

Improving Short Text Classification with Semi-Supervised Learning

Emanuela Mitreva ¹

¹ *Institute of Mathematics and Informatics, Bulgarian Academy of Science, Sofia, Bulgaria*

Abstract – Classifying short text poses significant challenges, especially when only a small portion of the data is labeled. Although supervised learning methods can achieve high accuracy on labeled datasets, overfitting often occurs, leading to poor generalization on unseen data. The primary concern is the significant decline in model accuracy when transitioning from a small labeled dataset to a larger set of unlabeled data. To address this issue, the article explores a few strategies: semi-supervised learning, k-means clustering, and stacking of classifiers. Semi-supervised learning leverages both labeled and unlabeled data by iteratively generating pseudo-labels and assigning confidence-based weights, thus expanding the training set without amplifying noise, because greater weight is assigned to more reliable labels. K-means clustering is similarly employed to partition unlabeled data into clusters, and pseudo-labels are inferred via majority voting based on similarity to labeled examples. These augmented methods are shown to mitigate the decline in accuracy observed when transitioning from the small, labeled dataset to previously unseen, unlabeled data. Although stacking multiple classifiers and methods to mitigate imbalanced classes did not improve performance, there are notable mentions and the lack of improvement was likely due to uniformly propagating errors from models that already struggle on unseen samples.

Keywords – Supervised-learning, unsupervised-learning, semi-supervised learning.

DOI: 10.18421/TEM151-80

<https://doi.org/10.18421/TEM151-80>


Corresponding author: Emanuela Mitreva,
Institute of Mathematics and Informatics, Bulgarian Academy of Science, Sofia, Bulgaria
Email: emitreva@gmail.com

Received: 10 June 2025.

Revised: 26 September 2025.

Accepted: 15 October 2025.

Published: 27 February 2026.

 © 2026 Emanuela Mitreva; published by UIKTEN. This work is licensed under the Creative Commons Attribution-NonCommercial-NoDerivs 4.0 License.

The article is published with Open Access at <https://www.temjournal.com/>

1. Introduction

The task of classifying short texts has received significant attention in recent years, as their limited length and lack of context pose difficulties for feature extraction and model generalization [1], [2], [3]. Traditional supervised methods, such as Naive Bayes, logistic regression, and support vector machines, have been widely applied to text classification tasks and can perform well when sufficient labeled data are available [2], [4]. However, when only small, labeled datasets exist, these methods tend to overfit and show poor generalization to unseen data, a limitation that is particularly pronounced for short texts, containing a few words, providing insufficient context for reliable categorization.

To mitigate these challenges, a range of alternative approaches has been explored. Unsupervised methods such as clustering enable the discovery of latent structure in short-text datasets [5], [6], [7]. However, because clustering does not incorporate label information, it generally produces inferior classification performance compared with supervised methods.

In contrast, semi-supervised learning (SSL) has emerged as a promising compromise, combining a small set of labeled examples data to improve model robustness and generalization [8], [9]. Semi-supervised learning, as indicated by its name, lies conceptually between supervised and unsupervised learning. The semi-supervised learning aims to improve the classification of datasets that have both labeled and unlabeled data. By combining a small amount of labeled data with a large pool of unlabeled data, semi-supervised learning can significantly improve model accuracy and generalization, making it an attractive solution for many real-world applications. SSL techniques such as self-training and pseudo-labeling iteratively assign labels to unlabeled samples with high classifier confidence, thereby expanding the effective training set. These approaches have demonstrated substantial improvements across domains ranging from cybersecurity [10] to natural language processing [11] and computer vision [12], [13], [14], [15].

Nevertheless, applying SSL to very short texts introduces unique challenges, particularly the risk of error propagation from noisy pseudo-labels and the heightened susceptibility to overfitting when the labeled dataset is extremely small.

Despite these advances, relatively little work has addressed the classification of extremely short tasks, where label scarcity and text brevity coincide. This gap raises the following research question: This work examines the extent to which semi-supervised learning strategies enhance the accuracy of short-text classification with limited labeled data and compares their effectiveness against alternative methods such as clustering and classifier stacking. Addressing this question is important both theoretically—because it advances the understanding of SSL behavior under extreme conditions of text sparsity—and practically—because accurate classification under these constraints is essential for adaptive learning systems designed to provide targeted support to learners.

This study makes three main contributions. First, it provides a systematic evaluation of several classifiers for short-text classification under conditions of limited labeled data, identifying those that perform reliably in this setting. Second, it examines clustering and classifier stacking as alternative augmentation strategies and shows their limitations. Third, it introduces a hybrid approach that builds on the best-performing supervised classifiers and enhances them through semi-supervised learning with dynamic thresholding and confidence-weighted pseudo-labels, yielding an improvement of approximately eight percentage points in accuracy on unseen data. Collectively, these contributions advance research on semi-supervised short-text classification and offer a practical framework for adaptive learning systems and related applications that depend on reliable categorization of short texts.

2. Initial Model

Before exploring hybrid or semi-supervised strategies, an initial supervised classification model was implemented as a baseline. The study began with preprocessing of both labeled and unlabeled samples, followed by model training, evaluation, and eventual refinement through semi-supervised learning.

Preprocessing began with the removal of HTML tags, hyperlinks, and other formatting artifacts that could introduce noise and diminish classifier performance. Entries containing fewer than three words were discarded, since extremely brief texts tend to undermine classifier accuracy.

Although the literature suggests that models perform optimally on texts of ten words or more, many of the tasks' descriptions fell below this threshold, necessitating a lower cutoff.

After filtering out overly short entries, common stop words were eliminated—names, prepositions, which carry little semantic weight—to focus on more informative tokens. This thorough cleaning phase rendered the dictionary suitable for computational analysis and laid the foundation for meaningful feature extraction.

Once preprocessing was complete, the remaining documents were transformed into numerical representations using three standard vectorization techniques from scikit-learn – CountVectorizer, HashingVectorizer and TfidfVectorizer. However, as testing indicated, the most promising one was TfidfVectorizer, which computes Term Frequency–Inverse Document Frequency (TF–IDF) weights, thereby amplifying the influence of terms that are frequent within a given document yet rare across the entire corpus. In conjunction with TF–IDF weighting, the n-gram range was extended from unigrams to bigrams and trigrams to capture short-phrase patterns that could not be represented by unigrams alone. Given the brevity of the texts, where one or two words were typically sufficient to differentiate categories, unigrams and bigrams were selected. This enhances the model's ability to understand the underlying context and relationships within textual data, potentially improving classification accuracy.

With vectorized representations in hand, a diverse set of classifiers known to be effective in short-text classification was evaluated [2], [3]. Traditional methods such as the Naive Bayes are expected to perform well. However, the Naive Bayes classifier assumes attribute independence, which means it works optimally when attributes are truly independent but may degrade when there is some dependence among them [4]. Given the short length of the texts in the dataset, dependencies between certain words are more likely to be random. Based on this assumption, classifiers using this algorithm were expected to produce satisfactory results. Additionally, considering the extensive research outlined in [2], the Support Vector Machine (SVM) was also considered as a potential classification candidate. Therefore, based on those results, the final selection included RidgeClassifier (Least Squares SVM), RidgeClassifierCV (Least Squares SVM), SGDClassifier (SVM), LinearSVC (SVM), PassiveAggressiveClassifier and MultinomialNB (Naive Bayes).

Table 1. Accuracy tests with *TfidfVectorizer*

classifier	%	random state	accuracy
LinearSVC	10	12, 13, 14, 17, 21, 24, 27, 28, 32	100%
LinearSVC	20	21	100%
LinearSVC	30	35	97.62%
LinearSVC	20	11, 13, 24, 25, 27, 35, 41	96.43%
LinearSVC	30	13, 27, 41	95.24%
Multinomial NB	10	14	100%
Passive Aggressive Classifier	10	13, 14, 17, 21, 24, 27, 28, 32	100%
Passive Aggressive Classifier	30	35	97.62%
Passive Aggressive Classifier	20	11, 13, 21, 24, 25, 27, 28, 41	96.43%
Passive Aggressive Classifier	30	21, 27, 41	95.24%
RidgeClassifier	10	12, 13, 14, 21, 24, 27, 28, 32	100%
RidgeClassifier	30	35	97.62%
RidgeClassifier	20	13, 21, 24, 25, 27, 35, 41	96.43%
RidgeClassifier	30	13, 21, 27, 41	95.24%
RidgeClassifier CV	10	12, 13, 14, 17, 21, 24, 27, 28, 32, 38	100%
RidgeClassifier CV	20	24	100%
RidgeClassifier CV	30	35	97.62%
RidgeClassifier CV	20	11, 12, 13, 21, 25, 27, 28, 35, 41	96.43%
RidgeClassifier CV	30	13, 21, 27, 41, 42	95.24%
SGDClassifier	10	12, 16, 17, 21, 24, 27, 32, 40	100%
SGDClassifier	20	11, 21, 27, 35, 41	96.43%
SGDClassifier	30	21, 35, 41	95.24%

Each classifier underwent hyperparameter tuning via grid search on the labeled set, with parameters such as regularization strength, learning rate, and n-gram range optimized. On the labeled data, LinearSVC, RidgeClassifierCV, and PassiveAggressiveClassifier frequently exceeded 90% accuracy—and in some configurations even reached 100%—demonstrating that short-text tasks become linearly separable once appropriately vectorized and cleaned as shown in Table 1, where a small portion of results were provided.

However, when the classifiers trained on the labeled dataset were evaluated against previously unseen data, overall accuracy declined markedly—from near-perfect performance to 84.57%—a reduction of more than fifteen percentage points. This pronounced generalization gap can be attributed to two principal factors.

First, the models exhibited overfitting and possibly the class distributions underlying the unlabeled data deviated substantially from those of the labeled set, resulting in a disproportionate misclassification rate for underrepresented categories. Although rigorous preprocessing—comprising the removal of HTML tags, enforcement of a minimum document length, and elimination of stop words—combined with robust vectorization methods (notably TF-IDF enhanced with n-gram features) enables linear classifiers to achieve exemplary performance on labeled short-text inputs, a purely supervised training paradigm remains susceptible to substantial performance degradation when confronted with novel, unlabeled samples.

Consequently, subsequent sections review approaches designed to mitigate these issues and present a modified version of the model that improves accuracy on unseen data.

3. Overview of Approaches

This section of the article explores three main approaches to augment the model – semi-supervised learning, k-means and stacking of classifiers. In addition, this study examines whether class imbalance can be mitigated or whether the dataset is sufficiently balanced to allow the augmented classifier to achieve satisfactory performance. Semi-supervised learning was expected to yield the best results, with potential enhancement through the application of k-means.

3.1. Semi-supervised Learning

Improving classification with semi-supervised learning (SSL) is a significant area of research, particularly due to the challenges associated with acquiring large amounts of labeled data. SSL leverages both labeled and unlabeled data to enhance model performance, making it a cost-effective and efficient approach in various domains.

SSL offers several advantages over traditional supervised learning:

- **Reduced Need for Labeled Data:** SSL reduces the need for large amounts of labeled data, which can be expensive and time-consuming to obtain [8], [9].
- **Improved Model Generalization:** By leveraging unlabeled data, SSL can improve the generalization performance of models, especially in scenarios where labeled data is limited [10], [11], [12].
- **Handling Small Labeled Datasets:** SSL is particularly useful when the labeled dataset is small. It can effectively utilize the information in unlabeled data to improve model performance [8], [9].

One of the key techniques used in semi-supervised learning is self-training, where a model is initially trained on labeled data and then iteratively improves its performance by generating pseudo-labels for unlabeled data and incorporating them into the training process. Therefore, the classifier (model, based only on supervised learning) is first trained on the labeled data and then pseudo-labels are generated for the unlabeled data. The main idea of the semi-supervised learning is that it checks whether the text is similar or close to other texts. The premise is that if the texts are close, then the labels are close. Therefore, the method generates pseudo-labels that are close to a label, with which the texts are close [13], [14]. By leveraging the strengths of both labeled and unlabeled data, self-training can effectively bridge the gap between limited labeled data and the vast amounts of unlabeled data available, thus enhancing the robustness of machine learning models across various domains.

3.2. K-means

Clustering is a fundamental technique in unsupervised machine learning aimed at organizing data into meaningful groups, or clusters, based on inherent patterns or similarities. The objective is to group data that are more similar to each other than to those in other clusters [21]. Clustering algorithms partition the data into clusters such that points within the same cluster are as similar as possible, while points in different clusters are dissimilar [21]. This approach facilitates data exploration, pattern discovery, and segmentation, enabling insights into the underlying structure of the dataset without the need for predefined labels.

A widely used clustering method is the use of centroids (k-means) [5], [6]. This popularity is attributable to the simplicity - requiring no prior training or separation into training and test sets as well as its efficiency [5], [7]. Considering a set of n objects that must be partitioned into a predefined number k of non-overlapping, homogeneous clusters ($k < n$) [19]. The clusters are formed in such a way that the objects in one cluster are similar and the remaining objects in the other clusters are different [19], [20].

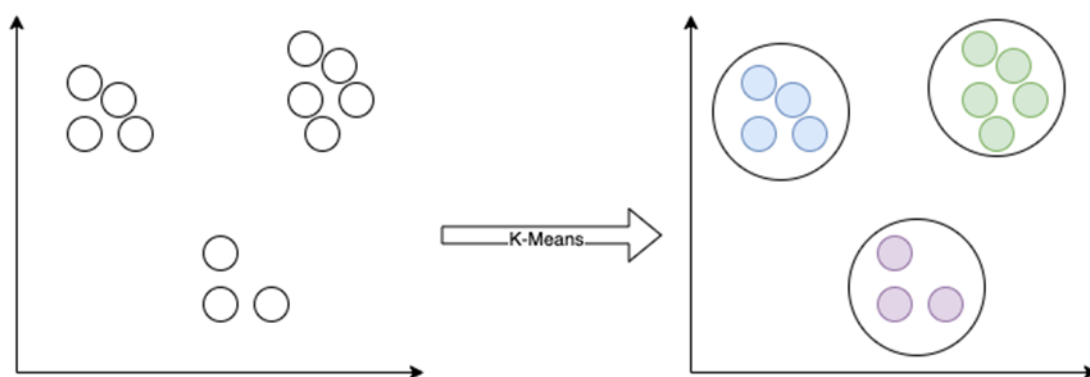


Figure 1. k-means

The algorithm works as follows (also shown in Figure 1):

1. Initially, the number of clusters K is determined and K initial cluster centres are randomly assigned [6].
2. Each data point is assigned to the cluster whose centre is closest to it based on a distance metric (typically Euclidean distance) [6].
3. Recalculate the cluster centres by taking the average of all data assigned to each cluster [6].
4. Each data point is assigned to the cluster whose centre is closest to it based on a distance metric (typically Euclidean distance) [6], [7].

Let $x = \{x_1, x_2, \dots, x_n\}$ denote data and S_1, S_2, \dots, S_k are the clusters, where as already mentioned $n > k$, a $\mu_1, \mu_2, \dots, \mu_k$ are the mean values of the data in the cluster S_i [19], [20]. Then the criterion for minimizing the mean squared error can be described by the following formula [22], [23]:

$$\operatorname{argmin}_S \sum_{i=1}^k \sum_{x \in S_i} \|x - \mu_i\|_2^2 \quad (1)$$

Unfortunately, the efficiency of this method depends largely on the number k or the number of clusters [12].

3.3. *Stacking of Classifiers*

This approach remains the least explored, as early investigations demonstrated limited effectiveness in enhancing the accuracy of the baseline model. The idea of this method is checking which classifiers provide good accuracy on the initial labeled dataset and to choose several that are accurate and consequently then to train several of them and when new data are to be classified, the predictions of those previously chosen classifiers are all considered. Unfortunately, this failed to improve the accuracy of the model and in fact decreased it. The probable cause is that the chosen classifiers are evenly behaving well on the labeled data and are degrading on the new data, thus stacking additional erroneous labels failed to improve the accuracy of the model.

3.4. *Mitigating Imbalanced Classes*

One possible problem that was expected would make the classifier perform worse on new data was the fact that labeled data is not perfectly balanced – some categories had 6-7 items and others had much more. Mitigating imbalanced classes usually is a critical step in scenarios where one or more classes are significantly underrepresented relative to others in a classification task. Without addressing this imbalance, a model trained on the raw data may develop a strong bias toward the majority class, resulting in diminished performance when predicting minority classes. Approaches for mitigating this issue can be broadly divided into data-level and algorithm-level strategies. Data-level methods involve modifying the training distribution, for instance by undersampling the majority class, oversampling the minority class, or synthesizing new instances of the minority class, as in SMOTE, the Synthetic Minority Over-sampling Technique. Algorithm-level techniques, on the other hand, adjust the learning process itself—examples include incorporating class weights, applying cost-sensitive learning, or using specialized loss functions such as focal loss. By redistributing class proportions or assigning higher penalties to minority misclassifications, these interventions ensure that the model devotes sufficient attention to the underrepresented classes, ultimately improving predictive performance and promoting a more balanced classification outcome.

4. **Modified Approach**

The research on k-means was promising, however, unsupervised learning alone clearly lacks effectiveness and its application is limited [7], therefore, it is intended to be applied solely as an augmentation to the initial supervised learning classifier, and only in cases where it demonstrates significant improvement.

However, as mentioned, some tasks from different categories were highly similar, which limited the potential for significant improvement of the accuracy of the model. Indeed, testing indicated a slight improvement, but only when there was a category with poor-quality data (the labeled data was not a good representation of the category) – this was later fixed, which resulted in insignificant improvement. Based on this outcome it was determined that for the improvement model k-means will not be used, unless there is clear evidence of noisy or low-quality data. The computational resources that are needed for k-means did not justify the marginal improvement observed.

Another approach that was used was stacking classifiers. The initial tests on the labeled data showed that several classifiers have good accuracy on the data ($\geq 95\%$) and those were selected to be used. Unfortunately, those tests also proved ineffective— the results indicate that accuracy on the unlabeled was dropping for all classifiers, because similar errors were consistently propagated across classifiers.

The final method investigated was semi-supervised learning. Semi-supervised learning has been shown to yield good results in improving text classification [20]. It is a powerful approach that leverages both labeled and unlabeled data to enhance the performance of classification models, particularly when obtaining labeled examples is costly or time-consuming [21]. As was exactly the case with the setup, this represents the most promising approach. Based on the results of the testing, it turned out that indeed semi-supervised learning provided the best improvement. Thus, the following section provides the details about the improved model that at first uses supervised learning on the labeled data to train the model and then augments it with semi-supervised learning on the unlabeled data.

4.1. Model Architecture

Based on the results and analysis of the training on the labeled set, a few classifiers with high accuracy are chosen to be first trained on the labeled data. Consequently, the initial model classifies the data by using those classifiers, then to improve the accuracy on the new data, semi-supervised learning is used. Semi-supervised learning operates on the principle that textual similarity implies label similarity. In practice, this means that when an unlabeled text is found to be close to labeled examples in the feature space, it can be assigned a pseudo-label corresponding to the majority or most confident label of its neighbors. Thus, the method generates pseudo-labels that are close to a label, with which the texts are close [13], [14].

As a result, pseudo-labels for all the unclassified data would be generated, however, it is not advisable to use all pseudo-labels that were generated, because some might be too far from the actual label. Once pseudo-labels are generated, only the ones with high probability of being correct should be filtered and added to the set with labeled data. However, what is to be decided is which pseudo-labels are to be considered good enough to be added to the training set of labeled data. The simplest approach is to use a high hard-coded threshold for predictions such as 0.95 and discard all other predictions. Nonetheless, this is a less effective approach as pointed out in [15], because if the labeled data is a small set, it is possible to overfit on the data and that predictions with lower credibility are more valuable for the classifier – both to mitigate overfitting and to improve the accuracy even more. This approach can enhance the training set, but it may accumulate errors over time, leading to biased predictions [16], [17].

Thus, the approach that was chosen was to use dynamic threshold that starts with the most confident labels (the upper limit is set to a very high threshold - 0.99) and then gradually decreases to incorporate some of the less confident labels reaching a lower limit of approximately 0.80. This partially mitigates the problem of overfitting on the initial data that is limited. This iterative process not only helps in refining the model's understanding of the data distribution but also enables it to learn from potentially valuable information hidden within the unlabeled examples, ultimately leading to better decision-making capabilities.

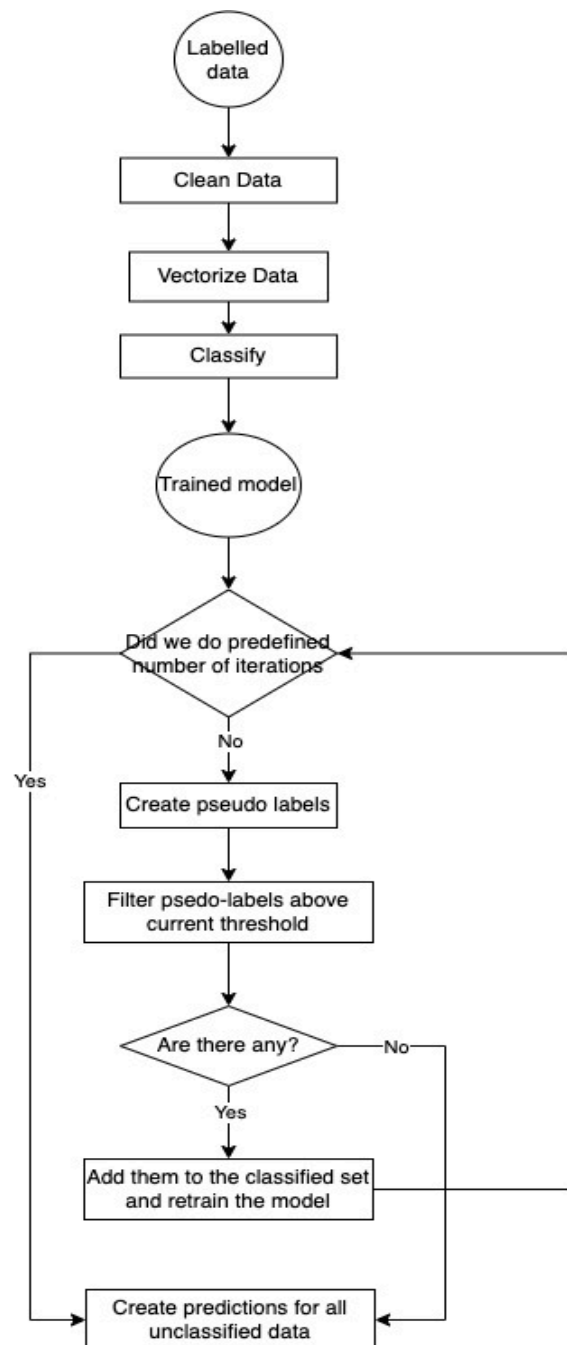


Figure 2. Modified algorithm

One other improvement that is incorporated in the modified model is using the confidence scores as sample weights for the new pseudo-labeled examples [18]. As a next step the model is retrained to take it into account also the new data with the pseudo-label. The final modified algorithm is presented in Figure 2.

5. Results and Discussion

To evaluate the two models and to show the improvement of the modified version, the same tests were conducted with the same data and hyper parameters on both.

Application of the semi-supervised self-training framework yielded a substantive increase in accuracy on the held-out unlabeled subset as shown in Figure 3.

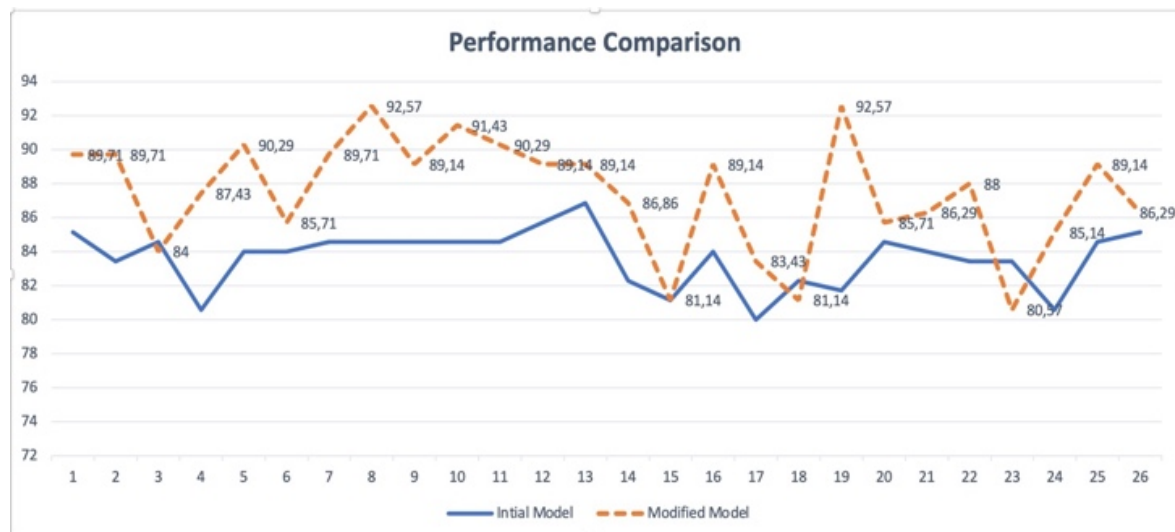


Figure 3. Performance comparison between initial model (solid line) and modified model (dashed line), when using *TfidfVectorizer*

Whereas the supervised models alone achieved 84.57 percent accuracy when applied to unseen data, incorporation of high-confidence pseudo-labels (selected via a dynamically adjusted threshold from 0.99 down to 0.80) raised accuracy to approximately 92.57 percent - an improvement of eight percentage points. This gain was consistent across multiple classifier architectures (e.g., *RidgeClassifierCV*, *LinearSVC*, *PassiveAggressiveClassifier*), indicating that leveraging unlabeled examples underpins more robust decision boundaries and mitigates overfitting to the labeled set.

Several factors contributed to this enhancement. First, pseudo-labeling fortified the training set with representative examples that better reflected the true distribution of the unlabeled corpus, thereby correcting class-distribution mismatches. Second, by weighting newly added instances according to model confidence scores, the algorithm prioritized reliable pseudo-labels, which in turn shaped more accurate parameter updates during retraining. Collectively, these mechanisms demonstrate that semi-supervised learning can substantially bridge the gap between limited labeled resources and diverse real-world data, ensuring that short-text classifiers remain effective under dataset shift.

6. Conclusion

This study has shown that meticulous preprocessing - comprising HTML-tag removal, minimum-length filtering, and stop-word elimination - combined with TF-IDF vectorization using unigrams and bigrams provides a strong foundation for short-text classification. Under these conditions, linear classifiers and passive aggressive classifiers achieved near-perfect accuracy on the labeled dataset. However, their performance deteriorated significantly when applied to previously unseen data, dropping to 84.57 percent, which highlights the challenges of overfitting and the mismatch between labeled and unlabeled distributions.

The introduction of a hybrid strategy, in which well-performing supervised classifiers are enhanced with semi-supervised learning, proved effective in addressing this limitation. Specifically, the integration of high-confidence pseudo-labels through a dynamic thresholding mechanism and the use of confidence-weighted self-training improved accuracy on the unlabeled subset by approximately eight percentage points, reaching 92.57 percent. This improvement demonstrates the viability of combining ordinary classification with semi-supervised augmentation as a practical methodology for contexts where labeled data are scarce and texts are extremely short.

The findings underscore two broader implications. First, preprocessing and feature engineering remain critical even in modern pipelines and should not be overlooked when dealing with short texts. Second, semi-supervised learning provides a powerful and cost-efficient means of improving classifier robustness in real-world applications, such as adaptive learning systems, where reliable categorization under data sparsity is essential.

Future research may proceed in two directions. From a methodological perspective, further attention should be given to addressing class imbalance and to exploring alternatives to standard k-means, such as fuzzy k-means, which can better capture the nuances of classes that are difficult to separate. And on the application side, the classified tasks will be integrated into a personalized recommendation system that adapts content to user needs—for instance, by suggesting additional tasks when a learner struggles with a particular type or by providing targeted practice in areas of weakness. This will extend the current contribution beyond classification accuracy to tangible educational benefits, offering an adaptive framework for supporting learners through intelligent task personalization.

References:

- [1]. Chen, J., Gong, Z., & Liu, W. (2020). A Dirichlet process biterm-based mixture model for short text stream clustering. *Applied Intelligence*, 50(5), 1609-1619.
- [2]. Li, Q., et al. (2022). A survey on text classification: From traditional to deep learning. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 13(2), 1-41.
- [3]. Zhang, Z., Wu, Z., & Shi, Z. (2022). An improved algorithm of TFIDF combined with Naive Bayes. *Proceedings of the 2022 7th International Conference on Multimedia and Image Processing*, 167-171.
- [4]. Guo, W. (2022). Applications of logistic regression and naive bayes in commodity sentiment analysis. *Proceedings of the 2022 4th International Conference on Image, Video and Signal Processing*, 224-230.
- [5]. Lu, Y., et al. (2022). Scalable multiple kernel k-means clustering. *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, 4279-4283.
- [6]. Raymaekers, J., & Zamar, R. H. (2022). Regularized k-means through hard-thresholding. *Journal of Machine Learning Research*, 23(93), 1-48.
- [7]. Wei, D., & Zhang, Z. (2023). K-means clustering algorithm based on improved density peak. *Proceedings of the 2023 3rd International Conference on Bioinformatics and Intelligent Computing*, 105-109.
- [8]. Ouali, Y., Hudelot, C., & Tami, M. (2020). An overview of deep semi-supervised learning. *arXiv preprint arXiv:2006.05278*.
- [9]. Mishra, S. K., Tripathi, A. M., & Chauhan, M. (2019). The role of semi-supervised learning in harnessing unlabeled data for model training. *The Pharma Innovation Journal*, 8(4S), 01-04.
- [10]. Kim, J., et al. (2023). An Efficient Noisy Label Learning Method with Semi-supervised Learning: An Efficient Noisy Label Learning Method with Semi-supervised Learning. *Proceedings of the 2023 6th International Conference on Machine Vision and Applications*, 161-166.
- [11]. Mey, A., & Loog, M. (2022). Improved generalization in semi-supervised learning: A survey of theoretical results. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(4), 4747-4767.
- [12]. Yang, X., et al. (2022). A survey on deep semi-supervised learning. *IEEE transactions on knowledge and data engineering*, 35(9), 8934-8954.
- [13]. Mvula, P. K., et al. (2024). A Survey on the Applications of Semi-supervised Learning to Cybersecurity. *ACM Computing Surveys*, 56(10), 1-41.
- [14]. Duarte, J. M., & Berton, L. (2023). A review of semi-supervised learning for text classification. *Artificial intelligence review*, 56(9), 9401-9469.
- [15]. Liang, Z., et al. (2024). Boosting Semi-Supervised Learning with Dual-Threshold Screening and Similarity Learning. *ACM Transactions on Multimedia Computing, Communications and Applications*, 20(9), 1-19.
- [16]. Zhu, Q., Xu, Z., & Hu, L. (2024). Semi-supervised image classification based on deep clustering and pre-trained models. *Proceedings of the 2024 7th International Conference on Computer Information Science and Artificial Intelligence*, 281-285.
- [17]. Wu, J., Pang, J., & Huang, Q. (2024). Feature-based Perturbation Makes a Better Ensemble Learning for SSL Classification. *Proceedings of the 2024 2nd Asia Conference on Computer Vision, Image Processing and Pattern Recognition*, 1-5.
- [18]. Zhuang, Y., et al. (2023). Self-supervised pre-training and semi-supervised learning for extractive dialog summarization. In *Companion Proceedings of the ACM Web Conference 2023*, 1069-1076.
- [19]. Yu, T., & Nwet, K. T. (2020). Comparing SVM and KNN algorithms for Myanmar news sentiment analysis system. *Proceedings of 2020 6th International Conference on Computing and Data Engineering*, 65-69.
- [20]. Jabari, I. K., et al. (2023). Learning-augmented K-means clustering using dimensional reduction. *Proceedings of the 8th International Conference on Sustainable Information Engineering and Technology*, 99-106.
- [21]. Jeffares, A. (2019). *K-means: A complete introduction*. Medium. Retrieved from: <https://medium.com/data-science/k-means-a-complete-introduction-1702af9cd8c> [accessed: 05 June 2025].
- [22]. Liu, W., Wang, N., & Huang, Y. (2021). Outlier detection method based on improved K-means clustering algorithm. *Proceedings of the 2021 5th International Conference on Electronic Information Technology and Computer Engineering*, 1350-1355.
- [23]. Makarychev, K., & Shan, L. (2022). Explainable k-means: don't be greedy, plant bigger trees!. *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, 1629-1642.