Several ways for Vectorization in Topic Classification with Basic Machine Learning Models

Minh Phuc Truong[1], Ha Phuong Linh[2], Nguyen The Quan[1], Hoang Danh Nhat[2], Bui Thi Bich Ngoc[1],

[1]ITE10, [2]ITE15

Abstract

This report investigates Latent Dirichlet Allocation (LDA) for text vectorization in topic classification, comparing it with TF, TF-IDF, and Word2Vec. A range of classifiers (KNN, Linear Regression, Logistic Regression, SVM, Decision Tree, Random Forest, Naive Bayes) and ensemble methods (Stacking, Voting) are evaluated on the 20NG, AG News, and R8 datasets.

1 Introduction

The exponential growth of digital text necessitates robust automated methods for text classification, a cornerstone task in Natural Language Processing (NLP) and machine learning. A critical precursor to effective classification is the transformation of unstructured textual data into meaningful numerical representations—feature vectors—that can be readily processed by machine learning algorithms. The choice of this vectorization strategy can significantly impact model performance, with different methods capturing varying aspects of the textual information.

This report explores and compares several distinct approaches to document vectorization for topic classification tasks. We investigate traditional count-based methods such as Term Frequency (TF) and Term Frequency-Inverse Document Frequency (TF-IDF), which provide strong lexical signals. We then delve into more semantically-aware techniques, including the widely adopted Word2Vec model, which learns distributed word embeddings from context. Furthermore, we examine Latent Dirichlet Allocation (LDA), a generative probabilistic model primarily known for topic modeling. LDA can also be leveraged for feature engineering by representing documents based on their inferred latent topic compositions, specifically through document-topic distributions (LDA-td) and aggregated word-topic information (LDA-wd).

The primary objective of this study is to empirically evaluate the efficacy of these diverse vectorization techniques when paired with a suite of basic machine learning classifiers. The feature sets generated by TF, TF-IDF, Word2Vec, and the two LDA variants will be used to train and test models including K-Nearest Neighbors (KNN), Linear Regression, Logistic Regression, Support Vector Machines (SVM), Decision Trees, Random Forests, and Naive Bayes. To further explore avenues for performance enhancement, we also investigate the impact of ensemble learning, applying Stacking and Voting techniques to combinations of the best-performing base models (Logistic Regression, SVM, and Naive Bayes). All experiments are conducted on three standard benchmark datasets for topic classification: the 20 Newsgroups (20NG) collection, the AG News corpus, and the R8 subset of the Reuters-21578 dataset. This comparative analysis aims to provide insights into the relative strengths, weaknesses, and practical considerations of each vectorization method in the context of topic classification using fundamental machine learning approaches.

2 Background

This section outlines foundational concepts. We first define text classification, then detail Latent Dirichlet Allocation (LDA) for topic modeling and feature extraction.

2.1 Problem Definition: Text Classification

Text classification (or categorization) assigns predefined categories (classes) to text. Formally:

1. A training set $TD = \{(d_i, c_i)\}$, where d_i is a document and c_i its class from a set C.

2. A set of M classes, $C = \{C_1, \ldots, C_M\}$.

The goal is a model $f : \mathcal{D} \to C$, mapping new documents $D \in \mathcal{D}$ to a class in C. Effective document representation is key.

2.2 Latent Dirichlet Allocation (LDA)

LDA (Blei et al., 2003) is a generative probabilistic model for discovering latent topics in document collections. It models documents as mixtures over topics, where topics are word distributions.

2.2.1 Generative Process

For each document m in a corpus M, LDA assumes:

1. Choose topic distribution $\theta_m \sim \text{Dirichlet}(\alpha)$.
2. For each word $w_{m,n}$ in document m:

a) Choose topic $z_{m,n} \sim \text{Multinomial}(\theta_m)$.

b) Choose word $w_{m,n} \sim \text{Multinomial}(\phi_{z_{m,n}})$, where $\phi_k \sim \text{Dirichlet}(\beta)$ is the word distribution for topic k.

α, β are corpus-level parameters; θ_m is per-document topic proportions; ϕ_k is per-topic word distribution; $z_{m,n}$ is word topic assignment; $w_{m,n}$ is the observed word. (See Figure 1 for a graphical model).

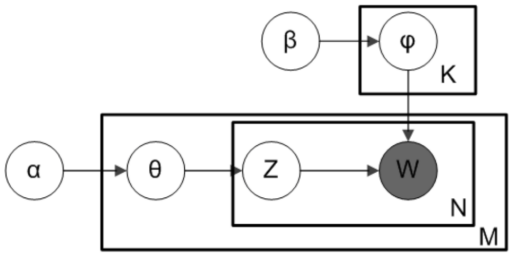

Figure 1: Graphical Model for LDA. (Placeholder)

2.2.2 Inference

The primary inferential challenge in LDA is to determine the latent topic structures (the per-topic word distributions ϕ and per-document topic proportions θ) given the observed documents W. As noted by Blei et al. (Blei et al., 2003), exact computation of the posterior distribution $P(\theta, \phi, Z|W, \alpha, \beta)$ (where Z are the word-topic assignments) is intractable. This intractability arises from the difficulty in computing the marginal likelihood of the documents, $P(W|\alpha, \beta)$, which would require summing over all possible latent variable configurations.

To overcome this, approximate inference techniques are essential. The original LDA paper (Blei et al., 2003) introduced a Variational Bayes (VB) approach, which approximates the true posterior with a simpler, factorized distribution and optimizes its parameters. Another widely used method is Collapsed Gibbs Sampling (Xiao and Stibor, 2010). As described in the provided report (Section 2.3), collapsed Gibbs sampling simplifies the problem by integrating out, or "summing out," certain variables—specifically θ and ϕ in the context of LDA. This allows for direct sampling of the topic assignments Z for each word. The joint distribution over the observed words W and their topic assignments Z, after marginalizing out θ and ϕ, can be expressed as:

$$P(Z, W; \alpha, \beta) = \int_{\theta} \int_{\phi} P(W, Z, \theta, \phi; \alpha, \beta)\, d\phi\, d\theta \tag{1}$$

In this MCMC approach, the topic assignment $z_{m,n}$ for each word $w_{m,n}$ in document m is iteratively sampled from its conditional distribution given all other current topic assignments and the observed words. After a sufficient number of iterations (burn-in), the samples of Z approximate its true posterior distribution, from which estimates of θ and ϕ can be derived. Our work leverages an LDA implementation that relies on one such established approximate inference technique for model estimation.

3 Methodology

3.1 Term Frequency (TF):

Term Frequency, denoted $\text{tf}(t, d)$, represents a document d as a vector of the raw counts of each term t (or word) present in that document. Each dimension in the vector corresponds to a unique term in the corpus vocabulary.

3.2 Term Frequency-Inverse Document Frequency (TF-IDF):

Previous works refine TF by weighting terms based not only on their frequency within a document but also on their rarity across the entire document corpus D. The TF-IDF score for a term t in document d from corpus D is calculated as:

$$\text{tfidf}(t, d, D) = \text{tf}(t, d) \cdot \text{idf}(t, D) \tag{2}$$

Where $\text{tf}(t, d)$ is the term frequency as defined above, and $\text{idf}(t, D)$ is the inverse document frequency, calculated as:

$$\text{idf}(t, D) = \log\left(\frac{N}{|\{d' \in D : t \in d'\}|}\right) \tag{3}$$

Here, N is the total number of documents in the corpus D, and $|\{d' \in D : t \in d'\}|$ is the number of documents in the corpus that contain the term t. The logarithm helps to dampen the effect of very rare terms.

3.3 Word Embeddings (GloVe):

We utilize pre-trained GloVe (Global Vectors for Word Representation) embeddings (Pennington et al., 2014). These techniques use neural networks to learn dense, continuous vector representations for words from large text corpora, capturing semantic relationships (e.g., words with similar meanings tend to have similar vector representations). To obtain a feature vector for an entire document d, we calculate the element-wise mean of the GloVe embedding vectors of all the words present in that document. As per the reference, we use pre-trained GloVe vectors where each word is represented by a 100-dimensional feature vector.

3.4 LDA Topic Distributions (LDA-td):

After training an LDA model, each document d is represented by its inferred topic distribution. The feature vector f for d is:

$$f[i] = p(\text{topic}_i|d), \tag{4}$$

where $1 \leq i \leq K$ (K is a number of topics).

3.5 LDA Averaged Word-Topic Probabilities (LDA-wd):

The feature vector f for document d is computed by averaging, for each topic i, the probabilities of the words in d given that topic.

$$f_i = \frac{1}{N_d} \sum_{j=1}^{N_d} \phi_{i,w_j^d} \tag{5}$$

where N_d is the number of words in document d, w_j^d is the j-th word, and ϕ_{i,w_j^d} is $p(w_j^d|\text{topic}_i)$ from the LDA model's topic-word distributions. The vector length is K.

For LDA-based methods, the number of topics K is a hyperparameter tuned during experimentation.

4 Experiments

4.1 Datasets

This report presents a machine learning classification project focused on classifying text data into predefined categories. The study analyzes a dataset of textual entries, examining class distributions, text length, and word count characteristics. Multiple classification models are evaluated to identify the most effective approach for categorizing the text. Key findings include insights into class imbalances, textual feature distributions, and model performance metrics, demonstrating the application of machine learning to text classification tasks.

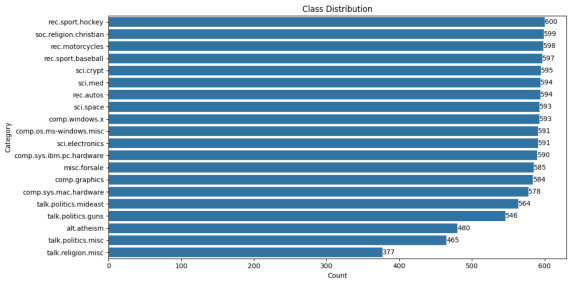

Figure 2: Class distribution of the 20 New Groups dataset.

The 20 New Groups dataset comprises *11314* samples as text feature, representing a classification problem with *20* classes. The dataset structure is summarized in Table 2.

Table 1: Summary of the 20 New Groups Dataset

Property	Value
Number of Samples	11314
Number of Classes	20
Feature Types	Text
Target Variable	Class Label

The R8 dataset comprises *4937* samples as text feature, representing a classification problem with *8* classes. The dataset structure is summarized in Table 2.

Table 2: Summary of the R8 Dataset

Property	Value
Number of Samples	4937
Number of Classes	8
Feature Types	Text
Target Variable	Class Label

Key preprocessing steps included converting text to lowercase, removing stopwords, eliminating extra whitespace, and dropping duplicate entries to ensure clean and consistent data for model training.

4.2 Machine Learning Models

This study employs a diverse range of supervised machine learning algorithms to perform the topic classification task using the features generated by the aforementioned vectorization techniques. Additionally, two ensemble methods are utilized to

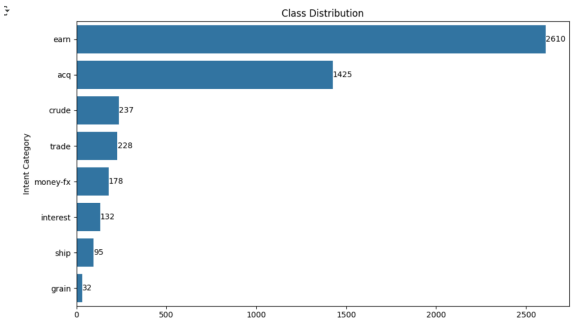

Figure 3: Class distribution of the R8 dataset.

potentially enhance classification performance by combining the predictions of selected base models.

Base Classifiers The following individual classification models are implemented and evaluated:

- **K-Nearest Neighbors (KNN):** A non-parametric, instance-based learning algorithm that classifies a new data point based on the majority class among its k nearest neighbors in the feature space. The distance metric (e.g., Euclidean) and the value of k are key hyperparameters.

- **Linear Regression:** While primarily designed for regression tasks, Linear Regression can be adapted for binary classification by applying a threshold to its output. For multi-class problems, strategies like One-vs-Rest (OvR) can be employed. However, it is generally less suited for classification compared to models specifically designed for it.

- **Logistic Regression:** A linear model that uses a logistic (or sigmoid) function to model the probability of a binary outcome. For multi-class classification, it is commonly extended using strategies like One-vs-Rest (OvR) or by fitting a multinomial logistic regression model. It is widely used due to its interpretability and efficiency.

- **Support Vector Machines (SVM):** A powerful classification algorithm that aims to find an optimal hyperplane that best separates data points of different classes in a high-dimensional feature space. SVMs can use various kernel functions (e.g., linear, polynomial, RBF) to handle non-linearly separable data and are known for their effectiveness, particularly with high-dimensional data like text features (Joachims, 1999; Tong and Koller, 2001).

- **Decision Tree:** A non-parametric supervised learning method that creates a model predicting the value of a target variable by learning simple decision rules inferred from the data features. It is represented as a tree structure where internal nodes test features, branches represent outcomes, and leaf nodes assign class labels.

- **Random Forest:** An ensemble learning method that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) of the individual trees. Random Forests generally exhibit higher accuracy and are more robust to overfitting than single decision trees.

- **Naive Bayes:** A family of probabilistic classifiers based on applying Bayes' theorem with a "naive" assumption of conditional independence between every pair of features given the class variable. Despite this strong assumption, Naive Bayes classifiers (e.g., Multinomial Naive Bayes, Gaussian Naive Bayes) often perform well in text classification tasks, especially with high-dimensional sparse features like TF-IDF (McCallum and Nigam, 1998).

Ensemble Methods To leverage the strengths of multiple classifiers and potentially improve overall predictive performance, two ensemble learning techniques are employed. In this study, the ensembles are constructed using Logistic Regression, Support Vector Machines, and Naive Bayes as the base estimators, as these often show strong individual performance or complementary strengths.

- **Voting Classifier:** This ensemble method combines predictions from multiple base models.
  - *Hard Voting:* The predicted class label is the one that receives the majority of votes from the base classifiers.
  - *Soft Voting (if applicable):* If base classifiers can output class probabilities, soft voting predicts the class label by averaging these probabilities and selecting

the class with the highest average probability. This often leads to better performance if the probability estimates are well-calibrated.

- **Stacking Classifier (Stacked Generalization):** Stacking involves training a "meta-classifier" to combine the predictions of several "base classifiers." The process typically involves:

  1. Splitting the training data into folds (e.g., k-fold cross-validation).
  2. Training each base classifier on $k-1$ folds.
  3. Making predictions with each trained base classifier on the remaining fold (out-of-fold predictions).
  4. These out-of-fold predictions from all base classifiers are then used as input features to train the meta-classifier.
  5. The base classifiers are then trained on the full original training set.

  The final prediction is made by the trained meta-classifier using the predictions of the base classifiers (trained on the full set) on new, unseen data. Stacking can capture more complex relationships between base model predictions than simple voting.

These models, both individual and ensembled, provide a comprehensive basis for evaluating the effectiveness of different text vectorization strategies.

4.3 Results and Analysis

This section presents a comparative analysis of text vectorization methods based on classification accuracy across the 20 Newsgroups and R8 datasets. Key findings from Table 3 (20 Newsgroups) and Table 4 (R8) are discussed.

Performance on 20 Newsgroups and R8 Datasets On the 20 Newsgroups dataset (Table 3), traditional TF-IDF representations demonstrated superior performance, achieving peak accuracies with SVM (71.04%) and Logistic Regression (67.87%). Term Frequency (TF) also performed well, particularly with ensemble methods (Stacking: 63.02%). Word2Vec (GloVe) embeddings yielded moderate results (Logistic Regression: 58.74%). LDA-based features (LDA-td: 48.65% with Logistic Regression; LDA-wd: 45.90% with Logistic Regression) lagged behind TF-IDF, potentially due to the dataset's topical diversity and keyword-driven class distinctions.

Conversely, on the R8 dataset (Table 4), LDA-derived document-topic distributions (LDA-td) emerged as the most effective vectorization strategy, achieving top accuracies with SVM (98.29%), Logistic Regression (95.88%), and the Stacking ensemble (98.02%). This suggests a strong alignment between LDA's latent topics and R8's more focused class structure. TF-IDF and TF remained highly competitive (e.g., TF-IDF with SVM: 97.44%). Word2Vec showed improved performance on R8 (Logistic Regression: 94.70%) compared to 20 Newsgroups. LDA-wd performance, while better on R8 than 20NG, still trailed other leading methods. The high overall accuracies on R8 indicate its relative ease of classification.

Comparative Discussion The contrasting results highlight several key insights:

1. **Dataset-Specific Efficacy:** Optimal vectorization is dataset-dependent. TF-IDF excelled on the diverse 20 Newsgroups, while the semantically richer LDA-td was superior for the more thematically coherent R8 dataset.

2. **Robustness of TF-IDF/TF:** These traditional methods remain highly effective baselines and often achieve state-of-the-art performance, especially with strong classifiers like SVM.

3. **LDA-td's Potential:** LDA-td can yield excellent results when dataset topics align with its latent structure and when tuned appropriately, as seen with R8.

4. **Limited Utility of LDA-wd and Averaged Word2Vec:** LDA-wd and simple averaging of Word2Vec embeddings generally underperformed more specialized or traditional techniques in these classification tasks, suggesting potential loss of discriminative information.

5. **Ensemble Benefits:** Ensemble methods often provided incremental improvements, underscoring their value in robust model development.

These findings emphasize the importance of empirical evaluation and tailoring feature engineering to dataset characteristics.

Table 3: Comparison of Vectorization Methods by Accuracy (%) on the 20 Newsgroups Dataset. Each cell shows the accuracy for the given vectorization method and classifier. The highest accuracy for each classifier (column-wise) is bolded.

Vectorization Method	**Classifier Accuracy (%)**							
	KNN	**SVM**	**LogReg**	**DTree**	**RForest**	**NBayes**	**Voting**	**Stacking**
Term Frequency (TF)	16.79	57.97	60.11	40.45	58.97	55.74	61.82	**63.02**
TF-IDF	10.85	**71.04**	**67.87**	39.55	58.92	**60.82**	**69.82**	68.93
Word2Vec (GloVe)	**49.12**	54.97	58.74	30.83	52.52	46.94	59.55	57.43
LDA Topics (LDA-td)	37.09	43.47	48.65	36.44	47.53	47.68	48.30	43.55
LDA Words (LDA-wd)	32.79	28.90	45.90	28.27	43.44	41.71	43.15	37.06

Table 4: Comparison of Vectorization Methods by Accuracy (%) on the R8 Dataset. Each cell shows the accuracy for the given vectorization method and classifier. The highest accuracy for each classifier (column-wise) is bolded.

Vectorization Method	**Classifier Accuracy (%)**							
	KNN	**SVM**	**LogReg**	**DTree**	**RForest**	**NBayes**	**Voting**	**Stacking**
Term Frequency (TF)	87.53	96.48	96.21	**91.00**	92.78	**95.84**	97.26	97.03
TF-IDF	82.91	97.44	94.93	90.45	92.60	83.65	94.84	97.44
Word2Vec (GloVe)	92.46	94.15	94.70	83.28	92.14	72.41	90.09	92.22
LDA Topics (LDA-td)	**93.33**	**98.29**	**95.88**	89.58	**94.34**	84.65	**98.79**	**98.02**
LDA Words (LDA-wd)	92.01	87.76	94.70	86.16	92.78	91.87	94.66	93.70

5 Conclusion

This study presented a comprehensive comparison of various vectorization techniques—TF, TF-IDF, Word2Vec, LDA-td, and LDA-wd—for topic classification using basic machine learning models. Through experiments on benchmark datasets (20 Newsgroups and R8), we demonstrated that the performance of vectorization methods is highly dataset-dependent. Traditional methods like TF-IDF performed best on the diverse and keyword-rich 20 Newsgroups dataset, whereas LDA-based representations, particularly LDA-td, excelled on the more focused R8 dataset. Ensemble methods, especially Stacking and Voting, consistently provided incremental performance gains across models, further affirming their utility in classification tasks. These results underscore the importance of selecting appropriate text representation strategies tailored to dataset characteristics, and reaffirm the effectiveness of classical machine learning models when paired with well-chosen features.

6 Limitations

Despite the informative findings, this study has several limitations. First, the investigation was limited to classical machine learning models and did not explore recent advancements in deep learning, such as transformer-based architectures (e.g., BERT), which have demonstrated state-of-the-art performance in text classification. Second, the Word2Vec representation was computed using a simple average of word vectors, potentially losing important contextual and syntactic information. More sophisticated techniques, such as attention mechanisms or weighted averaging based on term importance, could yield improved results. Third, while LDA showed strong performance on R8, its effectiveness is sensitive to the choice of the number of topics and may not generalize well without careful hyperparameter tuning. Lastly, the datasets used are in English and relatively clean; real-world applications often involve noisy, multilingual, or domain-specific text that could impact performance differently.

References

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3(Jan):993–1022.

Thorsten Joachims. 1999. Transductive inference for text classification using support vector machines. In *Proceedings of the Sixteenth International Conference on Machine Learning (ICML '99)*, pages 200–209, Bled, Slovenia.

Andrew McCallum and Kamal Nigam. 1998. A comparison of event models for Naive Bayes text classification. In *AAAI-98 Workshop on Learning for Text Categorization*, volume 752, pages 41–48. AAAI Press.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Simon Tong and Daphne Koller. 2001. Support vector machine active learning with applications to text classification. *Journal of Machine Learning Research*, 2(Nov):45–66.

Han Xiao and Thomas Stibor. 2010. Efficient collapsed Gibbs sampling for latent Dirichlet allocation. In *Proceedings of the 2nd Asian Conference on Machine Learning (ACML)*, volume 13 of *JMLR Workshop and Conference Proceedings*, pages 296–310.