A Comparative Analysis of Deep Learning and Machine Learning Approaches for Spam Identification on Telegram

Shuo Xu , Zhanyi Ding , Zijing Wei , Chao Yang , Yixiang Li , Xuanjie Chen , Hailiang Wang *

Posted Date: 28 October 2025

doi: 10.20944/preprints202510.2167.v1

Keywords: natural language processing; spam detection; telegram; deep learning; machine learning; ALBERT; text classification; information security

Article

A Comparative Analysis of Deep Learning and Machine Learning Approaches for Spam Identification on Telegram

Shuo Xu [1], Zhanyi Ding [2], Zijing Wei [3], Chao Yang [4], Yixiang Li [5], Xuanjie Chen [6] and Hailiang Wang [7,*]

[1] Computer Science and Engineering Department, University of California San Diego, La Jolla, USA
[2] Center for Data Science, New York University, NY, USA
[3] College of Liberal Arts & Sciences, University of Illinois at Urbana-Champaign, Urbana, IL, USA
[4] Duke University, Durham, NC
[5] Department of Computer Science, The George Washington University, Washington, DC, USA
[6] Department of Applied Mathematics, University of Washington, Seattle, WA, USA
[7] School of Computer Science, College of Computing, Georgia Institute of Technology, Atlanta, GA, USA
* Correspondence: nealgatech@gmail.com

Abstract

Spam on messaging apps like Telegram is a serious threat to user security and experience. In this paper, we compared several machine learning (ML) and deep learning (DL) models to find the most effective way to detect it. We tested our models on a dataset of 20,348 messages. We put classic approaches like Logistic Regression and Tree-based Models including bagging and boosting against modern neural networks—a GRU and the ALBERT transformer. The results demonstrate that both GRU and ALBERT were the clear winners. The ALBERT model was the top performer, achieving state-of-the-art results with a weighted F1-score of 0.97 and an AUC of 0.9943. The GRU model also delivered excellent performance, with an F1-score of 0.94. Their real strength was in identifying the tricky minority 'spam' class. Here, ALBERT reached an F1-score of 0.95, and the GRU model scored 0.90, significantly outperforming the other methods. We used McNemar's test to confirm these findings were statistically significant. Ultimately, our study sets a new benchmark for spam detection. It proves that transformer models can effectively secure messaging platforms using only the content of the message itself.

Keywords: natural language processing; spam detection; telegram; deep learning; machine learning; ALBERT; text classification; information security

1. Introduction

The proliferation of instant messaging platforms has transformed digital communication but has also introduced significant security vulnerabilities. Malicious actors increasingly exploit these platforms to disseminate spam, which serves as a primary vector for phishing, malware distribution, and fraudulent schemes [1–4]. The sheer volume and real-time nature of messaging on applications like Telegram render traditional, manual moderation insufficient [4,5], creating an urgent need for automated, intelligent, and scalable spam detection systems to protect users [1,4].

Historically, spam filtering has relied on rule-based keyword filters and static blocklists. However, these methods are brittle and struggle to keep pace with the evolving tactics of spammers, who constantly adapt their language and techniques to evade detection [1,4]. In contrast, machine learning (ML) has become as a powerful paradigm for creating more adaptive and robust filters [1,4]. Supervised models such as tree-based models—boosting (e.g., gradient boosting machine) and bagging (e.g., Random Forest)—as well as Logistic Regression and can learn to identify spam from

 doi:10.20944/preprints202510.2167.v1

labelled data using engineered features like TF-IDF, offering both strong performance and a degree of model interpretability. Nevertheless, their reliance on explicit keywords can limit their ability to understand the contextual and semantic nuances of more sophisticated spam messages.

Recent breakthroughs in deep learning, especially Natural Language Processing (NLP) models, have introduced even more powerful tools for this task [1,4]. Neural networks like Gated Recurrent Units (GRUs) are skilled at picking up sequential patterns in text, while transformer-based models like A Lite Bidirectional Encoder Representations from Transformers (ALBERT) have set a new standard in understanding deep contextual meaning. Despite their promise, a systematic, head-to-head comparison of classical ML versus modern deep learning architectures on real-world data from a platform like Telegram is needed to establish clear performance benchmarks, a methodological approach that has proven valuable in related security domains like fake news detection [5,6].

In this study, we wanted to find the most effective way to automatically detect spam in Telegram messages. We systematically compared traditional machine learning models with modern deep learning approaches to see which worked best:

- We thoroughly benchmarked a range of classical machine models. We tested classic methods like Logistic Regression and tree-based models including bagging and boosting methods against modern NLP approaches like GRU and ALBERT. All models were evaluated on the 'Telegram Spam or Ham' dataset [7] using weighted F1-score and AUC to ensure a fair comparison.
- We also validated our results statistically. To do this, we used McNemar's tests to confirm that the performance differences between the models were significant and not just due to chance.
- Finally, we looked into how the classical models made their decisions. We analyzed the key features and words they used to identify spam, which gave us insight into the linguistic patterns of spam on Telegram.

Our approach deliberately focuses on the practical challenge of classifying short, often informal, text messages. Unlike email spam detection, which can leverage metadata and long-form content, filtering on instant messaging platforms must rely on highly condensed and limited textual information. This constraint provides a rigorous test of a model's ability to extract meaningful signals from sparse input and has several practical advantages: (i) it necessitates models that are efficient enough for real-time inference; (ii) it reflects a realistic deployment scenario where only the message content is available; and (iii) it establishes which architectures are most capable of discerning intent from minimal context.

This is a roadmap for the rest of the paper. First, in Section II, we will describe our dataset, how we prepared the data, and the models we built. Section III presents our experimental results and statistical comparisons. Finally, Section IV concludes the paper with a discussion of our findings and provides direction for future research.

2. Methods

In this section, we lay out the complete methodological framework designed for our comparative study. We detail: (1) the dataset and the distinct preprocessing pipelines developed for classical and deep learning approaches; (2) the implementation and hyperparameter tuning of the selected machine learning models, including Logistic Regression and tree-based models—Random Forest (i.e., bagging) and LightGBM (i.e., boosting), and deep learning architectures, such as a GRU (Gated Recurrent Unit) and ALBERT (Lite Bidirectional Encoder Representations from Transformers); (3) the comprehensive evaluation framework, including the performance metrics; and (4) statistical tests used to compare the models rigorously.

2.1. Dataset and Preprocessing

For this current study, we employed a publicly available dataset from Kaggle entitled '*Telegram Spam or Ham*' [7] designed for binary text classification. After an initial data quality check where null entries were removed, the final corpus consisted of 20,348 messages. Each record contains the

 doi:10.20944/preprints202510.2167.v1

message text and a corresponding text_type label, which was binarized to '0' for legitimate messages ('ham') and '1' for spam. The dataset presents a notable class imbalance, containing 14,337 legitimate messages and 6,011 spam messages, a factor that guided our choice of evaluation metrics and modeling strategies. To ensure a robust and fair comparison, distinct preprocessing pipelines were tailored for the classical supervised learning and deep learning models.

To ensure a fair and rigorous comparison, and to isolate model performance from feature engineering choices, we harmonized the initial preprocessing pipeline for both the classical supervised learning and deep learning models. A minimal text cleaning pipeline was applied to all data prior to model-specific vectorization or tokenization. All text was converted to lowercase, and noise elements such as URLs, user mentions, and hashtags were systematically replaced with special tokens, such as HASHTAG, URL, and USER, using regular expressions. In addition, stopwords and punctuation were intentionally retained for all models, as they can provide important contextual cues. This unified approach eliminates potential confounding variables and allows for a direct comparison of the architectural strengths of classical versus deep learning models.

After cleaning the text, we used a method named Term Frequency–Inverse Document Frequency (TF-IDF) to convert the words into numerical vectors for our classical models. To prevent any "data leakage," we trained this TF-IDF tool exclusively on the training data before using it to transform the validation and test sets. For all experiments, we split the dataset into three parts: 60% for training, 20% for validation, and 20% for testing, using a fixed random seed to ensure our results could be reproduced.

2.2. Classical Supervised Learning Models

To assess the effectiveness of classical supervised learning on this task, we first set a baseline using a linear classifier. From there, we evaluated whether more complex, non-linear tree-based ensembles—using both bagging and boosting methodologies—could provide better results.

2.2.1. Logistic Regression

Logistic Regression [8] was selected as the approach model given its high interpretability, computational efficiency, and robust performance in text classification tasks. This model estimates the probability of a message being spam by applying a sigmoid function of a linear combination of its input TF-IDF features. To prevent overfitting and address the dataset's class imbalance, we systematically tuned key hyperparameters, including the optimization solver, the type of penalty, the regularization strength (C), and the class_weight parameter, with the final model decided by the highest F1-score (weighted) on the validation Telegram set.

2.2.2. Tree-Based Models

Building upon this baseline, we explored tree-based models. A single decision tree classifier can model complex, non-linear relationships by recursively spliting the input data based on feature values. Nevertheless, individual trees are highly prone to overfitting, which is a common problem when working with text data, as it creates a feature space with a vast number of dimensions and very few non-zero values. To overcome this limitation, this study focuses on advanced ensemble methods that aggregate the outputs of multiple trees to produce more robust and generalizable predictions.

2.2.2.1. Random Forest

For the bagging approach, we implemented Random Forest, an ensemble method that builds a multitude of decision trees on bootstrapped samples of the training data [9]. By also considering only a random subset of features at each split, Random Forest effectively reduces variance and captures complex feature interactions. A comprehensive grid search was conducted to optimize several key hyperparameters, including n_estimators (i.e., the number of trees), max_depth (i.e., the maximum

 doi:10.20944/preprints202510.2167.v1

tree depth), the minimum samples required for splits and leaves, and the set-up of class weight, again using the F1-score (weighted) as the selection criterion.

2.2.2.2. LightGBM

For the boosting approach, we utilized LightGBM, a high-performance gradient boosting framework [10,11]. Boosting models work differently than Random Forest by building trees in a sequence. Each new tree in the series focuses on correcting the errors made by the one before it. LightGBM is particularly well-suited for this task because of its efficiency and scalability, which stem from its use of a leaf-wise tree growth strategy and histogram-based feature binning. We tuned a series of its core hyperparameters, including the max_depth, number of estimators, learning_rate, and class_weight, to maximize the weighted F1-score on the validation data. The performance of these two advanced tree-based ensembles was then compared against the logistic regression baseline to quantify the benefits of non-linear modeling for this task.

2.3. Deep Learning Models

In addition to the classical supervised models, we investigated two powerful deep learning architectures capable of learning feature representations directly from text, thereby moving beyond the constraints of fixed feature engineering like TF-IDF. We selected a Gated Recurrent Unit (GRU) to represent recurrent neural networks and the ALBERT model to represent the state-of-the-art transformer architecture.

2.3.1. GRU (Gated Recurrent Unit)

A GRU is one type of RNN (recurrent neural network) particularly designed to address sequential data like text by understanding the dependencies and context between words [12]. Unlike traditional "bag-of-words" models, a GRU processes text in order. It uses internal gating mechanisms to decide what information to remember or forget, which allows it to learn nuanced contextual relationships. For our implementation, we converted input messages into padded integer sequences using a vocabulary built only from the training data. The model architecture included a single GRU layer, an embedding layer, and a fully connected output layer that used dropout for regularization. We conducted a randomized search to tune key hyperparameters, including the learning rate, the embedding dimension, hidden unit size, and number of epochs. To account for the class imbalance in the dataset, we trained the model using a weighted cross-entropy loss function. The final model was decided with the one that achieved the highest weighted F1-score on the validation set.

2.3.2. ALBERT (A Lite Bidirectional Encoder Representations from Transformers)

We also used ALBERT, a modern and state-of-the-art transformer model known for its high performance and parameter efficiency [13]. Unlike RNNs that process text word-by-word, transformers use a self-attention approach, which enables the model to weigh the importance of all words in a message at the same time, helping it to capture complex, long-range relationships and deeper contextual meanings. For this study, we fine-tuned the pre-trained albert-base-v2 model specifically for our spam classification task. First, we tokenized the input messages using the official ALBERT tokenizer. Then, we truncated or padded the sequences to a uniform length to meet the model's input requirements. We ran a randomized hyperparameter search to find the optimal learning rate, dropout rate, and number of fine-tuning epochs. Just like with the GRU model, we used a weighted cross-entropy loss function to handle the label imbalance in the data. The final configuration was chosen based on which one achieved the highest weighted F1-score on the validation set.

2.4. Evaluation and Comparison Framework

 doi:10.20944/preprints202510.2167.v1

To ensure we could compare all the models in a comprehensive and reliable way, we created a multi-faceted evaluation framework. We assessed performance using standard metrics like accuracy, precision, recall, and the weighted F1-score. Because our dataset had a significant class imbalance, we made the weighted F1-score our primary metric for tuning and selecting the best models. This score provides a balanced measure of how well a model performs on both the majority ("ham") and the minority ("spam") classes.

In addition to these, we calculated the Area Under the Receiver Operating Characteristic Curve (AUC-ROC). This helped us evaluate how well each model could distinguish between the two classes across all possible decision thresholds. To ensure our results were stable, we also computed 95% confidence intervals for both the F1-score and AUC using a bootstrapping method with 1,000 iterations.

Finally, we needed to determine if the performance differences we observed were statistically meaningful. To do this, we performed pairwise McNemar's tests with a Holm-Bonferroni correction on the test set predictions. This rigorous statistical test gave us a solid basis for concluding whether one model was significantly better than another for this specific task [14,15].

3. Results

In this section, we present the results of our experiments, breaking down how each model performed on the spam detection task. We will cover three main points: First, we show the overall performance of all models—both classical and deep learning—on the held-out test set. We evaluated this using standard metrics, including the weighted F1-score and AUC. Second, we share the results from the pairwise statistical significance tests, which we used to validate that the observed differences in performace between the models were meaningful. Finally, we provide an analysis of the most influential features identified by the classical models. This provides insight into their decision-making processes.

3.1. Model Performance Metrics

We conducted a thorough assessment of both classical supervised learning and deep learning models on the Telegram spam detection task. A summary of each model's performance on the held-out test set is presented in Table 1, which details the weighted-average recall, precision, F1-score, and Area Under the Curve (AUC). The results show a clear performance advantage for the transformer-based ALBERT architecture.

Table 1. Summary of Model Performance on the Held-Out Test Set.

Models	AUC	Macro Average		
		Recall	Precision	F1-Score
Logistic Regression	0.9567	0.90	0.90	0.9012
Random Forest	0.9499	0.88	0.88	0.8805
LightGBM	0.9581	0.91	0.91	0.9082
ALBERT	0.9943	0.97	0.97	0.9695
GRU	0.9816	0.94	0.94	0.9405

Among the classical supervised learning models, the boosting ensemble LightGBM performed as the top performer, achieving a weighted F1-score of 0.9082. It was followed closely by the Logistic Regression baseline, which obtained an F1-score of 0.9012. The bagging-based Random Forest model also performed competently with an F1-score of 0.8805. All three classical models demonstrated strong discriminative power, achieving high AUC values of approximately 0.96, indicating a reliable capability to distinguish spam from ham messages.

 doi:10.20944/preprints202510.2167.v1

The deep learning models, however, yielded superior results. The ALBERT model achieved the highest performance across all metrics, with a weighted-average precision of 0.97, recall of 0.97, and an F1-score of 0.9695, along with a near-perfect AUC of 0.9943. This indicates a robust and highly balanced detection capability with a minimal trade-off between identifying spam correctly and avoiding false positives. The GRU model also delivered excellent performance, achieving a notably higher F1-score of 0.9405 and an exceptionally high AUC of 0.9816. This firmly places its performance above the classical models and establishes a clear hierarchy where both deep learning architectures outperform the traditional methods. These findings underscore the advantage of advanced neural architectures, particularly transformers, in capturing the complex linguistic patterns indicative of spam.

3.2. Analysis of Per-Class Performance

A closer examination of the per-class performance, detailed in Table 2, reveals a consistent trend: while most models performed well on the majority 'ham' class, the true difference in capability emerged in the classification of the minority 'spam' class. For instance, the top-performing classical model, LightGBM, achieved an F1-score of 0.94 for 'ham' but only 0.84 for 'spam'. In contrast, the deep learning models demonstrated a superior ability to identify the challenging spam messages. The GRU model showed a marked improvement, surpassing the classical models with a spam F1-score of 0.90, and the ALBERT model achieved an exceptional F1-score of 0.95 on the spam class, significantly outperforming all other approaches in this critical regard.

Table 2. Per-Class Performance Metric on the Held-out Test Set.

Model	Class	Recall	Precision	F1-Score
Logistic Regression	Ham (0)	0.95	0.91	0.93
	Spam (1)	0.79	0.87	0.83
Random Forest	Ham (0)	0.90	0.92	0.91
	Spam (1)	0.83	0.78	0.81
LightGBM	Ham (0)	0.96	0.91	0.94
	Spam (1)	0.79	0.90	0.84
ALBERT	Ham (0)	0.97	0.99	0.98
	Spam (1)	0.98	0.93	0.95
GRU	Ham (0)	0.94	0.97	0.96
	Spam (1)	0.93	0.88	0.90

This discrepancy between classical and deep learning models on the spam class is likely due to two factors. First is the class imbalance in the training data (14,337 'ham' vs. 6,011 'spam' examples). Second, the 'spam' class may possess greater linguistic diversity and adversarial variations, making it an inherently more challenging target. The superior performance of the deep learning models suggests they are more capable of learning generalizable patterns from the imbalanced and complex minority class [16–20].

3.3. Feature Importance and Model Interpretation

To interpret how the classical models were making their decisions, the most importance text features (TF-IDF) as identified by each model's built-in influential metrics. This interpretive analysis revealed a strong consensus across the different architectures regarding the key linguistic markers of spam, while also highlighting subtle differences in their approaches.

A clear theme emerged as all three models consistently identified features related to financial transactions, urgent calls to action, and superlative offers as strong predictors of spam. Across the ensemble models in particular, terms such as 'free', 'earn', 'money', 'prize', and 'guaranteed' were highly ranked. Action-oriented words like 'click', 'join', 'link', and 'visit' were also flagged as highly

 doi:10.20944/preprints202510.2167.v1

predictive. This significant overlap underscores that all the classical models successfully learned to associate classic spam keywords with the positive class.

While the models converged on these core spam indicators, the Logistic Regression model provided unique insights. By analyzing its coefficients, we observed that it also assigned high importance (via large negative coefficients) to words characteristic of legitimate, non-spam correspondence, such as 'wrote', 'date', 'research', and 'discus'. This indicates its decision-making process was based on both the presence of spam markers and the presence of words typical in normal conversation, a distinction less explicitly captured by the feature importance metrics of the ensemble models.

4. Discussion and Conclusion

4.1. Interpretation of Findings

Our study systematically benchmarked a range of classical and deep learning models for the application of spam identification using only the text of Telegram messages. Our results conclusively show that while traditional models like LightGBM and Logistic Regression provide a strong performance baseline, advanced neural architectures—specifically the transformer-based ALBERT model—achieve a superior level of accuracy and robustness. This advantage was particularly evident on the more challenging minority 'spam' class. As shown in Table 2, ALBERT achieved an F1-score of 0.95 for spam detection, a substantial improvement over the 0.84 F1-score from the best-performing classical model, LightGBM. Furthermore, the GRU model also clearly outperformed the classical methods, achieving a spam F1-score of 0.90. This demonstrates that deep learning models, and especially transformers, are better equipped to learn the diverse and subtle linguistic patterns of spam from imbalanced data.

Within the classical models, the boosting-based LightGBM consistently outperformed the others, benefiting from its ability to effectively model the sparse TF-IDF feature space. Our feature importance analysis revealed that the ensemble models heavily prioritized overt spam keywords related to finance and calls to action. A core finding of this study is that state-of-the-art spam identification is achievable with a minimal feature set. We demonstrate that high accuracy is attainable without relying on richer metadata, which addresses a key deployment challenge on platforms where user privacy is a priority.

4.2. Practical Implications

Our results lead to some important takeaways for anyone creating practical spam filtering applications:

- Model Determination: For systems where computational resources/budgets are limited or model interpretability is a priority, LightGBM offers an effective and high-performing solution. For applications requiring higher accuracy without the full computational overhead of a large transformer, a GRU-based model offers an excellent balance of performance and efficiency. However, for mission-critical applications that demand the highest possible detection accuracy and resilience against evolving threats, transformer-based models such as ALBERT are the recommended solution.
- Threat Intelligence: The consistent importance of features related to finance ('money', 'earn', 'profit'), offers ('free', 'prize', 'guaranteed'), and direct engagement ('click', 'join', 'link') confirms that these classic spam themes remain prevalent on modern platforms. This insight can inform simpler, rule-based pre-filtering layers in a tiered security architecture.
- Deployment Efficiency: The success of a text-only approach lowers the barrier for deploying effective spam filters, as it removes the need for complex data integration and respects user privacy by not requiring metadata.

4.3. Study Limitations and Future Directions

 doi:10.20944/preprints202510.2167.v1

The limitations of this project also highlight several promising directions for future research:

- Dataset and Language: Our results are specific to the single English-language dataset used. Further studies should validate these findings on larger, multilingual datasets to test the generalizability of the models. Such an extension would introduce additional technical challenges, including the need for language-specific or multilingual transformer models (e.g., mBERT, XLM-R) capable of handling diverse tokenization requirements. Moreover, these models must be robust against code-switching—the practice of mixing languages within a single message—and informal transliterations, which are common on global messaging platforms. Finally, the scarcity of large, publicly available, and accurately labeled multilingual spam datasets would need to be addressed through further data collection or annotation efforts.
- Metadata Exclusion: By design, this study omitted metadata such as sender reputation or URL analysis. Future work could quantify the performance gains from integrating these features into a hybrid model.
- Interpretability of Deep Models: While ALBERT's performance is superior, its lack of transparency presents a significant challenge when trying to interpret its decision-making process. Future research could apply model-agnostic interpretation techniques like SHAP or LIME to increase the transparency of transformer-based filters.
- Dynamic Threats: The models were trained on a static dataset. Spammers continuously evolve their tactics. Future work should explore online or continual learning frameworks that allow models to adapt to new spam campaigns in real-time.
- Emerging Models: The rapid advancements in Large Language Models (LLMs) present a promising avenue for developing even more sophisticated, zero-shot, or few-shot spam classifiers that require less labelled training data.

In summary, our work provides a rigorous benchmark for spam detection on a modern messaging platform, demonstrating that text-only, transformer-based pipelines is able to arrive the highest performance. This study provides a practical guide to create highly accurate and scalable security solutions for real-time communication environments where efficiency and effectiveness are paramount.

References

1. Karim A., Azam. S, Shanmugam B., Kannoorpatti K. and Alazab, M. "*A Comprehensive Survey for Intelligent Spam Email Detection*," in *IEEE Access*, vol. 7, pp. 168261-168295, 2019, doi: 10.1109/ACCESS.2019.2954791.
2. Pattanaik C. B., Das S., Arsh A. and Kar N., "*A Survey on Phishing Attacks and Their Counter-Measures*," in Intelligent Systems and Sustainable Computing (ICISSC 2022), Reddy V. S., Prasad V. K., Wang J. and Rao Dasari N. M. (eds), Smart Innovation, Systems and Technologies, vol. 363, Springer, Singapore, 2023, https://doi.org/10.1007/978-981-99-4717-1_45.
3. Xu S., Cao Y., Wang Z., and Tian Y., "*Fraud detection in online transactions: Toward hybrid supervised–unsupervised learning pipelines*," in *Proc. 2025 6th Int. Conf. Electron. Commun. Artif. Intell.* (ICECAI 2025), 2025
4. Liu Y., Shen X., Zhang Y., Wang Z., Tian Y., Dai J., and Cao Y., "*A systematic review of machine learning approaches for detecting deceptive activities on social media: methods, challenges, and biases*," *International Journal of Data Science and Analytics*, vol. 20, pp. 6157–6182, 2025.
5. Gillespie J. T., *Custodians of the Internet: Platforms, Content Moderation, and the Hidden Decisions that Shape Social Media*, Yale University Press, 2018.
6. Tian Y., Xu S., Cao Y., Wang Z., and Wei Z., "*An empirical comparison of machine learning and deep learning models for automated fake news detection*," *Mathematics*, vol. 13, no. 20, p. 2086, 2025.
7. Mexwell, M. O. "*Telegram Spam or Ham*," Kaggle, 2022. [Online].
8. Hosmer D. W. and Lemeshow S., *Applied Logistic Regression*, 2nd ed., New York, NY: John Wiley & Sons, Inc., 2000.
9. Breiman L., "*Random forests*," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

 doi:10.20944/preprints202510.2167.v1

10. Friedman J. H., "*Greedy function approximation: A gradient boosting machine*," *Annals of Statistics*, vol. 29, no. 5, pp. 1189–1232, 2001.
11. Ke G., Meng Q., Finley T., Wang T., Chen W., Ma W., Ye Q., and Liu T.-Y., "*LightGBM: A highly efficient gradient boosting decision tree*," in *Proc. 31st Int. Conf. Neural Information Processing Systems* (NeurIPS), pp. 3149–3157, 2017.
12. Cho K. et al., "*Learning Phrase Representations Using RNN Encoder–Decoder for Statistical Machine Translation*," in Proc. *2014 Conf. on Empirical Methods in Natural Language Processing* (EMNLP), pp. 1724–1734, 2014.
13. Lan Z., Chen M., Goodman S., Gimpel K., Sharma P., & Soricut R. "*ALBERT: A Lite BERT for self-supervised learning of language representations*." In *Proceedings of the 8th International Conference on Learning Representations (ICLR)*, 2020
14. Powers D. M. W., "*Evaluation: From precision, recall and F-measure to ROC, informedness, markedness and correlation*," *Journal of Machine Learning Technologies*, vol. 2, no. 1, pp. 37–63, 2011.
15. Davis J. and Goadrich M., "*The relationship between Precision-Recall and ROC curves*," in *Proc. 23rd Int. Conf. Machine Learning (ICML)*, pp. 233–240, 2006.
16. Cao Y., Dai J., Wang Z., Zhang Y., Shen X., Liu Y., and Tian Y., "*Machine learning approaches for depression detection on social media: A systematic review of biases and methodological challenges*," *Journal of Behavioral Data Science*, vol. 5, no. 1, Feb. 2025.
17. Liu Y., Shen X., Zhang Y., Wang Z., Tian Y., Dai J., and Cao Y., "*A systematic review of machine learning approaches for detecting deceptive activities on social media: methods, challenges, and biases*," *International Journal of Data Science and Analytics*, vol. 20, pp. 6157–6182, 2025.
18. Xu S., Tian Y., Cao Y., Wang Z., and Wei Z., "*Benchmarking machine learning and deep learning models for fake news detection using news headlines*," Preprints, article 2025061183, 2025.
19. Ding Z., Wang Z., Zhang Y., Cao Y., Liu Y., Shen X., Tian Y., and Dai J., "*Trade-offs between machine learning and deep learning for mental illness detection on social media*," *Scientific Reports*, vol. 15, article no. 14497, 2025.
20. Zhang Y., Wang Z., Ding Z., Tian Y., Dai J., Shen X., Liu Y., and Cao Y., "*Employing machine learning and deep learning models for mental illness detection*," *Computation*, vol. 13, no. 8, p. 186, 2025.