='scale', degree=3, coef0=0.5, C=10*, achieved an F1 score of 0.8592 and an accuracy of 0.8433. The KNN model, with algorithm='auto', leaf_size=10, metric='manhattan', n_neighbors=11, p=1, weights='distance', showed strong performance with an F1 score of 0.8793 and an accuracy of 0.8662. The LSTM model, using *batch_size=16, dropout_rate=0.2, units=128*, had an F1 score of 0.8609 and an accuracy of 0.8442. The BiLSTM model, configured with *batch_size=32, dropout_rate=0.2, units=128*, achieved an F1 score of 0.8636 and an accuracy of 0.8492. The CNN model, with *pool_size=2, kernel_size=5, filters=64, dropout_rate=0.3, batch_size=32*, had an F1 score of 0.8691 and an accuracy of 0.8558. Using *batch_size=32, dropout_rate=0.2, units=128*, the GRU model achieved an F1 score of 0.8625 and an accuracy of 0.8467.

For the YNACC dataset: The LR model, with hyperparameters *C=1, max_iter=100, penalty='l2', solver='newton-cg'*, achieved an F1 score of 0.6074 and an accuracy of 0.5716. The RF model, with *bootstrap=False, max_depth=20, max_features='sqrt', min_samples_leaf=1, min_samples_split=2, n_estimators=200*, achieved the highest F1 score of 0.6453 and an accuracy of 0.6214. The SVM model, with hyperparameters *kernel='rbf', gamma='auto', degree=2, coef0=1, C=10*, had an F1 score of 0.6054 and an accuracy of 0.5707. The KNN model, with *algorithm='kd_tree', leaf_size=10, metric='manhattan', n_neighbors=3, p=1, weights='distance'*, achieved an F1 score of 0.5950 and an accuracy of 0.6067. The LSTM model, configured with hyperparameters *batch_size=64, dropout_rate=0.2, units=64*, had an F1 score of 0.6188 and an accuracy of 0.5733. The BiLSTM model, configured with *batch_size=64, dropout_rate=0.3, units=128*, had an F1 score of 0.6005 and an accuracy of 0.5557. The CNN model, using *pool_size=3, kernel_size=5, filters=32, dropout_rate=0.2, batch_size=32*, achieved an F1 score of 0.6243 and an accuracy of 0.5665. With *batch_size=64, dropout_rate=0.3, units=64*, the GRU model, had an F1 score of 0.5869 and an accuracy of 0.5656.

The RF model outperformed others on both datasets. In contrast, LR and GRU models had the lowest performances on these datasets. The performance of models varied significantly between the datasets, indicating the impact of dataset characteristics on model efficacy.

Dataset	Model	Hyperparameters	F1 score	Accuracy
C3	LR	C=0.1, max_iter=100, penalty='l2', solver='lbfgs'	0.8044	0.7875
	RF	bootstrap=True, max_depth=None, max_features='sqrt', min_samples_leaf=1, min_samples_split=5, n_estimators=300	0.8942	0.8833
	SVM	kernel='poly', gamma='scale', degree=3, coef0=0.5, C=10	0.8592	0.8433
	KNN	algorithm='auto', leaf_size=10, metric='manhattan', n_neighbors=11, p=1, weights='distance'	0.8793	0.8662
	LSTM	batch_size=16, dropout_rate=0.2, units=128	0.8609	0.8442
	BiLSTM	batch_size=32, dropout_rate=0.2, units=128	0.8636	0.8492
	CNN	pool_size=2, kernel_size=5, filters=64, dropout_rate=0.3, batch_size=32	0.8691	0.8558
	GRU	batch_size=32, dropout_rate=0.2, units=128	0.8625	0.8467
YNACC	LR	C=1, max_iter=100, penalty='l2', solver='newton-cg'	0.6074	0.5716
	RF	bootstrap=False, max_depth=20, max_features='sqrt', min_samples_leaf=1, min_samples_split=2, n_estimators=200	0.6453	0.6214
	SVM	kernel='rbf', gamma='auto', degree=2, coef0=1, C=10	0.6054	0.5707
	KNN	algorithm='kd_tree', leaf_size=10, metric='manhattan', n_neighbors=3, p=1, weights='distance'	0.5950	0.6067
	LSTM	batch_size=64, dropout_rate=0.2, units=64	0.6188	0.5733
	BiLSTM	batch_size=64, dropout_rate=0.3, units=128	0.6005	0.5557
	CNN	pool_size=3, kernel_size=5, filters=32, dropout_rate=0.2, batch_size=32	0.6243	0.5665
	GRU	batch_size=64, dropout_rate=0.3, units=64	0.5869	0.5656

Table 6.12: Comparison of model performance on stylistic features: Best F1 Score and Accuracy from Hyperparameter Optimization

6.4.2 Performance of Complexity Features

T 6.13 presents a comparative analysis of various models applied to complexity features extracted from two datasets, C3 and YNACC, using different hyperparameters. Each model's performance is evaluated based on the F1 score and accuracy metrics.

For the C3 dataset: The LR model achieved an F1 score of 0.9424 and an accuracy of 0.9375 with hyperparameters *C=0.1, max_iter=100, penalty='none', solver='newton-cg',*. The RF model, with hyperparameters *bootstrap=True, max_depth=10, max_features='log2', min_samples_leaf=1, min_samples_split=10, n_estimators=100*, demonstrated superior performance with an F1 score of 0.9487 and an accuracy of 0.9442, making it the best-performing model for this dataset. The SVM model, with hyperparameters *kernel='rbf', gamma='auto', degree=4, coef0=0.0, C=10*, achieved an F1 score of 0.9479 and an accuracy of 0.9433. The KNN model, with *algorithm='ball_tree', leaf_size=10, metric='euclidean', n_neighbors=11, p=1, weights='uniform'*, showed strong performance with an F1 score of 0.9408 and an accuracy of 0.9354. The LSTM model, using *batch_size=32, dropout_rate=0.3, units=32*, had an F1 score of 0.9468 and an accuracy of 0.9421. The BiLSTM model, configured with *batch_size=32, dropout_rate=0.2, units=64*, achieved an F1 score of 0.9472 and an accuracy of 0.9429. The CNN model, with *pool_size=2, kernel_size=3, filters=128, dropout_rate=0.3, batch_size=16*, had an F1 score of 0.9370 and an accuracy of 0.9313. Using *batch_size=16, dropout_rate=0.2, units=128*, the GRU model achieved an F1 score of 0.9474 and an accuracy of 0.9429.

For the YNACC dataset: The LR model, with hyperparameters *C=0.1, max_iter=100, penalty='l1', solver='saga'*, achieved an F1 score of 0.6555 and an accuracy of 0.5745. The RF model, with *bootstrap=True, max_depth=10, max_features='sqrt', min_samples_leaf=1, min_samples_split=2, n_estimators=300*, achieved an F1 score of 0.6105 and an accuracy of 0.5841. The SVM model, with hyperparameters *kernel='poly', gamma='scale', degree=3, coef0=0.0, C=10*, had an F1 score of 0.6565 and an accuracy of 0.5620. The KNN model, with hyperparameters *algorithm='ball_tree', leaf_size=20, metric='manhattan', n_neighbors=11, p=1, weights='distance'*, achieved an F1 score of 0.5638 and an accuracy of 0.5648. The LSTM model, configured with *batch_size=16, dropout_rate=0.2, units=32*, had an F1 score of 0.6340 and an accuracy of 0.5735. The BiLSTM model, configured with *batch_size=32, dropout_rate=0.2, units=64*, had an F1 score of 0.6345 and an accuracy of 0.5748. The CNN model, using *pool_size=2, kernel_size=3, filters=128, dropout_rate=0.3, batch_size=16*, achieved an F1 score of 0.5846 and an accuracy of 0.5798. With *batch_size=64, dropout_rate=0.2, units=64*, the GRU model had an F1 score of 0.6444 and an accuracy of 0.5546.

In summary, the RF model again outperformed others on the C3 dataset with an F1 score of 0.9487 and an accuracy of 0.9442. However, on the YNACC dataset, the performance of models was more varied, with the SVM model achieving the highest F1 score of 0.6565 but with an accuracy of 0.5620. The KNN model performed relatively poorly on the YNACC dataset, achieving an F1 score of 0.5638 and an accuracy of 0.5648. The performance of models varied significantly between the datasets, indicating the impact of dataset characteristics on model efficacy.

Dataset	Model	Hyperparameters	F1 score	Accuracy
C3	LR	C=0.1, max_iter=100, penalty='none', solver='newton-cg'	0.9424	0.9375
	RF	bootstrap=True, max_depth=10, max_features='log2', min_samples_leaf=1, min_samples_split=10, n_estimators=100	0.9487	0.9442
	SVM	kernel='rbf', gamma='auto', degree=4, coef0=0.0, C=10	0.9479	0.9433
	KNN	algorithm='ball_tree', leaf_size=10, metric='euclidean', n_neighbors=11, p=1, weights='uniform'	0.9408	0.9354
	LSTM	batch_size=32, dropout_rate=0.3, units=32	0.9468	0.9421
	BiLSTM	batch_size=32, dropout_rate=0.2, units=64	0.9472	0.9429
	CNN	pool_size=2, kernel_size=3, filters=128, dropout_rate=0.3, batch_size=16	0.9370	0.9313
	GRU	batch_size=16, dropout_rate=0.2, units=128	0.9474	0.9429
YNACC	LR	C=0.1, max_iter=100, penalty='l1', solver='saga'	0.6555	0.5745
	RF	bootstrap=True, max_depth=10, max_features='sqrt', min_samples_leaf=1, min_samples_split=2, n_estimators=300	0.6105	0.5841
	SVM	kernel='poly', gamma='scale', degree=3, coef0=0.0, C=10	0.6565	0.5620
	KNN	algorithm='ball_tree', leaf_size=20, metric='manhattan', n_neighbors=11, p=1, weights='distance'	0.5638	0.5648
	LSTM	batch_size=16, dropout_rate=0.2, units=32	0.6340	0.5735
	BiLSTM	batch_size=32, dropout_rate=0.2, units=64	0.6345	0.5748
	CNN	pool_size=2, kernel_size=3, filters=128, dropout_rate=0.3, batch_size=16	0.5846	0.5798
	GRU	batch_size=64, dropout_rate=0.2, units=64	0.6444	0.5546

Table 6.13: Comparison of model performance on complexity features: Best F1 Score and Accuracy from Hyperparameter Optimization

6.4.3 Performance of Psychological Features

Table 6.14 presents a comparative analysis of various models applied to psychological features extracted from two datasets, C3 and YNACC, using different hyperparameters. Each model's performance is evaluated based on the F1 score and accuracy metrics.

For the C3 dataset: The LR model, with hyperparameters *C=0.1, max_iter=100, penalty='l1', solver='saga'*, achieved an F1 score of 0.6707 and an accuracy of 0.5058. The RF model, with *bootstrap=True, max_depth=10, max_features='log2', min_samples_leaf=1, min_samples_split=5, n_estimators=200*, demonstrated an F1 score of 0.6688 and an accuracy of 0.6583. The SVM model, with hyperparameters *kernel='rbf', gamma='auto', degree=3, coef0=0.1, C=10*, achieved an F1 score of 0.6699 and an accuracy of 0.6600. The KNN model, with *algorithm='brute', leaf_size=10, metric='manhattan', n_neighbors=11, p=1, weights='distance'*, showed an F1 score of 0.6446 and an accuracy of 0.6158. The LSTM model, using *batch_size=32, dropout_rate=0.2, units=128*, had an F1 score of 0.6467 and an accuracy of 0.6562. The BiLSTM model, configured with *batch_size=64, dropout_rate=0.3, units=128*, achieved an F1 score of 0.6462 and an accuracy of 0.6546. The CNN model, with *pool_size=1, kernel_size=1, filters=64, dropout_rate=0.3, batch_size=16*, had an F1 score of 0.6462 and an accuracy of 0.6546. Using *batch_size=16, dropout_rate=0.2, units=64*, the GRU model achieved an F1 score of 0.6461 and an accuracy of 0.6558.

For the YNACC dataset: The LR model, with hyperparameters *C=0.1, class_weight=None, max_iter=100, penalty='l1', solver='liblinear'*, achieved an F1 score of 0.6929 and an accuracy of 0.5302. The RF model, configured with *bootstrap=True, max_depth=None, max_features='log2', min_samples_leaf=4, min_samples_split=5, n_estimators=300*, demonstrated an F1 score of 0.4210 and an accuracy of 0.5306. The SVM model, with hyperparameters *kernel='rbf', gamma='scale', degree=4, coef0=0.1, C=1*, had an F1 score of 0.4209 and an accuracy of 0.5304. The KNN model, with *algorithm='auto', leaf_size=10, metric='euclidean', n_neighbors=11, p=1, weights='distance'*, achieved an F1 score of 0.4185 and an accuracy of 0.5306. The LSTM model, configured with hyperparameters *batch_size=32, dropout_rate=0.2, units=32*, had an F1 score of 0.6753 and an accuracy of 0.5099. The BiLSTM model, configured with *batch_size=32, dropout_rate=0.2, units=128*, had an F1 score of 0.6753 and an accuracy of 0.5099. The CNN model, using *pool_size=1, kernel_size=1, filters=64, dropout_rate=0.3, batch_size=16*, achieved an F1 score of 0.4261 and an accuracy of 0.5393. The GRU model, with *batch_size=16, dropout_rate=0.3, units=128*, had an F1 score of 0.6753 and an accuracy of 0.5099.

In summary, for the C3 dataset, the LR model achieved the highest F1 score of 0.6707 with an accuracy of 0.5058, while the SVM model showed the highest accuracy of 0.6600. For the YNACC dataset, the LR model again showed the highest F1 score of 0.6929 with an accuracy of 0.5302, and the CNN model achieved the highest accuracy of 0.5393. The performance of models varied significantly between the datasets, indicating the impact of dataset characteristics on model efficacy. Overall, the LR model showed a consistently high F1 score, while the accuracy varied across different models and datasets.

Dataset	Model	Hyperparameters	F1 score	Accuracy
C3	LR	C=0.1, max_iter=100, penalty='l1', solver='saga'	0.6707	0.5058
	RF	bootstrap=True, max_depth=10, max_features='log2', min_samples_leaf=1, min_samples_split=5, n_estimators=200	0.6688	0.6583
	SVM	kernel='rbf', gamma='auto', degree=3, coef0=0.1, C=10	0.6699	0.6600
	KNN	algorithm='brute', leaf_size=10, metric='manhattan', n_neighbors=11, p=1, weights='distance'	0.6446	0.6158
	LSTM	batch_size=32, dropout_rate=0.2, units=128	0.6467	0.6562
	BiLSTM	batch_size=64, dropout_rate=0.3, units=128	0.6462	0.6546
	CNN	pool_size=1, kernel_size=1, filters=64, dropout_rate=0.3, batch_size=16	0.6462	0.6546
	GRU	batch_size=16, dropout_rate=0.2, units=64	0.6461	0.6558
YNACC	LR	C=0.1, class_weight=None, max_iter=100, penalty='l1', solver='liblinear'	0.6929	0.5302
	RF	bootstrap=True, max_depth=None, max_features='log2', min_samples_leaf=4, min_samples_split=5, n_estimators=300	0.4210	0.5306
	SVM	kernel='rbf', gamma='scale', degree=4, coef0=0.1, C=1	0.4209	0.5304
	KNN	algorithm='auto', leaf_size=10, metric='euclidean', n_neighbors=11, p=1, weights='distance'	0.4185	0.5306
	LSTM	batch_size=32, dropout_rate=0.2, units=32	0.6753	0.5099
	BiLSTM	batch_size=32, dropout_rate=0.2, units=128	0.6753	0.5099
	CNN	pool_size=1, kernel_size=1, filters=64, dropout_rate=0.3, batch_size=16	0.4261	0.5393
	GRU	batch_size=16, dropout_rate=0.3, units=128	0.6753	0.5099

Table 6.14: Comparison of model performance on psychological features: Best F1 Score and Accuracy from Hyperparameter Optimization

6.4.4 Performance of Emotion Features

Table 6.15 presents a comparative analysis of various models applied to emotion features extracted from two datasets, C3 and YNACC, using different hyperparameters. Each model's performance is evaluated based on the F1 score and accuracy metrics.

For the C3 dataset: The LR model achieved an F1 score of 0.7642 and an accuracy of 0.6775 with hyperparameters *C=0.1, max_iter=100, penalty='l1', solver='liblinear'.* The RF model, with *bootstrap=True, max_depth=30, max_features='log2', min_samples_leaf=4, min_samples_split=2, n_estimators=300*, obtained an F1 score of 0.8497 and an accuracy of 0.8350. The SVM model, with hyperparameters *kernel='rbf', gamma='auto', degree=3, coef0=0.5, C=10*, achieved an F1 score of 0.8244 and an accuracy of 0.8021. The KNN model, with *algorithm='ball_tree', leaf_size=20, metric='manhattan', n_neighbors=11, p=1, weights='uniform'*, showed performance with an F1 score of 0.8246 and an accuracy of 0.8096. The LSTM model, using *batch_size=16, dropout_rate=0.2, units=128*, had an F1 score of 0.8268 and an accuracy of 0.8117. The BiLSTM model, configured with *batch_size=32, dropout_rate=0.2, units=128*, achieved an F1 score of 0.8181 and an accuracy of 0.8037. The CNN model, with *pool_size=2, kernel_size=3, filters=64, dropout_rate=0.2, batch_size=16*, had an F1 score of 0.8203 and an accuracy of 0.8079. Using *batch_size=64, dropout_rate=0.2, units=128*, the GRU model achieved the highest performance with an F1 score of 0.8616 and an accuracy of 0.8496.

For the YNACC dataset: The LR model achieved an F1 score of 0.6236 and an accuracy of 0.5653 with hyperparameters *C=0.1, class_weight='balanced', max_iter=200, penalty='l1', solver='liblinear',* . The RF model, with *bootstrap=True, max_depth=20, max_features='sqrt', min_samples_leaf=2, min_samples_split=5, n_estimators=300*, achieved an F1 score of 0.5714 and an accuracy of 0.5900. The SVM model, with hyperparameters *kernel='rbf', gamma='scale', degree=3, coef0=0.1, C=1*, had an F1 score of 0.6063 and an accuracy of 0.5587. The KNN model, with *algorithm='auto', leaf_size=20, metric='manhattan', n_neighbors=7, p=1, weights='distance'*, achieved an F1 score of 0.5913 and an accuracy of 0.5519. The LSTM model, configured with *batch_size=64, dropout_rate=0.2, units=128*, had an F1 score of 0.6080 and an accuracy of 0.5636. The BiLSTM model, with *batch_size=16, dropout_rate=0.3, units=32*, had an F1 score of 0.6153 and an accuracy of 0.5627. The CNN model, using hyperparameters *pool_size=3, kernel_size=5, filters=128, dropout_rate=0.3, batch_size=32*, achieved an F1 score of 0.6018 and an accuracy of 0.5593. Using *batch_size=64, dropout_rate=0.2, units=64*, the GRU model had an F1 score of 0.6344 and an accuracy of 0.5564.

In summary, the GRU model outperformed both datasets C3 and YNACC with an F1 score of 0.8616 and 0.6344 and an accuracy of 0.8496 and 0.5564, respectively. In contrast, LR and RF models have the lowest performance on these datasets. The performance of the models varied significantly between the datasets, with some models performing better on one dataset than the other.

Dataset	**Model**	**Hyperparameters**	**F1 score**	**Accuracy**
C3	LR	C=0.1, max_iter=100, penalty='l1', solver='liblinear'	0.7642	0.6775
	RF	bootstrap=True, max_depth=30, max_features='log2', min_samples_leaf=4, min_samples_split=2, n_estimators=300	0.8497	0.8350
	SVM	kernel='rbf', gamma='auto', degree=3, coef0=0.5, C=10	0.8244	0.8021
	KNN	algorithm='ball_tree', leaf_size=20, metric='manhattan', n_neighbors=11, p=1, weights='uniform'	0.8246	0.8096
	LSTM	batch_size=16, dropout_rate=0.2, units=128	0.8268	0.8117
	BiLSTM	batch_size=32, dropout_rate=0.2, units=128	0.8181	0.8037
	CNN	pool_size=2, kernel_size=3, filters=64, dropout_rate=0.2, batch_size=16	0.8203	0.8079
	GRU	batch_size=64, dropout_rate=0.2, units=128	0.8616	0.8496
YNACC	LR	C=0.1, class_weight='balanced', max_iter=200, penalty='l1', solver='liblinear'	0.6236	0.5653
	RF	bootstrap=True, max_depth=20, max_features='sqrt', min_samples_leaf=2, min_samples_split=5, n_estimators=300	0.5714	0.5900
	SVM	kernel='rbf', gamma='scale', degree=3, coef0=0.1, C=1	0.6063	0.5587
	KNN	algorithm='auto', leaf_size=20, metric='manhattan', n_neighbors=7, p=1, weights='distance'	0.5913	0.5519
	LSTM	batch_size=64, dropout_rate=0.2, units=128	0.6080	0.5636
	BiLSTM	batch_size=16, dropout_rate=0.3, units=32	0.6153	0.5627
	CNN	pool_size=3, kernel_size=5, filters=128, dropout_rate=0.3, batch_size=32	0.6018	0.5593
	GRU	batch_size=64, dropout_rate=0.2, units=64	0.6344	0.5564

Table 6.15: Comparison of model performance on emotional features: Best F1 Score and Accuracy from Hyperparameter Optimization

6.4.5 Performance of POS Tag Features

Table 6.16 provides a comparative analysis of various models applied to POS tag features from two datasets, C3 and YNACC, with different hyperparameter settings. Each model's effectiveness is evaluated based on the F1 score and accuracy metrics.

For the C3 dataset: The LR model, with hyperparameters *C=0.1, max_iter=300, penalty='l1', solver='liblinear'*, achieved an F1 score of 0.7430 and an accuracy of 0.7250. The RF model, configured with *bootstrap=False, max_depth=None, max_features='log2', min_samples_leaf=1, min_samples_split=2, n_estimators=200*, demonstrated superior performance with an F1 score of 0.9029 and an accuracy of 0.8950, making it the highest-performing model for this dataset. The SVM model, with hyperparameters *kernel='rbf', gamma=scale, degree=2, coef0=0.0, C=10*, achieved an F1 score of 0.8663 and an accuracy of 0.8562. The KNN model, with *algorithm='auto', leaf_size=10, metric='manhattan', n_neighbors=11, p=1, weights='distance'*, performed well with an F1 score of 0.8907 and an accuracy of 0.8850. The LSTM model, using *batch_size=16, dropout_rate=0.3, units=128*, had an F1 score of 0.8794 and an accuracy of 0.8667. The BiLSTM model, configured with *batch_size=16, dropout_rate=0.3, units=128*, achieved an F1 score of 0.8923 and an accuracy of 0.8833. The CNN model, with *pool_size=2, kernel_size=3, filters=64, dropout_rate=0.2, batch_size=32*, had an F1 score of 0.8901 and an accuracy of 0.8771. Using *batch_size=32, dropout_rate=0.2, units=64*, the GRU model achieved an F1 score of 0.8907 and an accuracy of 0.8800.

For the YNACC dataset: The LR model, with hyperparameters *C=0.1, class_weight=None, max_iter=100, penalty='l1', solver='liblinear'*, achieved an F1 score of 0.6521 and an accuracy of 0.5106. The RF model, with *bootstrap=True, max_depth=None, max_features='log2', min_samples_leaf=2, min_samples_split=5, n_estimators=200*, had an F1 score of 0.5852 and an accuracy of 0.5643. The SVM model, with hyperparameters *kernel='poly', gamma='auto', degree=4, coef0=0.5, C=10*, achieved an F1 score of 0.5833 and an accuracy of 0.5351. The KNN model, with *algorithm='auto', leaf_size=10, metric='euclidean', n_neighbors=3, p=1, weights='distance'*, had an F1 score of 0.5413 and an accuracy of 0.5609. The LSTM model, using *batch_size=64, dropout_rate=0.3, units=128*, achieved an F1 score of 0.6726 and an accuracy of 0.5083. The BiLSTM model, configured with *batch_size=64, dropout_rate=0.2, units=64*, had an F1 score of 0.6742 and an accuracy of 0.5106. The CNN model, with *pool_size=2, kernel_size=3, filters=128, dropout_rate=0.2, batch_size=16*, achieved an F1 score of 0.6723 and an accuracy of 0.5106. The GRU model achieved the highest performance of an F1 score of 0.6749 and an accuracy of 0.5099 using *batch_size=16, dropout_rate=0.3, units=32.*

In summary, the RF model excelled on the C3 dataset, with an F1 score of 0.9029 and an accuracy of 0.8950. On the YNACC dataset, the BiLSTM and GRU models performed better with F1 scores of 0.6742 and 0.6749, respectively. On the other hand, the LR model had the lowest performance on the C3 dataset, with an F1 score of 0.7430 and an accuracy of 0.7250 while for the YNACC dataset, SVM showed the lowest performance with an F1 score of 0.5833 and an accuracy of 0.5351. Overall, model performance varied significantly between datasets, highlighting the impact of dataset characteristics on the effectiveness of different models.

Dataset	**Model**	**Hyperparameters**	**F1 score**	**Accuracy**
C3	LR	C=0.1, max_iter=300, penalty='l1', solver='liblinear'	0.7430	0.7250
	RF	bootstrap=False, max_depth=None, max_features='log2', min_samples_leaf=1, min_samples_split=2, n_estimators=200	0.9029	0.8950
	SVM	kernel='rbf', gamma=scale, degree=2, coef0=0.0, C=10	0.8663	0.8562
	KNN	algorithm='auto', leaf_size=10, metric='manhattan', n_neighbors=11, p=1, weights='distance'	0.8907	0.8850
	LSTM	batch_size=16, dropout_rate=0.3, units=128	0.8794	0.8667
	BiLSTM	batch_size=16, dropout_rate=0.3, units=128	0.8923	0.8833
	CNN	pool_size=2, kernel_size=3, filters=64, dropout_rate=0.2, batch_size=32	0.8901	0.8771
	GRU	batch_size=32, dropout_rate=0.2, units=64	0.8907	0.8800
YNACC	LR	C=0.1, class_weight=None, max_iter=100, penalty='l1', solver='liblinear'	0.6521	0.5106
	RF	bootstrap=True, max_depth=None, max_features='log2', min_samples_leaf=2, min_samples_split=5, n_estimators=200	0.5852	0.5643
	SVM	kernel='poly', gamma='auto', degree=4, coef0=0.5, C=10	0.5833	0.5351
	KNN	algorithm='auto', leaf_size=10, metric='euclidean', n_neighbors=3, p=1, weights='distance'	0.5413	0.5609
	LSTM	batch_size=64, dropout_rate=0.3, units=128	0.6726	0.5083
	BiLSTM	batch_size=64, dropout_rate=0.2, units=64	0.6742	0.5106
	CNN	pool_size=2, kernel_size=3, filters=128, dropout_rate=0.2, batch_size=16	0.6723	0.5106
	GRU	batch_size=16, dropout_rate=0.3, units=32	0.6749	0.5099

Table 6.16: Comparison of model performance on POS Tag features: Best F1 Score and Accuracy from Hyperparameter Optimization

6.4.6 Performance of All Features

Table 6.17 presents the performance of various ML and DL models when applied to a comprehensive set of features, including stylistic, complexity, psychological, emotion, and POS tag features. The models were evaluated on two datasets, C3 and YNACC, utilizing different hyperparameter settings. The performance metrics include F1 score and accuracy.

For the C3 dataset: The LR model achieved an F1 score of 0.9443 and an accuracy of 0.9396 with hyperparameters *C=0.1, max_iter=200, penalty='none', solver='lbfgs'*. The RF model, configured with *bootstrap=False, max_depth=30, max_features='sqrt', min_samples_leaf=1, min_samples_split=5, n_estimators=100*, showed a slightly higher F1 score of 0.9459 and an accuracy of 0.9413, making it the top performer for this dataset. The SVM model, using hyperparameters *kernel='linear', gamma=scale, degree=2, coef0=0.5, C=1*, achieved an F1 score of 0.9444 and an accuracy of 0.9396. The KNN model, with *algorithm='auto', leaf_size=10, metric='manhattan', n_neighbors=11, p=1, weights='uniform'*, obtained an F1 score of 0.9159 and an accuracy of 0.9087. The LSTM model, configured with *batch_size=64, dropout_rate=0.2, units=128*, had an F1 score of 0.8888 and an accuracy of 0.8779. The BiLSTM model, with *batch_size=64, dropout_rate=0.2, units=128*, achieved an F1 score of 0.8942 and an accuracy of 0.8812. The CNN model, using *pool_size=2, kernel_size=3, filters=64, dropout_rate=0.3, batch_size=64*, performed well with an F1 score of 0.9353 and an accuracy of 0.9304. With *batch_size=32, dropout_rate=0.2, units=64*, the GRU model achieved an F1 score of 0.8981 and an accuracy of 0.8892.

For the YNACC dataset: The LR model achieved an F1 score of 0.5902 and an accuracy of 0.5735 using hyperparameters *C=0.1, class_weight='balanced', max_iter=200, penalty='l1', solver='saga'*. The RF model, with *bootstrap=False, max_depth=30, max_features='log2', min_samples_leaf=4, min_samples_split=5, n_estimators=200*, showed an F1 score of 0.6291 and an accuracy of 0.6052. The SVM model, with *kernel='rbf', gamma='scale', degree=3, coef0=0.1, C=10*, achieved an F1 score of 0.6448 and an accuracy of 0.5640. The KNN model obtained an F1 score of 0.5873 and an accuracy of 0.5638 using *algorithm='auto', leaf_size=10, metric='manhattan', n_neighbors=9, p=1, weights='distance'*, which is lowest among all the models. The LSTM model, with *batch_size=64, dropout_rate=0.2, units=32*, achieved the highest performance of an F1 score of 0.6754 and an accuracy of 0.5103. The BiLSTM model, using *batch_size=32, dropout_rate=0.2, units=128*, obtained an F1 score of 0.6008 and an accuracy of 0.5712. The CNN model, with *pool_size=2, kernel_size=5, filters=128, dropout_rate=0.3, batch_size=32*, had an F1 score of 0.6155 and an accuracy of 0.5811. Using *batch_size=16, dropout_rate=0.2, units=64*, the GRU model achieved an F1 score of 0.6750 and an accuracy of 0.5103.

In summary, the RF model shows the highest performance on C3 datasets, with F1 scores of 0.9459 and an accuracy of 0.9413 while, the LSTM showed lower performance with an F1 score of 0.8888 and an accuracy of 0.8779. In the YNACC dataset, LSTM achieved the highest F1 score of 0.6754 and an accuracy of 0.5103 similar to GRU model results, while, the KNN model showed the lowest F1 score of 0.5873 and an accuracy of 0.5638. The model performance depends significantly on the datasets.

Dataset	Model	Hyperparameters	F1 score	Accuracy
C3	LR	C=0.1, max_iter=200, penalty='none', solver='lbfgs'	0.9443	0.9396
	RF	bootstrap=False, max_depth=30, max_features='sqrt', min_samples_leaf=1, min_samples_split=5, n_estimators=100	0.9459	0.9413
	SVM	kernel='linear', gamma=scale, degree=2, coef0=0.5, C=1	0.9444	0.9396
	KNN	algorithm='auto', leaf_size=10, metric='manhattan', n_neighbors=11, p=1, weights='uniform'	0.9159	0.9087
	LSTM	batch_size=64, dropout_rate=0.2, units=128	0.8888	0.8779
	BiLSTM	batch_size=64, dropout_rate=0.2, units=128	0.8942	0.8812
	CNN	pool_size=2, kernel_size=3, filters=64, dropout_rate=0.3, batch_size=64	0.9353	0.9304
	GRU	batch_size=32, dropout_rate=0.2, units=64	0.8981	0.8892
YNACC	LR	C=0.1, class_weight='balanced', max_iter=200, penalty='l1', solver='saga'	0.5902	0.5735
	RF	bootstrap=False, max_depth=30, max_features='log2', min_samples_leaf=4, min_samples_split=5, n_estimators=200	0.6291	0.6052
	SVM	kernel='rbf', gamma='scale', degree=3, coef0=0.1, C=10	0.6448	0.5640
	KNN	algorithm='auto', leaf_size=10, metric='manhattan', n_neighbors=9, p=1, weights='distance'	0.5873	0.5638
	LSTM	batch_size=64, dropout_rate=0.2, units=32	0.6754	0.5103
	BiLSTM	batch_size=32, dropout_rate=0.2, units=128	0.6008	0.5712
	CNN	pool_size=2, kernel_size=5, filters=128, dropout_rate=0.3, batch_size=32	0.6155	0.5811
	GRU	batch_size=16, dropout_rate=0.2, units=64	0.6750	0.5103

Table 6.17: Comparison of model performance on all features: Best F1 Score and Accuracy from Hyperparameter Optimization

6.4.7 Error Analysis

The error analysis in Figures 6.12 and 6.13 provided an in-depth comparison of various ML and DL models, across six feature sets: Stylistic, Complexity, Psychological, Emotional, POS Tag, and All Features for C3 and YNACC datasets. The bar plots illustrate False Positives (FP) and False Negatives (FN), offering insight into how each model handles different feature types. The C3 dataset consists of 12,000 comments, with 20% (2,400 comments) used for testing. Similarly, for the YNACC dataset, which contains 22,795 comments, 20% (4,559 comments) were used for testing.

For the C3 dataset, the psychological feature consistently generates the highest error rates across all models, especially in FP and FN counts. In the LR model, the psychological and emotional feature sets show the highest error rates for both FP and FN, particularly for FP. However, LR performs significantly better with complexity and all features, where error rates are much lower. The stylistic and POS Tag feature sets also exhibit high errors, but the complexity and all features have significantly lower error counts, indicating better performance with these features.

RF model demonstrates the highest FP count for the psychological feature set, while its FN count is moderate. The errors in the other feature sets, including the complexity and all features, are relatively low, showing that RF performs better on a broader range of features than other models.

The KNN model struggles the most with the psychological feature set, with a sharp rise in both FP and FN counts, particularly FNs. It performs comparatively better with complexity and all feature sets, where the error rates are more balanced and lower.

Similarly, the SVM shows a marked increase in FP and FN errors for the psychological feature set with FN counts slightly higher. SVM performs much better for the complexity and all feature sets, with lower error rates across both FP and FN counts.

Deep learning models, including LSTM, BiLSTM, and GRU, also encounter challenges with psychological features, particularly in generating FNs. However, they exhibit improved accuracy when working complexity and all features, where error rates drop significantly. CNN maintains relatively consistent performance across most feature sets but struggles with psychological features.

Overall, the error analysis figures for the C3 dataset indicate that each model has distinct strengths and weaknesses across the various feature sets. Abstract feature sets like psychological and emotional lead to higher error rates, while complexity and all features yield more accurate predictions with fewer FPs and FNs. This suggests that structured and comprehensive features improve classification performance, while models find it harder to generalize over abstract or sentiment-driven features.

For the YNACC dataset, the psychological feature consistently produces the highest error rates across all models, especially in FPs. Most models struggle significantly with these features. However, they show improvement when applied to more structured feature sets, such as the stylistic and complexity features, where the error rates between FPs and FNs are more balanced.

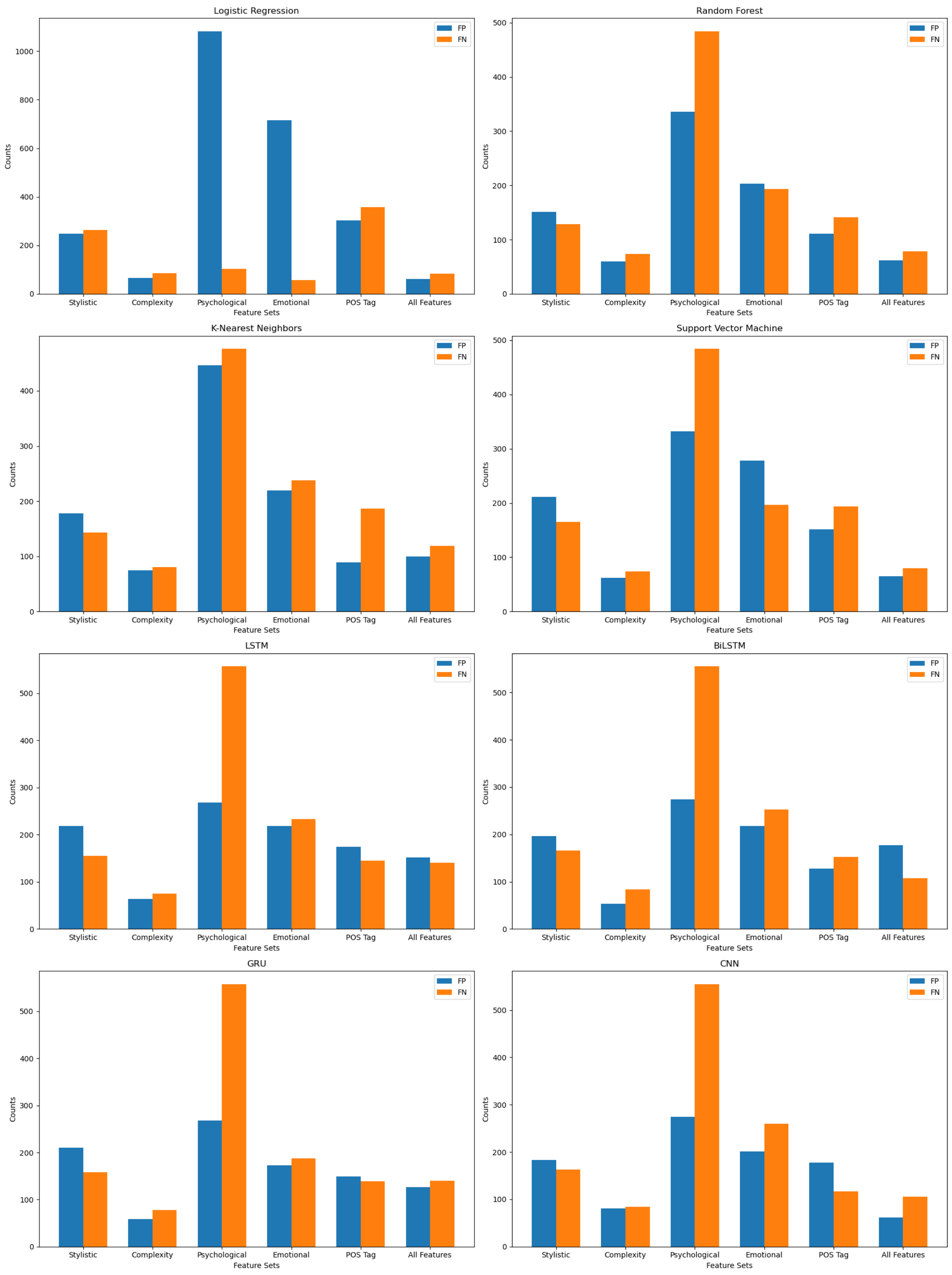


Figure 6.12: Error Analysis of C3 dataset with various models

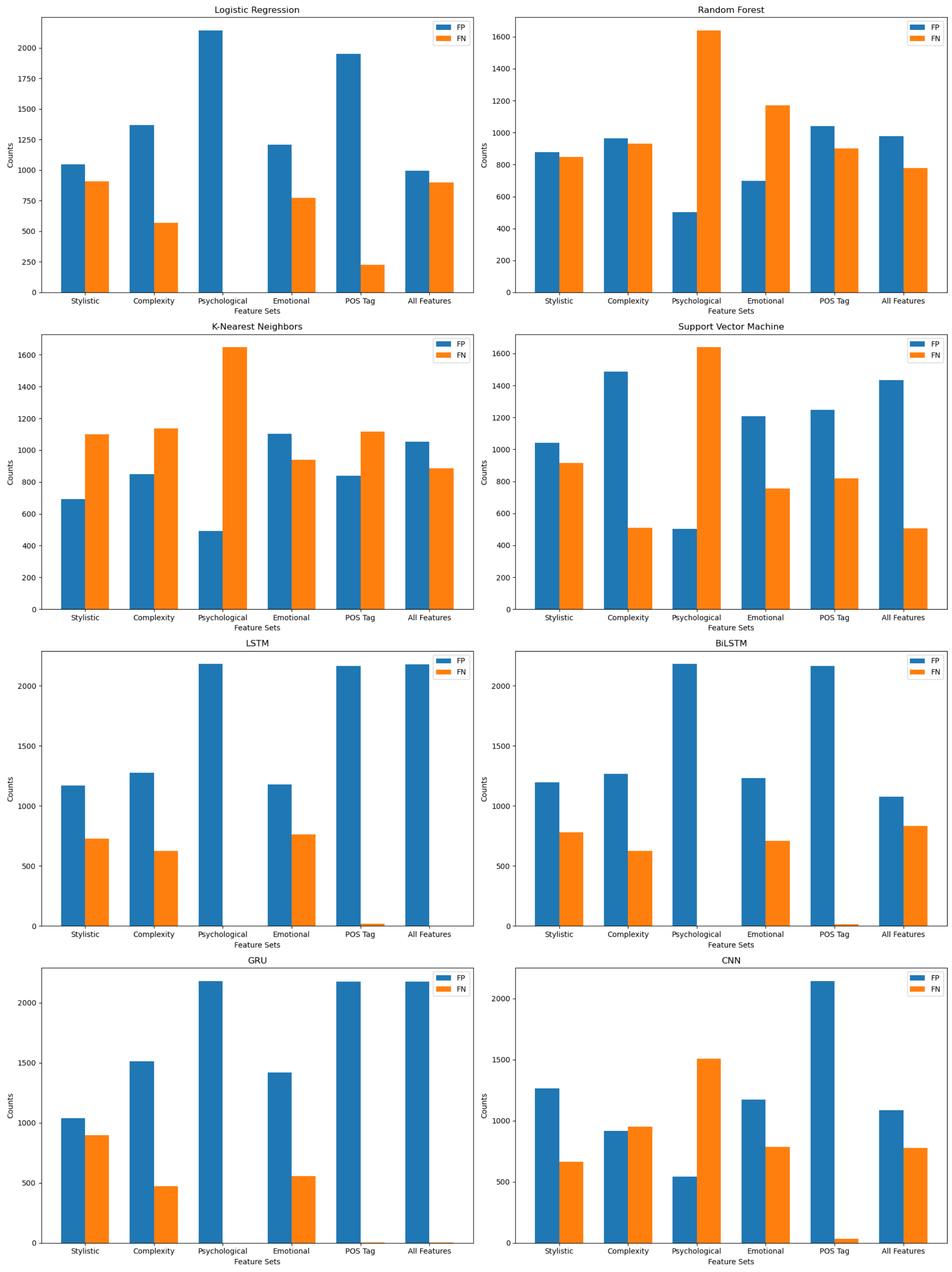


Figure 6.13: Error Analysis of YNACC dataset with various models

The LR model demonstrates a particularly high FP count for the psychological feature, exceeding 2,000, although its FN count is relatively low (below 50). The POS Tag features generate a high FP count, but its FN count remains below 250. LR shows a more balanced performance for the stylistic, complexity, emotional and all feature sets, with both FP and FN counts around 1,100, performing better than in the psychological features.

RF model shows higher FNs than FPs for psychological features, followed by emotional features. For other feature sets, FPs generally exceed FNs, though both remain below 1,000.

KNN exhibits high FNs for the psychological feature while showing more moderate FP and FN values across other feature sets.

The SVM model follows a similar error pattern, with high FNs for the psychological feature. Complexity and all features yield the lowest FN counts, but FP counts are higher across all other feature sets.

LSTM and GRU models consistently show higher FPs and lower FNs across all feature sets. For the psychological, POS Tag, and all feature sets, FP counts exceed 2,000, while FNs remain negligible. In feature sets such as stylistic, complexity, and emotional, FPs remain higher than FNs, but all are below 1,200.

The BiLSTM model follows the same pattern as LSTM and GRU, except for all feature sets, where both FP and FN counts are moderate and similar to the stylistic, complexity, and emotional feature sets.

CNN stands out for having the highest FN count for the psychological feature set, indicating difficulty in capturing the nuances of these features. However, it maintains moderate FP counts across other feature sets, except for the POS Tag feature, where FPs exceed 2,000 and FNs are negligible.

Overall, the psychological feature presents the greatest challenge across all models, particularly in FP counts for models like LR, LSTM, BiLSTM, and GRU. On the other hand, models like RF, KNN, SVM, and CNN tend to perform better at minimizing FNs, especially when using structured feature sets. The YNACC dataset consistently shows higher FP and FN counts than the C3 dataset across all models and feature sets, suggesting that error rates are influenced by dataset quality and their relationship to comment constructiveness.

In summary, similar to the C3 dataset, models in the YNACC dataset struggle the most with the psychological feature, which generates the highest error rates, followed by the POS Tag and emotional feature. However, models demonstrate improved accuracy when utilizing more structured feature sets, such as stylistic and complexity, which lead to reduced FPs and FNs. Generally, sentiment-driven features (e.g., Psychological and Emotional) are more prone to errors, while structured features contribute to better classification outcomes across datasets.

7 Discussion

In this section, we have presented our findings, highlighting the limitations of our experiment, and suggested potential areas for future enhancements.

7.1 Exchange of Views

This study explored various linguistic, emotional, and psychological features to understand their contributions to the constructiveness of online comments. By extracting features from the C3 and YNACC datasets, we gained insights into how these features correlate with and differentiate between constructive and non-constructive comments. Additionally, we evaluated the performance of various ML and DL models across different feature types and datasets, revealing notable trends and variances in model effectiveness. The error analysis further provided a deeper understanding of model weaknesses, particularly with more abstract feature sets like psychological and emotional features.

Stylistic features

Our analysis revealed significant differences in how stylistic features correlate with constructiveness across the C3 and YNACC datasets. For instance, the strong positive correlation of punctuation types with constructiveness in the C3 dataset suggests that comments with a range of punctuation marks are often perceived as more constructive, indicative of well-structured and coherent expression of ideas. In contrast, the weaker correlation in the YNACC dataset may reflect differences in discourse patterns or dataset characteristics. Features like Stopwords also showed positive correlations, supporting the assumption that grammatical flow and clarity contribute to perceived constructiveness. Conversely, Exclamation Marks and Hashtags had negative correlations, particularly in the C3 dataset, suggesting that these elements, are typically associated with informal discourse, and are more common in non-constructive comments.

The RF model consistently outperformed other models in handling stylistic features, achieving the highest F1 score and accuracy across both datasets. For the C3 dataset, the RF model led with an F1 score of 0.8942 and an accuracy of 0.8833, while for the YNACC dataset, it achieved an F1 score of 0.6453 and an accuracy of 0.6214. This reflects RF's robustness and adaptability in handling structured stylistic features, as shown in the error analysis, where RF maintained balanced FPs and FNs for these features across both datasets.

Complexity features

Features such as word count and MTLD showed positive correlations with constructiveness, especially in the C3 dataset, where longer and more lexically diverse comments were more likely to be constructive, reflecting deeper engagement and thoughtful contributions. However, TTR, which measures lexical diversity, showed an inverse correlation with constructiveness, suggesting that excessive lexical diversity might complicate comprehension or detract from constructive arguments.

The RF model again emerged as the top performer for the C3 dataset with an F1 score of 0.9487 and an accuracy of 0.9442. However, for the YNACC dataset, the SVM model achieved the highest F1 score of 0.6565, but with a lower accuracy of 0.5620. This indicates that while RF excels with the complexity features, the SVM model might be more sensitive to the specifics of the dataset. The error analysis shows that complexity features lead to relatively lower FPs and FNs, making them easier for most models to handle.

Psychological features

The role of Sentiment Score, in determining constructiveness was less effective than stylistic and complexity features. Positive but modest correlations in both datasets indicate that comments with more positive sentiment are slightly more likely to be constructive, showing the significance of a respectful and supportive attitude. However, sentiment alone is not a reliable predictor of constructiveness.

The LR model showed superior performance across both datasets, with an F1 score of 0.6707 on C3 and 0.6929 on YNACC. This indicates that LR is good at capturing temporal dependencies and nuances in psychological features, particularly in the YNACC dataset. The error analysis revealed that psychological features yielded the highest error rates across models, indicating that psychological features introduce more complexity and abstract patterns that are challenging for models to capture.

Emotional features

Positive emotions like joy and trust are associated with constructiveness in the C3 dataset, suggesting that these emotions contribute to constructive discourse. In contrast, negative emotions like anger and disgust were associated with non-constructive comments, especially in the YNACC dataset.

The GRU model achieved the highest F1 scores and accuracies for both datasets (0.8616 and 0.8496 for C3, and 0.6344 and 0.5564 for YNACC). This consistently high performance across both datasets suggests that GRU is well-suited to handle emotional characteristics. The error analysis indicated that emotional features produce high error rates, suggesting that these models struggle to capture emotional nuances accurately.

POS Tag features

Our analysis revealed a strong positive correlation between prepositions and subordinating conjunctions with constructiveness in the C3 dataset and a moderate correlation in the YNACC dataset.

The RF model was the top performer on the C3 dataset, with an impressive F1 score of 0.9029 and an accuracy of 0.8950. Conversely, on the YNACC dataset, GRU models performed better than others, highlighting their effectiveness in capturing sequential dependencies in POS tag features. The error analysis showed that these features resulted in lower error rates than psychological and emotional features, making them easier for most models to process effectively, however, struggled most with YNACC datasets

7.2 Limitation

Despite the comprehensive approach and significant findings, this study has some limitations. Recognizing these constraints is crucial for appropriately interpreting the results and guiding future research directions.

Firstly, our study relies on the C3 and YNACC datasets, which may not fully capture the diversity of online news comments across different platforms. These datasets may not represent all online news environments because they are derived from specific sources. This limitation impacts the ability to generalize the findings, highlighting the need for additional research that includes a wider range of datasets to validate and improve these results.

Secondly, the features analyzed are stylistic, complexity, psychological, and emotional which are comprehensive but do not cover every possible aspect of constructiveness. Other potential features, such as user metadata (e.g., user reputation, commenting history), temporal patterns (e.g., time of posting), and contextual information (e.g., surrounding comment thread), were excluded from this study. These additional dimensions could provide further insights into the dynamics of constructive comments and should be considered in future investigations.

The machine learning and deep learning models evaluated in this study, while effective, do have some limitations. Various factors can influence the model's performance, such as the quality and size of the training data, the selection of hyperparameters, and the specific architecture of the models. Additionally, models such as RF and GRU showed variability in performance depending on the dataset, indicating that no single model is universally superior. This highlights the importance of customizing and fine-tuning models to suit specific datasets. The error analysis also highlights that abstract features like psychological and emotional create more challenges for model accuracy, suggesting that future work should explore better feature engineering or alternative modeling approaches to handle these aspects effectively.

The evaluation metrics utilized in our study are F1 score and accuracy which provide a quantitative measure of model performance but might not capture all nuances of comment constructiveness. Metrics such as precision, recall and other qualitative assessments could provide additional insights.

Finally, this study did not thoroughly explore the ethical implications of automated content moderation. It is essential to consider the potential for bias and the impact on free speech. Developing transparent, fair, and accountable moderation systems is critical to balancing these concerns.

In conclusion, while this study provides valuable insights into the constructiveness of online news comments, it is essential to acknowledge these limitations. Addressing these issues in future research will deepen our understanding and aid in creating more effective and ethical approaches to promote positive online discussions.

8 Conclusion and Future Work

8.1 Conclusion

This research investigates the factors that influence the constructiveness of online comments by analyzing various linguistic, emotional, and psychological features across the C3 and YNACC datasets. The main objective is to explore several important research questions related to the features that contribute to or detract from the constructiveness of comments, the impact of these features as measured by correlation and mean percentage scores, the effectiveness of various ML and DL models in identifying constructive comments, and the influence of different datasets on model performance.

Our findings indicate that stylistic and complexity features play a significant role in the constructiveness of comments. For instance, punctuation types and word count were positively correlated with constructiveness, suggesting that well-structured and coherent comments are perceived as more constructive. Similarly, MTLD were associated with higher constructiveness, reflecting deeper engagement and more thoughtful contributions. Positive emotional expressions, including joy and trust, were also associated with constructive comments, while negative emotions like anger and disgust were observed in non-constructive comments.

When examining the impact of these features, it was evident that aspects such as stopwords and punctuation types positively influenced constructiveness, highlighting the significance of grammatical flow and structural coherence. In contrast, features like exclamation marks and hashtags, which typically signal emotional intensity and informal communication, detracted from constructiveness.

Regarding model performance, the RF model consistently outperformed others across various feature types and datasets, highlighting its robustness and adaptability in identifying constructive comments. The GRU model excelled at capturing temporal dependencies and nuances in psychological and emotional features.

There were significant variations in model performance when comparing the C3 and YNACC datasets. While the RF model showed consistent performance across both datasets, models like SVM and GRU exhibited more variability. This highlights the necessity of customizing model selection to address specific features and challenges of datasets.

The error analysis highlights that across both the C3 and YNACC datasets, models struggled the most with psychological and emotional features, showing the highest error rates. In contrast, structured features like stylistics and complexity resulted in better performance with fewer errors. While both traditional and DL models faced challenges with abstract features, they performed more accurately with structured ones. The YNACC dataset showed higher error rates than the C3 dataset, likely due to its complexity and size, indicating that dataset quality significantly impacts model performance. Overall, improving the representation of sentiment-driven features like psychological and emotional is essential for enhancing model accuracy in future research.

In conclusion, the research underscored the significant role of stylistic and complexity features in comment constructiveness, highlighting the challenges models face with abstract features like psychological and emotional aspects. The consistent performance of the RF model across datasets, combined with the insights from error analysis, suggests that more structured and comprehensive features lead to more accurate classifications. This points to the need for customizing model selection and feature engineering to address the specific challenges of different datasets and feature types.

8.2 Future Work

Future research can enhance our understanding of online comment constructiveness by exploring several key areas.

First, using a diverse range of datasets from various platforms and languages will help determine if our findings are universally applicable. This will reveal if different online communities have unique ways of fostering constructive comments.

Next, examining how different features interact, such as punctuation and emotional tone, can provide deeper insights into their combined effects on constructiveness.

Developing and testing new hybrid models that combine various machine learning techniques or transformer-based models like BERT, could improve accuracy and reduce error in identifying constructive comments.

Including additional data types, such as user history and context, will offer a more comprehensive understanding of comment constructiveness.

Future research should focus on addressing the challenges highlighted by the error analysis, particularly the difficulties models face with abstract features like psychological and emotional aspects.

Finally, ethical considerations are crucial. Future work should ensure that automated moderation tools are fair and transparent, balancing the promotion of constructive comments with the protection of free speech.

By focusing on these areas, future research can further deepen our understanding of online discourse and contribute to more effective methods for fostering constructive communication in digital environments. Ultimately, this can result in healthier, more vibrant, and productive online communities.

Bibliography

[ABP22] Ankit Aich, Souvik Bhattacharya, and Natalie Parde. “Demystifying Neural Fake News via Linguistic Feature-Based Interpretation”. In: *Proceedings of the 29th International Conference on Computational Linguistics*. Ed. by Nicoletta Calzolari et al. Gyeongju, Republic of Korea: International Committee on Computational Linguistics, Oct. 2022, pp. 6586–6599. URL: https://aclanthology.org/2022.coling-1.573.

[BES10a] Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. “SentiWordNet 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining”. In: *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*. Ed. by Nicoletta Calzolari et al. Valletta, Malta: European Language Resources Association (ELRA), May 2010. URL: http://www.lrec-conf.org/proceedings/lrec2010/pdf/769_Paper.pdf.

[BES10b] Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. “SentiWordNet 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining”. In: *LREC*. European Language Resources Association. 2010.

[Bis06] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. New York: Springer, 2006.

[Bre01] Leo Breiman. “Random Forests”. In: *Machine Learning* 45.1 (2001), pp. 5–32. DOI: 10.1023/A:1010933404324.

[Cho+14] Kyunghyun Cho et al. “Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling”. In: *Advances in Neural Information Processing Systems*. 2014. URL: https://arxiv.org/abs/1406.1078.

[CV95] Corinna Cortes and Vladimir Vapnik. “Support-Vector Networks”. In: *Machine Learning* 20.3 (1995), pp. 273–297. DOI: 10.1007/BF00994018.

[CXW11] Jianqing Chen, Hong Xu, and Andrew Whinston. “Moderated Online Communities and Quality of User-Generated Content”. In: *J. of Management Information Systems* 28 (Oct. 2011), pp. 237–268. DOI: 10.2139/ssrn.1481772.

[Eti17] B. Etim. “The Times sharply increases articles open for comments, using Google's technology”. In: *New York Times* (June 2017).

[HMM14] Ahmed Hassan Yousef, Walaa Medhat, and Hoda Mohamed. “Sentiment Analysis Algorithms and Applications: A Survey”. In: *Ain Shams Engineering Journal* 5 (May 2014). DOI: 10.1016/j.asej.2014.04.011.

[HS97] Sepp Hochreiter and Jürgen Schmidhuber. “Long short-term memory”. In: *Neural Computation* 9.8 (1997), pp. 1735–1780. DOI: 10.1162/neco.1997.9.8.1735.

[HTF09] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* New York: Springer, 2009.

[HXY15] Zhiheng Huang, Wei Xu, and Kai Yu. "Bidirectional LSTM-CRF Models for Sequence Tagging". In: *ArXiv* (2015). arXiv: `1508.01991 [cs.CL]`.

[Kob+21] Hayato Kobayashi et al. "A Case Study of In-House Competition for Ranking Constructive Comments in a News Service". In: *Proceedings of the Ninth International Workshop on Natural Language Processing for Social Media.* Ed. by Lun-Wei Ku and Cheng-Te Li. Online: Association for Computational Linguistics, June 2021, pp. 24–35. DOI: `10.18653/v1/2021.socialnlp-1.3`. URL: `https://aclanthology.org/2021.socialnlp-1.3`.

[Kol+20] Varada Kolhatkar et al. *Classifying Constructive Comments.* 2020. arXiv: `2004.05476`.

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems.* 2012. URL: `https://arxiv.org/abs/1409.1556`.

[MC14] Hans K. Meyer and Michael Clay Carey. "In Moderation". In: *Journalism Practice* 8.2 (2014), pp. 213–228. DOI: `10.1080/17512786.2013.859838`. eprint: `https://doi.org/10.1080/17512786.2013.859838`. URL: `https://doi.org/10.1080/17512786.2013.859838`.

[MJ10] Philip M. McCarthy and Scott Jarvis. "MTLD, vocd-D, and HD-D: A validation study of sophisticated approaches to lexical diversity assessment". In: *Behavior Research Methods* 42.2 (2010), pp. 381–392.

[MT13] Saif Mohammad and Peter Turney. "Crowdsourcing a word-emotion association lexicon". In: *Computational Intelligence* 29.3 (Aug. 2013), pp. 436–465. DOI: `10.1111/j.1467-8640.2012.00460.x`.

[Nap+17] Courtney Napoles et al. "Finding Good Conversations Online: The Yahoo News Annotated Comments Corpus". In: *Proceedings of the 11th Linguistic Annotation Workshop.* Ed. by Nathan Schneider and Nianwen Xue. Valencia, Spain: Association for Computational Linguistics, Apr. 2017, pp. 13–23. DOI: `10.18653/v1/W17-0802`. URL: `https://aclanthology.org/W17-0802`.

[ND16] Vlad Niculae and Cristian Danescu-Niculescu-Mizil. "Conversational Markers of Constructive Discussions". In: *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies.* Ed. by Kevin Knight, Ani Nenkova, and Owen Rambow. San Diego, California: Association for Computational Linguistics, June 2016, pp. 568–578. DOI: `10.18653/v1/N16-1070`. URL: `https://aclanthology.org/N16-1070`.

[NVN21] Luan Thanh Nguyen, Kiet Van Nguyen, and Ngan Luu-Thuy Nguyen. "Constructive and Toxic Speech Detection for Open-Domain Social Media Comments in Vietnamese". In: *Advances and Trends in Artificial Intelligence. Artificial Intelligence Practices.* Ed. by Hamido Fujita et al. Cham: Springer International Publishing, 2021, pp. 572–583. ISBN: 978-3-030-79457-6.

[Par+16] Deok Gun Park et al. “Supporting Comment Moderators in Identifying High Quality Online News Comments”. In: May 2016, pp. 1114–1125. DOI: 10.1145/2858036.2858389.

[PN08] Emily Pitler and Ani Nenkova. “Revisiting Readability: A Unified Framework for Predicting Text Quality”. In: *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*. Ed. by Mirella Lapata and Hwee Tou Ng. Honolulu, Hawaii: Association for Computational Linguistics, Oct. 2008, pp. 186–195. URL: https://aclanthology.org/D08-1020.

[RM07] David Reitter and Johanna D. Moore. “Predicting Success in Dialogue”. In: *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*. Ed. by Annie Zaenen and Antal van den Bosch. Prague, Czech Republic: Association for Computational Linguistics, June 2007, pp. 808–815. URL: https://aclanthology.org/P07-1102.

[Sal+17] Haji Saleem et al. “A Web of Hate: Tackling Hateful Speech in Online Social Spaces”. In: (Sept. 2017).

[See+19] Joseph Seering et al. “Designing User Interface Elements to Improve the Quality and Civility of Discourse in Online Commenting Behaviors”. In: Apr. 2019. DOI: 10.1145/3290605.3300836.

[WH12] William Warner and Julia Hirschberg. “Detecting Hate Speech on the World Wide Web”. In: *Proceedings of the Second Workshop on Language in Social Media*. Ed. by Sara Owsley Sood, Meenakshi Nagarajan, and Michael Gamon. Montréal, Canada: Association for Computational Linguistics, June 2012, pp. 19–26. URL: https://aclanthology.org/W12-2103.

[WTD16] Ellery Wulczyn, Nithum Thain, and Lucas Dixon. “Ex Machina: Personal Attacks Seen at Scale”. In: *CoRR* abs/1610.08914 (2016). arXiv: 1610.08914. URL: http://arxiv.org/abs/1610.08914.

Declaration of Academic Integrity / Eidesstattliche Erklärung

Ich erkläre, dass ich die vorliegende Arbeit selbständig, ohne fremde Hilfe und ohne Benutzung anderer als der angegebenen Quellen und Hilfsmittel verfasst habe und dass alle Ausführungen, die wörtlich oder sinngemäß übernommen wurden, als solche gekennzeichnet sind. Mit der aktuell geltenden Fassung der Satzung der Universität Passau zur Sicherung guter wissenschaftlicher Praxis und für den Umgang mit wissenschaftlichem Fehlverhalten vom 31. Juli 2008 (vABlUP Seite 283) bin ich vertraut. Ich erkläre mich einverstanden mit einer Überprüfung der Arbeit unter Zuhilfenahme von Dienstleistungen Dritter (z.B. Anti-Plagiatssoftware) zur Gewährleistung der einwandfreien Kennzeichnung übernommener Ausführungen ohne Verletzung geistigen Eigentums an einem von anderen geschaffenen urheberrechtlich geschützten Werk oder von anderen stammenden wesentlichen wissenschaftlichen Erkenntnissen, Hypothesen, Lehren oder Forschungsansätzen.

Passau, 13. September 2024

Firstname Lastname

I hereby confirm that I have composed this scientific work independently without anybody else's assistance and utilising no sources or resources other than those specified. I certify that any content adopted literally or in substance has been properly identified. I have familiarised myself with the University of Passau's most recent Guidelines for Good Scientific Practice and Scientific Misconduct Ramifications from 31 July 2008 (vABlUP Seite 283). I declare my consent to the use of third-party services (e.g., anti-plagiarism software) for the examination of my work to verify the absence of impermissible representation of adopted content without adequate designation violating the intellectual property rights of others by claiming ownership of somebody else's work, scientific findings, hypotheses, teachings or research approaches.

Passau, 13. September 2024

Firstname Lastname