A Hybrid Deep Learning System for Illegal Text Detection and High-Quality Comment Filtering

*

1st Junya YANG
Computer Science and Technology
BNU-HKBU UIC
Zhuhai, China
s230026188@mail.uic.edu.cn

2nd Shunxuan DING
Computer Science and Technology
BNU-HKBU UIC
Zhuhai, China
s230026232@mail.uic.edu.cn

2nd Jiaying SHAN
Computer Science and Technology
BNU-HKBU UIC
Zhuhai, China
s230026230@mail.uic.edu.cn

2nd Yue GUAN
Computer Science and Technology
BNU-HKBU UIC
Zhuhai, China
s230033013@mail.uic.edu.cn

2nd Haocheng XIE
Computer Science and Technology
BNU-HKBU UIC
Zhuhai, China
s230026178@mail.uic.edu.cn

***Abstract*—Text classification is the bedrock of natural language processing (NLP) activities, like content moderation, spam detection, and sentiment analysis. This report focuses on an innovative, well-designed system for identifying illegal texts and filtering high-quality comments, which combines traditional machine learning (ML) methods and novel deep learning (DL) techniques to attain the best performance. The illegal text detection module incorporates sparse rule-based sensitive word filtering in addition to FastText's rapid filtering and BERT for final classification, ending with a balance between processing power and accuracy. Subjects used as experimental subjects indicate that the deep learning methods, such as BERT and FastText, excel in comparison with traditional ML models, which are Multinomial Naive Bayes, Random Forest, and XGBoost, which give almost perfect accuracy and F1-scores. The operating principles of the high-quality comment filtering part are the utilization of a pretrained BERT model to get token embeddings and the trained Auto-Encoder for the reconstruction-based quality assessment. Evaluation is done at a standard of reconstruction errors to differentiate high-quality comments, and this is through a threshold-based method that gives reliable filtering. The experimental analysis focuses on the model's two-stage hybrid procedure for illegal text detection and concludes that the Auto-Encoder captures the content features for which it was trained properly.**

***Index Terms*—Text Classification, Natural Language Processing (NLP), Illegal Text Detection, High-Quality Comment Filtering, Reconstruction-based Quality Assessment**

I. Introduction

As digital platforms are expanding rapidly, the supervision of quality and legality of user content on them is one of the problems that needs immediate attention. Text classification, as one of the core features of natural language processing (NLP), is one of the most important solutions to this challenge. The use of text classification in applications like spam detection, sentiment analysis, and content moderation has a considerable reliance on its precision and speed. On the contrary, textual data's heterogenic nature, context variation, and semantic complexity are the major issues faced by conventional machine learning (ML) techniques.

Since traditional ML techniques require significant feature engineering to process text, some of the methods used are Multinomial Naive Bayes (MNB), Support Vector Machines (SVMs), and ensemble methods like Random Forest and XGBoost. Unlike these models, which perform their best with smaller datasets and comparatively simple tasks, they cannot cope with the deep semantic and contextual dependency that complex text data has.

On the contrary, DL approaches, in particular, such as FastText, Recurrent Neural Networks (RNNs), Long Short-Term Memory (LSTM), the Transformer architecture, and BERT, which have introduced a leap over the original text classification methods. These models avoid the manual feature construction process and leverage word embeddings and attention mechanisms to adaptively learn context-sensitive models of text. Among these, BERT has shown the best performance by incorporating context preservation, alongside RNNs.

In this report, we propose a comprehensive system designed for two primary tasks: **illegal text detection** and **high-quality comment filtering**. The system employs a **hybrid approach** that combines rule-based methods, lightweight DL models (FastText), and robust transformer-based models (BERT). Specifically:

- **Illegal Text Detection**: A two-stage process that uses **FastText** for fast inference and **BERT** for precise classification. This approach balances computational efficiency and accuracy while handling large-scale text data.
- **High-Quality Comment Filtering**: The system uses **BERT embeddings** and an **Auto-Encoder** to assess

reconstruction errors and distinguish high-quality comments from low-quality ones.

The experimental evaluation compares traditional ML methods with DL techniques. Results show that DL models, particularly **FastText** and **BERT**, significantly outperform ML models in accuracy, precision, recall, and F1-score. The proposed Auto-Encoder further ensures effective filtering of high-quality comments based on reconstruction error thresholds.

By integrating advanced deep learning methods with a hybrid filtering pipeline, the proposed system demonstrates its potential for scalable, efficient, and accurate text moderation in real-world applications.

II. Related Works

Text classification has been a pivotal task in natural language processing (NLP) and has seen substantial progress over the years. The methodologies for text classification can be categorized into two primary paradigms: traditional Machine Learning (ML) approaches and Deep Learning (DL) techniques. Each of these paradigms has evolved to address various challenges posed by textual data, such as high dimensionality, context dependency, and variability in text length and semantics.

A. Machine Learning Methods

Traditional ML approaches usually rely on strongly featured engineering and statistical strategies. In this context, Multinomial Naive Bayes (MNB) is one of the simplest and most commonly used techniques. MNB is based on Bayes' theorem, assuming conditional independence of features given the class label. While this assumption often fails for real-world datasets, MNB's efficiency and effectiveness on small datasets make it suitable for tasks like spam filtering and basic sentiment analysis. However, MNB struggles with interdependence among features, limiting its performance in complex scenarios.

Support Vector Machines (SVMs) are another widely used ML technique for text classification. SVMs find an optimal hyperplane that separates classes with maximum margin, offering robust performance for high-dimensional data. Kernel functions enable SVMs to handle non-linear decision boundaries, making them effective for tasks like document categorization and sentiment analysis. However, SVMs are computationally expensive, requiring extensive parameter tuning, which limits their scalability for large datasets.

Ensemble methods, such as Random Forest, XGBoost, and LightGBM, have also been extensively applied in text classification. Random Forest leverages an ensemble of decision trees to improve generalization and robustness. Boosting methods like XGBoost and LightGBM sequentially train weak classifiers to minimize errors, achieving strong performance in structured text data applications like biomedical text classification. However, these methods depend heavily on feature engineering and struggle with context-sensitive and sequential data.

B. Deep Learning Methods

Deep learning (DL) techniques have transformed text classification by learning feature representations directly from raw data. FastText, developed by Facebook, introduced a simple yet effective approach by representing text as dense vectors (word embeddings) and averaging them to create document-level embeddings. Its subword modeling capability makes it robust to spelling variations and morphological differences. However, FastText's simplicity may limit its ability to model complex relationships between words.

Recurrent Neural Networks (RNNs) and their variants, such as Long Short-Term Memory (LSTM) networks, bring sequential modeling capabilities to text classification. LSTMs address the vanishing gradient problem in traditional RNNs and capture long-term dependencies, making them suitable for context-heavy tasks like sentiment analysis and question answering. However, their sequential nature leads to high computational costs for large-scale datasets.

Convolutional Neural Networks (CNNs), initially developed for image processing, have been adapted for text classification by treating sentences as one-dimensional sequences. CNNs extract n-gram features through convolutional filters, making them effective for tasks requiring local context, such as short text classification. However, they struggle to model long-term dependencies in text.

Transformer-based models, particularly BERT (Bidirectional Encoder Representations from Transformers), have revolutionized text classification. BERT leverages self-attention mechanisms to capture bidirectional context and long-range dependencies. Pretrained on massive datasets, BERT achieves state-of-the-art performance across various NLP tasks, including text classification, sentiment analysis, and named entity recognition. Despite its high accuracy, BERT's computational complexity poses challenges for real-time applications.

C. Comparison of Methods

While ML methods like MNB and SVM provide computational simplicity and efficiency, they depend heavily on feature engineering and often fail to capture semantics and context. DL approaches, particularly transformer-based models like BERT, overcome these limitations but require substantial computational resources and data.

The combination of classical ML techniques and modern DL methods has significantly expanded the scope of text classification research. By tailoring the choice of methodology to task-specific requirements, researchers can achieve a balance between efficiency, accuracy, and scalability. These advancements provide the foundation for designing hybrid approaches that integrate rule-based methods, lightweight DL models, and robust transformer-based techniques, as proposed in this report.

III. Methodology

The proposed system consists of two main components: **Illegal Text Detection** and **High-Quality Comment Filtering**. These components are designed to operate sequentially for

TABLE I
TEXT CLASSIFICATION METHODS - ADVANTAGES

Method	Advantages
Multinomial Naive Bayes (MNB)	Simple, fast, effective for small datasets
Support Vector Machines (SVMs)	Works well with high-dimensional data
Random Forest	Handles noisy and imbalanced data
FastText	Fast and scalable, robust to misspellings
LSTM	Captures sequential dependencies, suitable for long texts
TextCNN	Efficient, captures n-gram features effectively
BERT and Variants	State-of-the-art accuracy, handles long texts with global context

TABLE II
TEXT CLASSIFICATION METHODS - LIMITATIONS

Method	Limitations
Multinomial Naive Bayes (MNB)	Independence assumption may not hold
Support Vector Machines (SVMs)	Computationally intensive for large datasets
Random Forest	Limited interpretability, not optimal for sparse data
FastText	Limited for complex relationships between words
LSTM	Slow training, struggles with very long sequences
TextCNN	Fails to model long-term dependencies
BERT and Variants	High computational cost, requires large-scale pretraining

TABLE III
TEXT CLASSIFICATION METHODS - BEST USE CASES

Method	Best Use Cases
Multinomial Naive Bayes (MNB)	Spam filtering, basic sentiment analysis
Support Vector Machines (SVMs)	Review categorization, topic modeling
Random Forest	Biomedical text classification, domain-specific tasks
FastText	Low-resource or domain-specific tasks
LSTM	Sentiment analysis, question answering
TextCNN	News classification, short text classification
BERT and Variants	High-stakes tasks like medical or legal text classification

robust content moderation. Each component is trained and evaluated independently, ensuring flexibility and scalability. Below, we describe the methodology for each stage in detail.

A. System Workflow

The overall workflow of the system is depicted in Figure 1. The process is structured as follows:

1) Input text undergoes sensitive word matching and FastText screening.
2) Texts flagged by both methods are classified as **illegal**.
3) Conflicting cases are resolved by the BERT classification model.
4) Non-illegal texts proceed to the high-quality comment filtering stage.
5) Token embeddings are generated using pretrained BERT.
6) An Auto-Encoder calculates reconstruction error to assess quality.
7) Texts with acceptable reconstruction errors are deemed **high-quality comments**.

B. Illegal Text Detection

The first stage is illegal text detection, which acts as a filtering point. The goal is to efficiently identify and classify texts containing sensitive or illegal content. This hybrid stage combines rule-based systems and machine learning for improved precision.

1) Sensitive Word Matching: The text is first analyzed using a rule-based sensitive term-matching approach. A predefined set of illegal or sensitive vocabulary is used to flag texts containing these terms. This approach is computationally efficient and interpretable. However, it may generate false positives or fail to detect texts with different wording but similar expressions.

Let the sensitive vocabulary be defined as $\mathcal{V} = \{v_1, v_2, \ldots, v_n\}$. For a given input text T, the detection condition is:

$$\text{Flag}(T) = \begin{cases} 1 & \text{if } \exists\ v \in \mathcal{V},\ v \in T, \\ 0 & \text{otherwise.} \end{cases}$$

2) FastText Model Screening: Simultaneously, the input text is processed by the FastText model. FastText represents text using word embeddings and efficiently predicts the probability of a text being illegal. Its robustness to spelling variations and slight textual modifications makes it a reliable initial filter.

The FastText model calculates the probability as:

$$P_{\text{illegal}}(T) = \sigma(\mathbf{W} \cdot \mathbf{h}_T + b),$$

where $\mathbf{h}_T$ is the embedding of text T, $\mathbf{W}$ is the weight matrix, b is the bias, and σ is the sigmoid activation function. The text

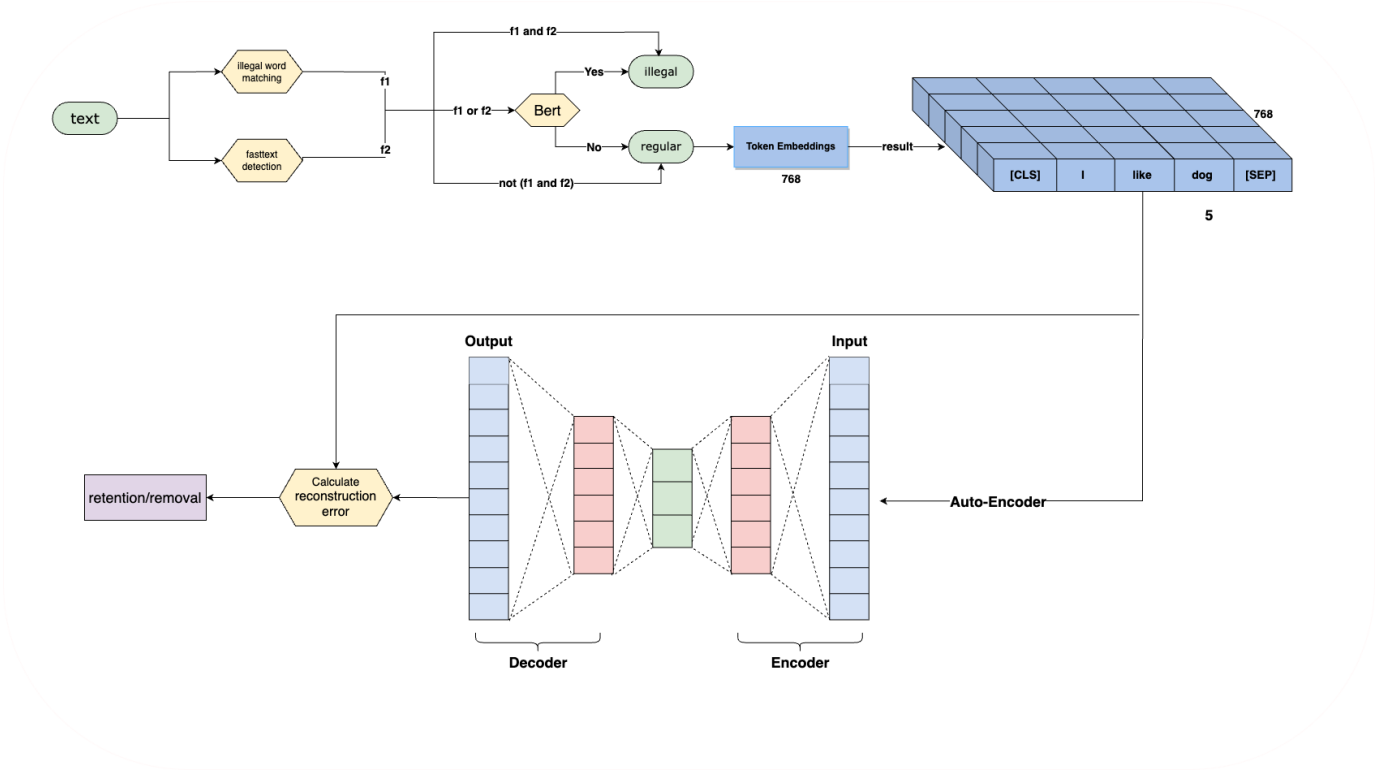

Fig. 1. System Workflow

is classified based on a threshold τ_{fasttext}:

$$\text{Flag}_{\text{FastText}}(T) = \begin{cases} 1 & \text{if } P_{\text{illegal}}(T) \geq \tau_{\text{fasttext}}, \\ 0 & \text{otherwise.} \end{cases}$$

3) Fusion of Results: The outputs from the sensitive word matching and FastText methods are fused for enhanced reliability:

- If both methods flag the text as illegal, it is classified as **illegal**.
- If both methods agree that the text is non-illegal, it proceeds to the next stage.
- If the results conflict, the BERT model is invoked for final classification.

4) BERT Model for Conflict Resolution: The BERT (Bidirectional Encoder Representations from Transformers) model is used to resolve conflicts. By leveraging its deep contextual understanding, BERT generates bidirectional representations of the input text and fine-tunes them for classification.

The contextual embedding $\mathbf{h}_T^{\text{BERT}}$ is computed, and a softmax layer predicts the class probabilities:

$$P_{\text{illegal}}, P_{\text{non-illegal}} = \text{Softmax}(\mathbf{W} \cdot \mathbf{h}_T^{\text{BERT}} + b).$$

The text is classified based on the maximum probability.

C. High-Quality Comment Filtering

For texts classified as **non-illegal**, the system evaluates their quality using the high-quality comment filtering process. This stage employs pretrained BERT for token embedding generation and an Auto-Encoder for reconstruction-based quality assessment.

1) Token Embeddings Generation with BERT: The input text is tokenized and converted into embeddings using a pretrained BERT model. The output is a multi-dimensional matrix where each row corresponds to the embedding of a token in the input sequence. This representation captures both semantic and syntactic information.

Mathematically, for an input text T with tokens $\{t_1, t_2, \ldots, t_n\}$, the embeddings are represented as:

$$\mathbf{E}_T = [\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_n],$$

where $\mathbf{E}_T \in \mathbb{R}^{n \times d}$, n is the number of tokens, and d is the embedding dimension.

2) Auto-Encoder Reconstruction: The BERT-generated embeddings are processed by a trained Auto-Encoder. The Auto-Encoder architecture consists of:

- **Encoder:** Compresses high-dimensional embeddings into a latent representation:

$$\mathbf{z} = f_{\text{encoder}}(\mathbf{E}_T).$$

- **Decoder:** Reconstructs the original embeddings from the latent representation:

$$\hat{\mathbf{E}}_T = f_{\text{decoder}}(\mathbf{z}).$$

The reconstruction error is calculated as the mean squared error (MSE):

$$\text{Reconstruction Error} = \frac{1}{n \times d} \sum_{i=1}^{n} \sum_{j=1}^{d} \left(\mathbf{E}_T[i,j] - \hat{\mathbf{E}}_T[i,j] \right)^2.$$

3) Reconstruction Error Threshold: A predefined threshold is applied to the reconstruction error:

- If the error is within the acceptable range, the text is classified as a **high-quality comment**.
- If the error exceeds the threshold, the text is discarded as low-quality.

D. Advantages of the Hybrid Approach

The proposed hybrid approach combines rule-based, lightweight ML, and advanced DL methods to achieve:

- **Efficiency:** FastText handles the bulk of text filtering, reducing computational load for BERT.
- **Accuracy:** BERT ensures precise classification for ambiguous cases.
- **Scalability:** The system is designed to handle large-scale text data with minimal latency.
- **Robustness:** The Auto-Encoder effectively differentiates high- and low-quality comments.

IV. Experimental Study and Result Analysis

This section presents the experimental evaluation of the proposed system. It focuses on two main components: **Illegal Text Detection** and **High-Quality Comment Filtering**. The results demonstrate the performance of both traditional Machine Learning (ML) methods and Deep Learning (DL) approaches, emphasizing the benefits of the proposed hybrid model.

A. Illegal Text Detection: Experimental Results and Model Selection

Extensive experiments were conducted to evaluate the performance of various ML and DL models for illegal text detection. The evaluation metrics include **Accuracy**, **Precision**, **Recall**, and **F1-Score**. The results are summarized in Tables IV and V.

TABLE IV
Performance of Machine Learning Models for Illegal Text Detection

Model	Accuracy	Precision	Recall	F1-Score
Multinomial Naive Bayes	0.5461	0.5427	0.5461	0.5441
Random Forest	0.5899	0.5905	0.5899	0.4942
XGBoost	0.5847	0.5672	0.5847	0.5165
LightGBM	0.5784	0.5548	0.5784	0.5086
Support Vector Machines	0.5680	0.5491	0.5680	0.5423
Logistic Regression	0.5600	0.5413	0.5600	0.5375

1) Key Observations and Model Selection:

- **Deep Learning models significantly outperform ML models.** BERT, FastText, and LSTM achieve near-perfect accuracy and F1-scores of 0.99, while ML methods perform below 0.60 in accuracy and F1-scores.

TABLE V
Performance of Deep Learning Models for Illegal Text Detection

Model	Accuracy	Precision	Recall	F1-Score
FastText	0.99	0.99	0.985	0.985
BERT	0.99	0.985	0.99	0.99
LSTM	0.99	0.99	0.985	0.99
TextCNN	0.98	0.985	0.985	0.985

- **Hybrid Approach:** FastText is used for initial filtering due to its computational efficiency, while BERT resolves ambiguous cases with high precision.

The proposed two-stage hybrid model leverages the strengths of both FastText and BERT, achieving a balance between efficiency and accuracy.

B. High-Quality Comment Filtering: Reconstruction Error Analysis

The Auto-Encoder architecture was evaluated based on its ability to differentiate high- and low-quality comments through reconstruction error thresholds.

1) Training and Validation Performance: Figure of autoencoder loss shows the training and validation loss curves for the Auto-Encoder over 200 epochs. Key observations include:

- The loss decreases monotonically, converging to a minimum of approximately 0.1, indicating effective learning.
- Minimal discrepancy between training and validation loss demonstrates strong generalization performance.

2) Reconstruction Error Distribution: Figure 3 illustrates the distribution of reconstruction errors:

- A threshold (red dashed line) at the 95th percentile separates high-quality comments from low-quality ones.
- Most comments have reconstruction errors below 0.50, indicating high-quality content.
- A small proportion of comments with errors above the threshold are classified as low-quality.

3) Performance Metrics:

- **Precision:** The Auto-Encoder achieves high precision in identifying high-quality comments, minimizing false positives.
- **Scalability:** The reconstruction-based approach is computationally efficient, suitable for large-scale deployments.

The experimental results validate the effectiveness of the proposed system:

- The hybrid FastText-BERT approach achieves near-perfect accuracy for illegal text detection while maintaining computational efficiency.
- The Auto-Encoder successfully filters high-quality comments based on reconstruction errors, demonstrating strong generalization and scalability.

The system's performance highlights its potential for real-world applications requiring robust text moderation.

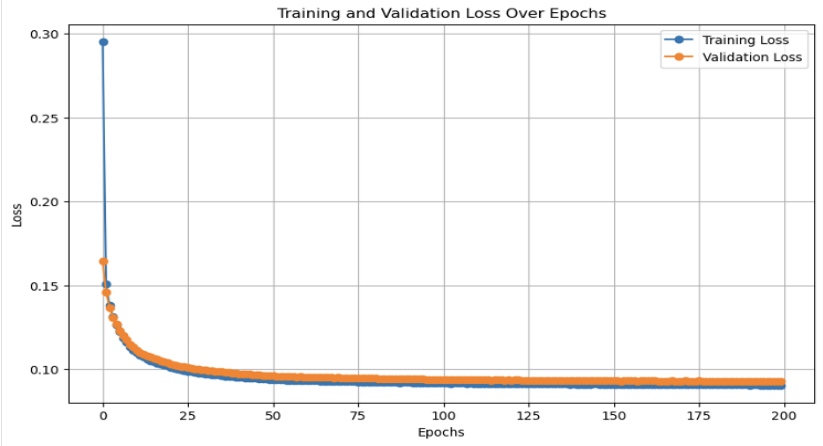

Fig. 2. Auto-Encoder Loss Curve

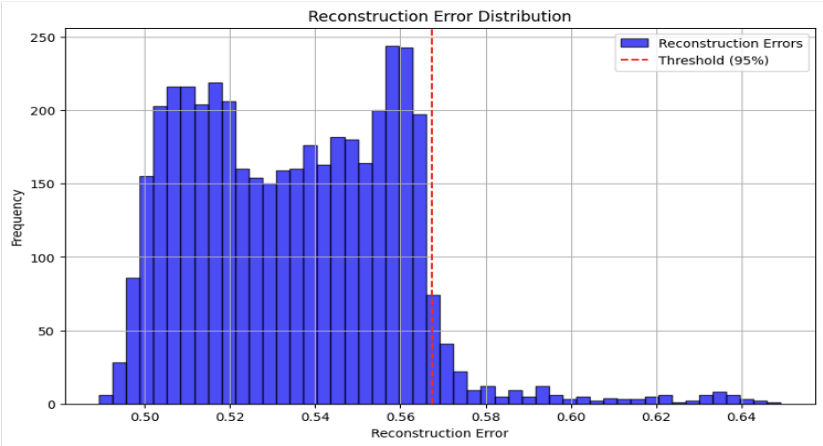

Fig. 3. Reconstruction Error Distribution

V. Downstream Application: Box Office Analysis

A. Overview

Building on the filtered dataset, we aim to predict box office performance for movies, leveraging a graph-based learning framework. By integrating review-derived keywords, textual embeddings, and causal inference, we construct a graph representation that encapsulates both semantic richness and causal structure. This graph is processed using a Graph Attention Network (GAT) enhanced with causal regularization, providing robust predictions even for movies with limited annotated data.

B. From Keywords to Graph Nodes

Keywords, extracted from the filtered Maoyan reviews using the TextRank algorithm, play a pivotal role in enriching the graph's semantic structure. These keywords capture themes and contextual nuances from user reviews, supplementing the semantic information embedded in movie titles. For example, a keyword such as "heroic sacrifice" derived from a review highlights thematic elements that might not be apparent from the movie title alone.

Each keyword node is initialized with its BERT-derived word embedding $\mathbf{e}_k$, allowing it to share a feature space with movie titles. Keywords are connected to movies via edges weighted by their respective TextRank scores, denoted as w_{km}, ensuring that their semantic relevance influences the graph.

C. Graph Construction and Feature Design

The graph $G = (V, E)$ consists of heterogeneous nodes and weighted edges, with features designed to capture the unique characteristics of each node type:

a) Node Features.: Each node i is represented by a feature vector $\mathbf{x}_i$ based on its type:

- **Movie nodes:** Feature vector $\mathbf{x}_m = [\mathbf{t}, r, y, b]$, where:
  - $\mathbf{t}$: BERT-derived embedding of the movie title.
  - r: Movie rating, capturing audience feedback.
  - y: Release year, providing temporal context.
  - b: Box office revenue (for annotated movies).
- **Actor, genre, and keyword nodes:** Feature vector includes:

$$\mathbf{x}_a = [\bar{b}], \quad \mathbf{x}_g = [\bar{b}], \quad \mathbf{x}_k = [\mathbf{e}_k, \bar{b}],$$

where:

$$\bar{b} = \frac{1}{|M|} \sum_{m \in M} b_m,$$

represents the average box office revenue of movies associated with the node, and $\mathbf{e}_k$ is the keyword's BERT embedding.

b) Edge Weights.: Edges (i, j) between nodes are initialized with weights w_{ij} based on their relationships:

- For keyword-to-movie edges, w_{km} is the TextRank score between keyword k and movie m.
- For other edges, w_{ij} is initialized using causal effects estimated via a dual machine learning framework:

$$w_{ij} = \mathbb{E}[Y \mid \text{do}(X = i)] - \mathbb{E}[Y \mid \text{do}(X = j)],$$

where $\text{do}(X)$ represents the intervention on node X, estimated through separate treatment and control models.

D. Training the Graph Attention Network

The graph is processed using a Graph Attention Network (GAT) to dynamically aggregate information. The GAT attention mechanism computes attention scores α_{ij} for each edge, emphasizing important relationships:

$$\alpha_{ij} = \frac{\exp(\text{LeakyReLU}(\mathbf{a}^\top [\mathbf{W}\mathbf{h}_i || \mathbf{W}\mathbf{h}_j]))}{\sum_{k \in \mathcal{N}(i)} \exp(\text{LeakyReLU}(\mathbf{a}^\top [\mathbf{W}\mathbf{h}_i || \mathbf{W}\mathbf{h}_k]))},$$

where:

- $\mathbf{h}_i$ is the feature vector of node i.
- $\mathbf{W}$ is a learnable weight matrix.
- $\mathcal{N}(i)$ denotes the neighbors of node i.

The GAT's loss function incorporates a causal regularization term to align learned attention scores α_{ij} with pre-computed causal weights w_{ij}:

$$\mathcal{L} = \mathcal{L}_{\text{GAT}} + \lambda \mathcal{L}_{\text{causal}},$$

where:

$$\mathcal{L}_{\text{causal}} = \sum_{(i,j) \in E} (\alpha_{ij} - w_{ij})^2,$$

and λ controls the strength of the regularization.

E. Training Details

To ensure robust performance and prevent overfitting, we adopt the following training strategies:

- **Dropout:** A dropout rate of 0.2 is applied to node features and attention scores during training.
- **Early Stopping:** Training is halted if the validation loss does not improve for 100 consecutive epochs.
- **Optimization:** The Adam optimizer is used with a learning rate of 10^{-3}, and model performance is evaluated on a validation set to fine-tune hyperparameters.

0.48

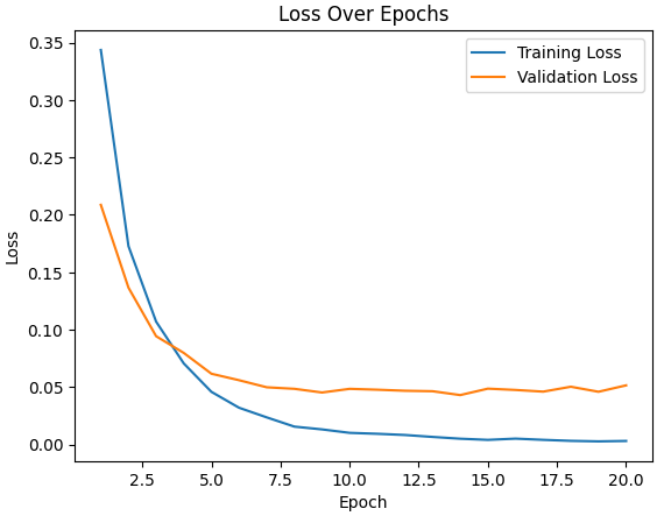

Fig. 4. Training and validation loss over epochs.

0.48

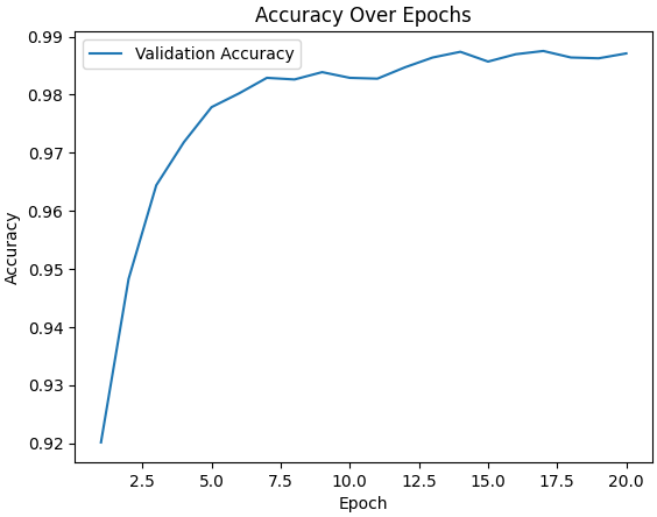

Fig. 5. Validation accuracy over epochs.

Fig. 6. Performance metrics during training: (a) Loss and (b) Accuracy.

F. Insights and Applications

The final GAT model predicts box office revenues for movies, providing insights for stakeholders:

- **Market Analysis:** Identify trends in actors, genres, or keywords contributing to high revenues.
- **Content Optimization:** Guide decisions on casting, themes, and marketing strategies.
- **Recommendation Systems:** Improve movie recommendations by incorporating predicted box office performance.

This method bridges structured and unstructured data, combining semantic richness (via BERT embeddings and TextRank keywords) with causal insights to deliver a novel, robust framework for box office analysis.

VI. Future Work and Conclusion

A. Future Work

While the proposed system demonstrates strong performance in illegal text detection and high-quality comment filtering, there are several avenues for improvement and exploration in future research:

1) Validation of Effectiveness: Since the current system uses an **unsupervised model** for high-quality comment filtering, it is challenging to directly validate its effectiveness without labeled data.

- **Proposed Solution:** Deploy the system in a real-world application environment and collect user feedback to assess accuracy and reliability. Metrics such as user satisfaction, content retention rates, and manual moderation corrections can provide insights.
- **Semi-Supervised Fine-Tuning:** Incorporate semi-supervised learning techniques using a small labeled dataset to improve performance.

2) Improving the Auto-Encoder Architecture: The current Auto-Encoder architecture uses fully connected layers, which may not capture complex relationships in text data. Proposed enhancements include:

- **Convolutional Layers:** Introduce convolutional layers to capture local dependencies and patterns in text embeddings.
- **Transformer-Based Auto-Encoders:** Leverage Transformer architectures like BERT or GPT to improve the Auto-Encoder's ability to capture long-range dependencies and semantic relationships.
- **Variational Auto-Encoders (VAEs):** Use VAEs to introduce a probabilistic framework, improving robustness to noise and ambiguity in text.
- **Hybrid Models with Attention Mechanisms:** Incorporate attention mechanisms to prioritize important tokens during the reconstruction process.

3) Data Augmentation and Expansion: Augment the dataset to enhance model robustness and generalization:

- Apply techniques like back-translation, paraphrasing, and noise injection to create diverse training samples.
- Include multilingual data to make the system adaptable for global applications.

4) Real-Time Deployment and Scalability: Evaluate the system's performance in real-time scenarios:

- Optimize latency and throughput for large-scale deployments.
- Integrate with live content moderation systems and measure operational performance.

5) Human-in-the-Loop Validation: Incorporate human moderators to fine-tune the model:

- Use human feedback to refine thresholds and improve the model's interpretability.
- Implement a feedback loop to adapt to evolving user behavior and content trends.

B. Conclusion

This report presents a robust and scalable system for illegal text detection and high-quality comment filtering. The key contributions include:

- A hybrid approach combining **FastText** for efficient initial filtering and **BERT** for precise classification of illegal text.
- A reconstruction-based quality assessment using **Auto-Encoder**, which effectively filters high-quality comments.
- Experimental evaluation demonstrating that the proposed system significantly outperforms traditional ML approaches in terms of accuracy, precision, and recall.

The hybrid model balances computational efficiency and classification accuracy, making it suitable for large-scale applications. The Auto-Encoder architecture effectively distinguishes high-quality content, providing an additional layer of robust filtering.

Future Outlook: By addressing current limitations, such as incorporating advanced Auto-Encoder architectures and semi-supervised learning techniques, the system can be further enhanced. Real-world validation and user feedback will ensure its adaptability and effectiveness in evolving content moderation scenarios.

In conclusion, the proposed system serves as a foundation for scalable, efficient, and accurate text moderation in real-world applications. Its flexibility and modular design pave the way for future advancements in content analysis and classification.

References

[1] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 135–146, 2017. Available: https://arxiv.org/abs/1607.04606

[2] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2019. Available: https://arxiv.org/abs/1810.04805

[3] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006. DOI: https://doi.org/10.1126/science.1127647

[4] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997. DOI: https://doi.org/10.1162/neco.1997.9.8.1735

[5] Y. Kim, "Convolutional neural networks for sentence classification," in *Proc. of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014. Available: https://arxiv.org/abs/1408.5882

[6] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001. DOI: https://doi.org/10.1023/A:1010933404324

[7] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2016. Available: https://arxiv.org/abs/1603.02754

[8] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," in *Proc. of the 2nd International Conference on Learning Representations (ICLR)*, 2014. Available: https://arxiv.org/abs/1312.6114

[9] J. Wei and K. Zou, "EDA: Easy data augmentation techniques for boosting performance on text classification tasks," in *Proc. of the 2019 Conference on Empirical Methods in Natural Language Processing*, 2019. Available: https://arxiv.org/abs/1901.11196