Effectiveness of Predicting Acceptance from Peer Reviews Using Classical Machine Learning Methods

PLEASANT BALLENGER, CHRISTIAN SODANO, MANNE WENNBERG, ANNA EMSBACH, University of North Carolina at Chapel Hill, USA

Predicting whether a paper will be accepted solely from its peer reviews presents not only an interesting challenge but can be useful in the practical use of gauging the efficacy of peer reviews. While most natural language processing approaches would focus on deep learning methods, this study seeks to understand the performance of classical machine learning methods—namely support vector machines, random forests, and logistic regression—and multiple vectorization techniques. We performed grid search on all models to maximize performance to compare their best possible performance instead of differences in hyperparameters. Our results highlight the importance of vectorization throughout various models. This analysis provides insights into how classic methods compare to comparable literature. The code for this project can be found at https://github.com/Bawllp/562FinalProject.

1 Introduction

As the world's infatuation with machine learning grows, the volume of academic literature and research expands at a rapid pace. This creates an opportunity to explore alternative methodologies and evaluate their effectiveness. Compared to traditional natural language processing methods such as multilayer perceptrons and 1-D convolutional neural networks using BERT, we sought to employ more traditional machine learning methods to compare their performance alongside different vectorization methods. Their performance was, of course, always expected to not be comparable to deep learning models. This paper seeks to systematically compare three classical machine learning methods (support vector machines, random forests, and logistic regression) alongside three vectorization techniques (CountVectorizer(), TfidfVectorizer() using IDF, and TfidfVectorizer() without IDF). First, however, we explore the related literature for this task. Then, to our methodology of data, preprocessing, and models. Our results and discussion sections follow with a discussion of the various limitations to conclude.

2 Related Literature

This study focuses on classifying academic paper acceptance or rejection using peer review text as the sole input. Previous works, such as PeerRead and PeerAssist, have taken different approaches by incorporating paper content along with peer review text into their models [1, 2]. For example, PeerRead primarily used peer reviews to predict sub-attribute scores like clarity and novelty rather than directly predicting acceptance or rejection, which distinguishes our approach [1]. Similarly, while exploring alternative datasets like PeerConf [3], we found them to be unsuitable for our purposes due to insufficient data length and quality.

Our decision to include random forests as one of our models was guided by findings in Thelwall et al. (2022), which demonstrated strong performance of this method in similar tasks [4]. Their study demonstrated that random forest classifiers perform well on a article quality prediction task, which motivated our choice of random forest as one of our models for this similar task. Additionally, we followed their feature extraction approach, namely the use of unigram, bigram, and trigram feature representations. On the other hand, other studies, such as PeerAssist and the Checco et al. 2021 studying AI-assisted peer review opted for convolutional neural networks (CNN) approaches[2,5]. These models perform well when paired with advanced word vector embeddings and large training

Author's Contact Information: Pleasant Ballenger, Christian Sodano, Manne Wennberg, Anna Emsbach, University of North Carolina at Chapel Hill, Chapel Hill, North Carolina, USA.

sets. We chose to focus on classical machine learning methods for interpretability and simplicity, but used some of their choices in preprocessing as basis for our model training–for example, in choosing the top 2,000 most frequent word features.

Both PeerRead and PeerAssist used document length—measured in either pages or sentence counts—as a feature, a concept which we also considered in our feature selection [1,2]. Moreover, the PeerRead dataset [6] and its accompanying paper [1] provided invaluable insights, even though the reviewers' evaluations were primarily used to predict sub-attribute scores rather than accept/reject decisions. During our review of PeerAssist [2], we identified several data quality issues in the ICLR 2017 dataset originally introduced by PeerRead, such as duplicate reviews, improper inclusion of meta-reviews, and author responses mistakenly treated as peer reviews. We addressed these inconsistencies in our preprocessing steps, which are detailed in the Methods section.

By building on these prior works and addressing their limitations, including data quality issues and modeling choices, we aim to present a robust methodology for classifying paper acceptance using peer review text alone.

3 Methodology

3.1 Data

The dataset used in this study was derived from the PeerRead paper dataset, focusing specifically on reviews and metadata from the ICLR 2017 conference [6]. This dataset contains peer reviews, meta-reviews, and decision labels, but required preprocessing to address issues such as duplicates, missing entries, and improperly labeled fields.

3.2 Preprocessing

Preprocessing involved extensive cleaning and preparation of the dataset to ensure high-quality input for the models. The primary goal was to address inconsistencies in the raw data and standardize the format for effective feature extraction and training. The following steps were implemented systematically:

First, all rows with empty reviews were dropped to ensure only meaningful data was used for training. Special characters (e.g., ", . ? ! : ;") and newline characters were removed, while all text was converted to lowercase to eliminate case sensitivity. Text was then lemmatized using the WordNetLemmatizer to standardize words to their base forms.

Duplicate reviews and author responses containing phrases like "we thank" or "reviewer*" were eliminated, as they were deemed irrelevant for prediction purposes. Additionally, the "official decision" field was removed to simulate an editor-less context, aiming for generalization beyond scenarios that rely on explicit decisions. Paper titles were also omitted during feature extraction to avoid overfitting, ensuring the model focused solely on the content of the peer reviews.

Stopwords were handled with care: a supervised approach was applied to modify the default stop-word list, retaining words with contextual importance in reviews (e.g., "seemed," "several," "however"). Punctuation was removed, and text was tokenized and lemmatized to standardize input further. Lastly, reviews shorter than 300 characters were filtered out to maintain dataset quality, as shorter reviews often lacked substantive information.

These preprocessing steps were critical in addressing inconsistencies within the dataset, improving its overall quality, and ensuring the data was well-suited for model training and evaluation.

3.3 Models

We chose three of the most widely employed classical machine learning models to compare their effectiveness on this dataset: a support vector machine, random forest, and logistic regression. They were built to perform binary classification from the peer reviews to predict whether the paper would be rejected or accepted, labeled as "false" and "true" in the dataset, respectively, and mapped to 0 and 1, respectively, for training. We used scikit-learn's SVC(), RandomForestClassifier(), and LogisticRegression

In addition to comparing the three models' performance, we also employed three forms of vectorization for each: CountVectorizer(), TfidfVectorizer() using IDF, and TfidfVectorizer() without IDF.

We trained each model 3 times, once for each of the vectorizations. After the initial training with arbitrary hyperparameters, we ran GridSearchCV() on every version of each model (a total of nine models) to tune the hyperparameters. For the SVM, the parameter grid included 'C': [0.1, 1, 10], 'kernel': ['linear', 'rbf', 'poly', 'sigmoid'], 'gamma': ['scale', 'auto'], 'degree': [2, 3, 4, 5], 'coef0': [0.0, 0.1, 0.5, 1.0]. For the random forest, the parameter grid included 'n_estimators': [25, 40, 50], 'criterion': ['gini', 'entropy', 'log_loss'], 'max_features': ['log2', 'sqrt'], 'max_depth': [2,4], 'bootstrap': [True, False]. Finally, the random forest parameter grid included 'penalty':['l2'],'C':[1, 10, 100], 'max_iter': [100, 200, 500, 1000], 'solver': ['lbfgs', 'saga', 'liblinear']. Each model was then retrained with the best parameters from grid search. Results of five-fold cross validation for each model with the best parameters are presented in Table 1.

4 Results

Benchmark \Metric	Accuracy
Majority decision	0.605
PeerRead*	0.551
PeerAssist** Paper + Review	0.696

SVM

Metric \Feature Representation	Count	TF	TF.IDF
F1	0.516	0.549	0.457
Accuracy	0.635	0.652	0.669

Random Forest

Metric \Feature Representation	Count	TF	TF.IDF
F1	0.524	0.485	0.503
Accuracy	0.589	0.580	0.617

Logistic Regression

Metric \Feature Representation	Count	TF	TF.IDF
F1	0.559	0.540	0.543
Accuracy	0.652	0.648	0.649

Table 1. *Original paper **link**, however accuracy presented is taken from table 3 of **PeerAssist's paper **link** due to concerns over how representative the PeerRead results are due to unaddressed data quality issues raised by the PeerAssist authors that we confirmed. The printed number for PeerRead shows PeerAssist group's reproduction of the PeerRead model on cleaned data.

5 Discussion

Following previous research, we believed that the length of review would be meaningfully predictive of acceptance classification. For this reason we chose to train models using different feature representation methods, one of which being a raw count (unormalized token frequency in document). Additionally, we believed that due to the content of peer reviews being hyper-specific to the individual paper discussed's content, we hypothesized that inverse document frequency encoding may harm the model training by over-weighting terms that are hyper-specific to a single paper. These terms may be repeated many times in the review, allowing it to pass the maximum feature number cutoff which operated on term frequency, but be highly unpredictive. Our data support this hypothesis because, as we can see from Table 1, count-based feature representations had generally higher F1 scores than TF.IDF. Higher accuracies for TF.IDF suggests overfitting.

The best performing model (ranked on F1 score) during cross validation was Logistic Regression using unnormalized count-based feature representation, with a F1 score of 0.559 and an accuracy of 65.2 percent. Because our preproccessing resulted in a different sized training set than PeerRead or PeerAssist, our majority rule accuracy would be 57.6 percent, thus our best model achieved only a 7.6 percent higher accuracy than majority rule. It is important to note, however, that benchmark results failed to achieve much better results even when using each paper's fulltext information to supplement the classification decision.

5.1 Limitations

Underfitting We decided to use a standard dataset discussed in multiple previous works in this research area, however, the extensive cleaning required for this dataset resulted in a serious underfitting issue for our model. After preprocessing, we had 349 papers with a resultant 1179 reviews. The class imbalance at the document level was slightly greater than at the review level (60.2% reject compared to 57.6% reject) due to a higher average review count for accepted documents (3.6 reviews/doc compared to 3.25 reviews/doc). Due to time constraints we were not able to optimize the max number of features hyperparameter in our token vector representation, instead using the top 2000 most common tokens, following previous research **here**. Our resulting models used only 1179 instances to train 2000 features, a clear recipe for underfitting.

Vectorizer optimization Another hyperparameter that was not optimized was the document rarity threshold, which we kept at 1. A larger optimized threshold may have selected features that are more generalizable across reviews, requiring the token to appear frequently in more than one review to be selected as a trainable feature instead of merely frequently.

Text Representation Our text representation included unigrams, bigrams, and trigrams. Upon manual inspection it's clear that there are many highly predictive phrases that would require longer n-grams to represent, for instance, "this was not a very well written paper" in our scheme would mach for "very well written" or "well written" but not "not...well written" or "not very well written". Additionally, we did not successfully enforce manual (supervised) features in our model. Surprisingly, the term "reject" was uncommon (appearing in only 20 reviews), and thus was not included in training due to our maximum feature threshold. However, were we to enforce the training of this feature, it's probable that highly confusing borderline cases that weigh pros and cons equally would easily be resolved by observing the summarizing decision at the end (e.g. "Considering these pros and cons holistically, I recommend to reject this paper".

6 Citations and Bibliography

1. https://doi.org/10.18653/v1/N18-1149
2. https://doi.org/10.1007/978-3-030-91669-5_33
3. https://data.mendeley.com/datasets/wfsspy2gx8/1
4. https://doi.org/10.1162/qss_a_00258
5. https://www.nature.com/articles/s41599-020-00703-8
6. https://github.com/allenai/PeerRead/tree/master/data