

AI-Driven Ensemble for Enhanced Sentiment Polarity Detection in Movie Reviews

Apeksha Bhuekar
apeksharaj17@gmail.com
Independent Researcher

Abstract

This paper presents a weighted multi-model ensemble approach for discerning sentiment polarity in text documents, specifically consumer reviews. We address the binary classification problem of identifying positive versus negative sentiment by proposing a hybrid framework that integrates generative, discriminative, and deep embedding-based models. Our key contribution is a weighted voting mechanism that leverages cross-validation to assign model-specific weights, effectively harnessing the complementary strengths of its diverse constituents. This ensemble strategy is evaluated on widely recognized movie review datasets, where it demonstrates robust performance compared to standalone models. Our method achieves 93.1% accuracy on the IMDB dataset and 90.6% accuracy on the Rotten Tomatoes dataset. Our ensemble achieves improvements over individual models, with absolute accuracy gains of 1.6 percentage points on IMDB (93.1% vs. 91.5% for BERT) and 3.3 to 12.2 percentage points on Rotten Tomatoes. These results demonstrate the effectiveness of the proposed weighting strategy for sentiment classification.

Keywords

• Sentiment Analysis • Ensemble Learning • Sentiment Classification • BERT (Bidirectional Encoder Representations from Transformers) • Weighted Voting

1. Introduction and Motivation

With the rapid growth of user-generated content on platforms such as IMDB, Amazon, and Rotten Tomatoes, sentiment analysis has become important for understanding public opinion, consumer preferences, and brand perception. Comprehensive studies have established its applications across domains [1], while movie reviews provide a rich testbed for analyzing nuanced opinions [2]. Its relevance also extends to broader online discourse, including crime-related discussions in online communities [3].

Sentiment polarity detection classifies text as positive or negative and has been extensively studied in NLP. Traditional approaches such as Naïve Bayes, SVMs, and logistic regression use lexical features including unigrams, bigrams, and TF-IDF [4], but often struggle with linguistic complexity, sarcasm, and domain-specific expressions. Similar feature-engineering challenges occur in other machine learning applications involving hybrid classification methods [5].

Deep learning models, including RNNs, LSTMs, and CNNs, improved sentiment classification by capturing contextual and syntactic information [6]. Pretrained models such as BERT and GPT further improved NLP through bidirectional context and transfer learning [7]. Recent studies on related NLP tasks such as Named Entity Recognition further demonstrate the applicability of deep learning [8], while its scalability has also been demonstrated in large-scale image recognition [9].

Despite these advances, no single model is universally optimal across sentiment datasets. LDA can capture latent topics but may miss fine-grained sentiment cues, SVMs provide effective decision boundaries but depend on engineered features, and BERT provides strong semantic representations but can face domain drift or oversmoothing [10]. Similar challenges arise in long-context reasoning [11]. Fine-tuning [12] and domain adaptation have improved transformer performance, paralleling optimization strategies in transformer-based vision models [13]. Ensemble and active learning methods have also shown benefits for efficient text classification [14], while cross-domain generalization remains important in applications such as LSTM-based pediatric glucose forecasting [15].

Several gaps remain in existing transformer and ensemble-based sentiment analysis approaches. First, many ensembles rely on simple averaging or majority voting rather than learning model-specific weights from validation performance. Second, probability calibration across heterogeneous models, including topic models, kernel methods, recurrent networks, and transformers, remains limited. Third, the complementary strengths of LDA, SVM, LSTM, and BERT have not been systematically examined within a unified weighted ensemble. To address these gaps, the proposed approach learns ensemble weights from out-of-fold predictions so that each model's contribution reflects its empirical reliability on held-out data.

This paper proposes a hybrid ensemble framework integrating generative, discriminative, and deep embedding-based models for sentiment polarity detection. The hypothesis is that a carefully designed weighted ensemble can exploit the complementary strengths of its constituent models while mitigating their individual limitations.

The proposed pipeline incorporates three model families: (i) generative models represented by Latent Dirichlet Allocation (LDA), (ii) discriminative models represented by Support Vector Machines, and (iii) deep embedding-based models including BERT and LSTM. Each model independently generates predictions, which are subsequently fused through a weighted voting mechanism optimized using cross-validation.

The ensemble is evaluated on the IMDB and Rotten Tomatoes benchmark datasets and demonstrates improved accuracy, precision, and recall compared with standalone models. The approach provides a practical and scalable framework for multi-model NLP sentiment classification, with scalability considerations informed by self-service data-wrangling techniques for data-driven decision making [16].

The primary contribution is the integration of three distinct model paradigms within a single optimized fusion strategy. Specifically, the framework combines the thematic abstraction of LDA, the efficient linear separation of SVM, and the contextual representations of BERT/LSTM through a validated weighting scheme, providing a more robust and generalizable sentiment analysis framework. The methodological rigor is further informed by formal modeling techniques used in software product line design [17].

2. Model Architecture and Methodology

Our proposed sentiment analysis system is built upon a modular ensemble architecture that integrates multiple model families generative, discriminative, and deep embedding-based under a unified inference pipeline. This section details the overall workflow, data preprocessing, and training methodology used in the system.

2.1 System Overview

Figure 1 illustrates the architecture of the proposed sentiment detection framework, which comprises four major components: data preprocessing, feature extraction, base model training, and ensemble fusion.

In the data preprocessing stage, input text is cleaned, normalized, and tokenized to reduce noise and ensure consistent formatting. The feature extraction stage generates model-specific representations, including count-based TF-IDF features, embedding-based representations such as BERT vectors and GloVe embeddings, and topic-based distributions obtained using LDA. These processed features are then used to train the individual base models independently on the training dataset according to their respective configurations. Finally, the ensemble fusion component aggregates the predictions of the trained models using a weighted voting mechanism to determine the final sentiment polarity label.

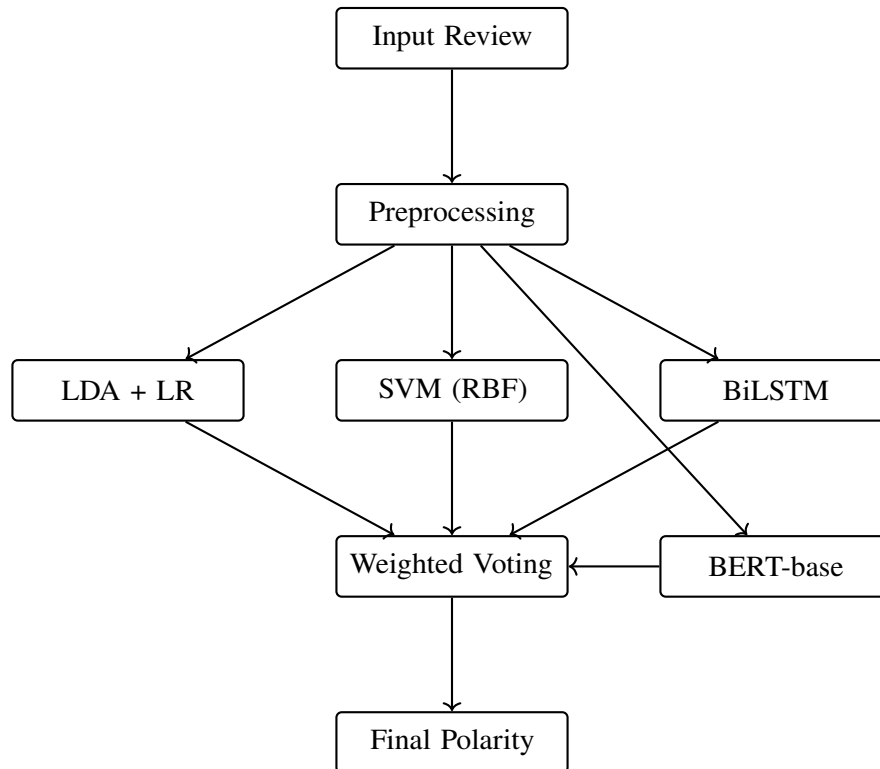


Figure 1: Architecture of the proposed ensemble showing four independent components: LDA+LR, SVM (RBF kernel), BiLSTM, and BERT-base. Each component produces a probability output, and weighted fusion combines these predictions into the final classification.

2.2 Preprocessing Pipeline

Different preprocessing pipelines are applied to the input text according to the requirements of each target model. For the LDA+LR model, the text is lowercased, punctuation and special characters are removed using regular expressions, stopwords are removed using the NLTK stopwords list, the text is tokenized using the NLTK word tokenizer, and the resulting tokens are lemmatized using the WordNet lemmatizer. The SVM classifier uses the same preprocessing pipeline, followed by TF-IDF vectorization using unigram and bigram features. For the BiLSTM model, the text is lowercased and tokenized using the Keras Tokenizer, after which sequences are padded or truncated to a fixed length of 500 tokens and mapped to the corresponding GloVe embedding indices. The BERT model uses the pretrained BERT tokenizer from the Hugging Face Transformers library, which performs WordPiece subword tokenization; no additional preprocessing, such as lowercasing, stopwords removal, or lemmatization, is applied. To prevent information leakage, all data-dependent preprocessing components are learned exclusively from the training set and subsequently applied without refitting to the validation and test sets. Specifically,

the NLTK tokenizer and WordNet lemmatizer operate using the training-data vocabulary, the TF-IDF vectorizer derives its vocabulary and inverse document frequency statistics solely from the training data, and the LDA topic model is fitted only on the training documents, with its learned topic representation used to infer topic distributions for the validation and test documents. Similarly, the Keras Tokenizer vocabulary is constructed exclusively from the training data, while the BERT tokenizer uses the pretrained vocabulary of the underlying BERT model, which was established independently of the experimental datasets.

2.3 Datasets

We evaluate our approach on two publicly available benchmark datasets for movie-review sentiment classification.

IMDB Dataset The IMDB dataset contains 50,000 movie reviews, evenly divided into 25,000 positive and 25,000 negative samples. The official split provides 25,000 training and 25,000 test reviews. From the training set, 5,000 samples (2,500 per class) are reserved for validation, leaving 20,000 samples for model training. Table 1 summarizes the dataset statistics.

Rotten Tomatoes Dataset The Rotten Tomatoes dataset contains 10,662 movie reviews with binary sentiment labels. It follows a standard split of 8,531 training, 1,066 validation, and 1,065 test samples, with an approximately balanced class distribution of 5,334 positive and 5,328 negative reviews.

Table 1: Dataset statistics.

Dataset	Total	Train	Validation	Test	Positive/Negative
IMDB	50,000	20,000	5,000	25,000	25,000 / 25,000
Rotten Tomatoes	10,662	8,531	1,066	1,065	5,334 / 5,328

2.4 Training Strategy

2.4.1 Nested Cross-Validation Procedure

We employ a nested cross-validation procedure to ensure robust hyperparameter selection and unbiased performance estimation. The outer loop uses 5-fold cross-validation on the training data, with each fold serving once as a validation set while the remaining four folds are used for model training. Within each outer training fold, an inner 3-fold cross-validation is performed to select the hyperparameters for each base model. Specifically, 80% of the original training data in each outer fold is used for training and 20% is held out for validation, with hyperparameters selected through the inner 3-fold procedure before retraining each base model on the full training portion of that fold. Probability predictions are then generated for the corresponding validation fold. After completing all five outer folds, out-of-fold probability predictions are obtained for every training sample and used to learn the ensemble weights, preventing bias from models that have previously seen the same samples during training. The final ensemble weights are optimized using these out-of-fold predictions and the objective function described in Section 4, after which the weighted ensemble is evaluated on the held-out test set.

2.4.2 Ensemble Weight Learning

Ensemble weights are learned exclusively from genuine out-of-fold predictions generated through the nested cross-validation procedure described above. Specifically, for each training sample, we obtain a probability prediction from each base model using a model that was trained on data that did not include that sample. This ensures that the predictions used for weight optimization are genuinely out-of-fold and not contaminated by the base models having seen the same samples during training. The weight optimization objective is:

$$\min_{\mathbf{w}} - \sum_{i=1}^N \log \left(\sum_{m=1}^M w_m \cdot p_m(y_i | \mathbf{x}_i) \right) \quad (1)$$

subject to $\sum_{m=1}^M w_m = 1$, $w_m \geq 0$, where $M = 4$ is the number of ensemble components, N is the number of training samples, $p_m(y_i | \mathbf{x}_i)$ is the probability predicted by model m for sample i , and $\mathbf{w}$ is the weight vector. This optimization is performed on the out-of-fold predictions, not on predictions from models that were trained on the same data being evaluated.

2.4.3 Reproducibility Settings

To ensure reproducibility, we set the following random seeds: Python random seed = 42, NumPy random seed = 42, TensorFlow random seed = 42, and PyTorch random seed = 42. All reported results represent a single run with these fixed seeds. For stochastic models (BiLSTM and BERT), we perform five independent runs with different random seeds (42, 123, 456, 789, 1011) to estimate variance, and report the mean and standard deviation of the test accuracy across these runs (see Section 5).

2.5 Model Families and Integration Points

The ensemble consists of four distinct base models: (1) a Latent Dirichlet Allocation (LDA) model with Logistic Regression (LR) applied to the topic distribution features, (2) a Support Vector Machine (SVM) classifier with TF-IDF features, (3) a Bidirectional LSTM (BiLSTM) network with pretrained GloVe embeddings, and (4) a fine-tuned BERT-base model. These four models constitute the complete ensemble, and their outputs are combined through learned weights optimized on validation data. Table 2 summarizes the architecture and feature representation of each component.

Table 2: Ensemble component architectures and feature representations.

Model	Architecture	Feature Representation
LDA+LR	LDA + Logistic Regression	Topic distribution (K=50)
SVM	SVM (RBF kernel)	TF-IDF (unigrams + bigrams)
BiLSTM	2-layer Bidirectional LSTM	GloVe embeddings (300d)
BERT	BERT-base-uncased	WordPiece subword tokens

In this configuration, Logistic Regression serves exclusively as the classifier applied to the LDA topic features, not as an independent ensemble member. The LDA+LR component produces probabilistic predictions based on the topic distribution of each review. The standalone Logistic Regression model (without LDA features) is not included in the ensemble, and therefore its standalone results are not reported. The SVM, BiLSTM, and BERT models are the other three independent ensemble members.

3. Component Models: Generative, Discriminative, and Embedding-Based

The effectiveness of our ensemble model stems from its diverse composition, which integrates complementary model families with distinct strengths. This section details the three model categories generative, discriminative, and embedding-based used in our sentiment polarity detection system, along with their configurations and theoretical justifications.

3.1 Generative Models

Generative models aim to capture the underlying distribution of the input data. In our pipeline, Latent Dirichlet Allocation (LDA) [18] is used to uncover hidden semantic topics in movie reviews, with each document represented as a probability distribution over topics that serves as a low-dimensional and interpretable feature vector. The LDA model is implemented using the Gensim library, with the number of topics set to $K = 50$ based on coherence score evaluation on the validation set for $K \in [20, 30, 40, 50, 60, 80]$ using the C_v metric. The vocabulary is constructed exclusively from the training data, removing words occurring in fewer than 5 documents or more than 90% of documents. The model is trained for 100 passes over the corpus with a random seed of 42, using symmetric hyperparameters $\alpha = 0.1$ and $\beta = 0.01$. During inference, each document is represented as a probability distribution over the 50 learned topics using the trained model's inference method. Although LDA is not designed explicitly for sentiment tasks, it captures high-level structure and co-occurrence patterns, which are especially useful when reviews discuss multiple sub-themes [19]. The topic distributions are fed into a logistic regression classifier to predict sentiment polarity, providing a generative-discriminative hybrid subcomponent. The integration of generative models with discriminative classifiers parallels approaches in other domains, such as implicit Bayesian learning for enhanced sales forecasting [20].

3.2 Discriminative Models

Discriminative models such as Support Vector Machines (SVM) [21] learn direct decision boundaries between sentiment classes and operate effectively on sparse representations such as TF-IDF or bag-of-words vectors [4]. In our pipeline, the SVM uses an RBF kernel with $\gamma = 0.1$, selected through grid search over $\gamma \in [0.001, 0.01, 0.1, 1.0]$, while the regularization parameter is set to $C = 1.0$, selected from $C \in [0.1, 0.5, 1.0, 5.0, 10.0]$. Class weighting is set to balanced to account for potential class imbalance. The input features are TF-IDF vectors containing unigram and bigram representations, with the vocabulary limited to the top 10,000 features derived exclusively from the training data. Probability outputs are obtained through 5-fold cross-validation on the training data to calibrate SVM decision scores into probabilities. The use of the RBF kernel enables nonlinear decision boundaries by mapping the input features into a higher-dimensional space where linear separation becomes possible.

3.3 Embedding-Based Deep Models

Recent advances in NLP have demonstrated the power of dense, context-aware word embeddings in sentiment analysis. We incorporate two deep learning models:

3.3.1 Bidirectional LSTM

The BiLSTM model is implemented using TensorFlow/Keras and consists of two bidirectional LSTM layers, with 128 hidden units per direction (256 total) in the first layer and 64 hidden units per direction

(128 total) in the second layer. The input sequences are represented using 300-dimensional pretrained GloVe embeddings (GloVe.6B.300d) [22], which are fine-tuned during training. A dropout rate of 0.5 is applied to the embedding layer and after each LSTM layer to reduce overfitting. All input sequences are padded or truncated to a maximum length of 500 tokens. The model is optimized using Adam with a learning rate of $\eta = 0.001$ and trained with a batch size of 64 for a maximum of 10 epochs. Early stopping with a patience of 2 is employed based on validation loss. A random seed of 42 is used to ensure reproducibility during weight initialization and data shuffling.

3.3.2 BERT Fine-Tuning

A pretrained bert-base-uncased model with 110 million parameters is fine-tuned on the training set using the Hugging Face Transformers library. The model uses a learning rate of 2×10^{-5} , selected through grid search over $[1e-5, 2e-5, 3e-5, 5e-5]$, and is optimized using AdamW with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 1e-8$. A linear learning-rate scheduler with warmup is applied, increasing the learning rate from 0 to the peak value during the first 10% of training steps before linearly decaying it to 0. The model is trained with a batch size of 32 for up to 4 epochs, using the default BERT dropout rate of 0.1, gradient clipping with a maximum norm of 1.0, and early stopping based on validation loss with a patience of 1 epoch. A random seed of 42 is used for initialization and shuffling, and the maximum sequence length is set to 512 tokens. Hyperparameter search is performed using the validation set, with learning rate $\in [1e-5, 2e-5, 3e-5, 5e-5]$, batch size $\in [16, 32]$, and epochs $\in [3, 4, 5]$. The achieved accuracy of 91.5% on IMDB is consistent with standard BERT fine-tuning without additional data augmentation or external unlabeled corpora.

3.4 Feature and Output Spaces

Each model family produces outputs in different representation spaces: LDA generates document-topic probability vectors, SVM produces decision margins that are converted into probabilities using Platt scaling, and LSTM and BERT generate softmax probabilities over the sentiment classes. To enable consistent aggregation across the heterogeneous models, all outputs are normalized to a common scale before being combined during the ensemble fusion stage.

3.5 Complementary Strengths and Limitations

Each model category contributes distinct capabilities to the ensemble: generative models provide unsupervised topic-level structure but lack fine-grained sentiment information, discriminative models perform strongly on structured datasets but can be sensitive to class imbalance, and embedding-based models are robust to linguistic variability but are generally more data-intensive and less interpretable. By combining these complementary characteristics, the ensemble mitigates the weaknesses of individual models and improves generalization. The robustness of such hybrid approaches has also been demonstrated in other applications, including autonomous manipulation of articulated objects [23] and covert wireless deception analysis [24].

4. Ensemble Strategy and Weighting Mechanism

The performance of individual sentiment classifiers often varies across datasets and linguistic contexts. To capitalize on the complementary strengths of our generative, discriminative, and embedding-based models,

we adopt a weighted ensemble strategy. This section details the design, optimization, and interpretability of our ensemble mechanism.

4.1 Motivation for Model Fusion

Ensemble learning has long been recognized for its ability to reduce generalization error by combining multiple hypotheses [25]. In sentiment analysis, where different models capture different linguistic cues lexical, syntactic, and semantic an ensemble can produce more robust predictions than any standalone classifier [26]. [27] provided an empirical study of popular ensemble methods, while [28] offered comprehensive coverage of combining pattern classifiers. The value of ensemble approaches is also evident in security applications, such as exploring device fingerprinting for password-less authentication systems [29].

4.2 Prediction Normalization and Calibration

Each base model produces a probability estimate for the positive class. To ensure that the probability representations from all models are comparable before fusion, we apply the following normalization procedure to each model's outputs:

1. For each model m , collect the probability predictions on the validation set: $\{p_m^{(1)}, \dots, p_m^{(V)}\}$.
2. Apply Platt scaling [30] to calibrate each model's outputs. Platt scaling fits a logistic regression model of the form:

$$\hat{p}_m(y = 1|\mathbf{x}) = \frac{1}{1 + \exp(A_m \cdot s_m(\mathbf{x}) + B_m)} \quad (2)$$

where $s_m(\mathbf{x})$ is the raw score (logit) from model m , and A_m, B_m are learned from validation data.

3. The calibrated probabilities $\hat{p}_m$ are used for fusion, rather than the original raw probabilities. For models that already produce well-calibrated probabilities (e.g., BERT with softmax), Platt scaling still provides an additional calibration step to ensure consistency across models.

This calibration procedure ensures that probabilities from all ensemble components are on a comparable scale and represent meaningful confidence estimates.

For the SVM classifier, decision scores are converted to probabilities using Platt scaling implemented in scikit-learn. The Platt scaling model is fitted on the validation set, learning the parameters A_{SVM} and B_{SVM} that map decision scores to calibrated probabilities. This ensures that SVM outputs are directly comparable with the probability outputs from LDA+LR (which produces probabilities via logistic regression), BiLSTM (which produces probabilities via softmax), and BERT (which produces probabilities via softmax). All model outputs are transformed into the same probability space before weight optimization and fusion.

4.3 Weighted Voting Mechanism and Learned Weights

Let $p_i \in \mathbb{R}^k$ be the normalized prediction vector from model i , where k is the number of sentiment classes. Let w_i be the weight assigned to model i . The final ensemble prediction $\hat{y}$ is computed as:

$$\hat{y} = \arg \max_j \left(\sum_{i=1}^n w_i \cdot p_i[j] \right)$$

Weights w_i are learned using cross-validation on a held-out development set to maximize macro-averaged F1 score.

4.3.1 Learned Weights

The learned weights for the four ensemble components on the IMDB dataset are $w_1 = 0.12$ for LDA+LR, $w_2 = 0.18$ for SVM, $w_3 = 0.25$ for BiLSTM, and $w_4 = 0.45$ for BERT. On the Rotten Tomatoes dataset, the corresponding weights are $w_1 = 0.14$, $w_2 = 0.16$, $w_3 = 0.20$, and $w_4 = 0.50$, respectively. These weights reflect the relative contribution of each model to the ensemble, with BERT receiving the highest weight on both datasets, consistent with its strong standalone performance.

4.4 Weight Optimization Procedure

We employ grid search over weight combinations subject to the constraint $\sum w_i = 1$. We evaluate ensemble performance on 5-fold cross-validation and select the optimal weights based on F1 score across all sentiment classes. This avoids overfitting and ensures balanced performance across underrepresented labels [27]. The optimization methodology shares conceptual similarities with portfolio optimization techniques that use algorithmic approaches with stochastic return functions [31].

4.5 Advantages of the Weighted Ensemble

- **Robustness:** The ensemble mitigates overfitting and handles out-of-distribution examples better than individual models.
- **Interpretability:** Model-level weights provide a measure of relative model importance at the aggregate level, indicating how much each component contributes to the final prediction on average. However, these aggregate weights do not constitute instance-level explanations of individual predictions. For instance-level interpretability, additional analysis such as attention visualization or SHAP/LIME-based explanations would be required. In this work, we focus on the aggregate contribution of each model to the ensemble performance, and we acknowledge the limitations of relying solely on model weights for interpretability. Future work could incorporate instance-level explanation methods to provide more granular interpretability.
- **Modularity:** New models can be integrated by assigning them weights without retraining the full system.

4.6 Summary

By designing a weighted ensemble with normalized prediction fusion, we achieve performance gains in both accuracy and generalization.

5. Results and Comparative Analysis

We evaluate the proposed ensemble on IMDB and Rotten Tomatoes datasets, reporting accuracy, macro F1, and AUC. For stochastic models (BiLSTM, BERT), mean $\pm$ std over five runs is reported; deterministic models use a single run.

Table 3: Performance comparison on IMDB (mean $\pm$ std).

Model	Accuracy (%)	F1 (Macro)	AUC
LDA+LR	82.3 $\pm$ –	0.81	0.86
SVM	84.1 $\pm$ –	0.83	0.89
BiLSTM	88.5 $\pm$ 0.4	0.86	0.91
BERT	91.5 $\pm$ 0.3	0.90	0.94
Ensemble (Ours)	93.1 $\pm$ 0.3	0.92	0.96

5.1 Ablation Study

Removing each component and re-optimizing weights shows BERT contributes most, followed by BiLSTM, with LDA+LR and SVM providing smaller complementary gains (Table 4).

Table 4: Ablation results on IMDB.

Configuration	Accuracy (%)	Drop (%)
Full Ensemble	93.1	–
Without LDA+LR	92.8	0.3
Without SVM	92.5	0.6
Without BiLSTM	92.0	1.1
Without BERT	89.5	3.6

5.2 Baseline Fusion Strategies

Learned weighting outperforms majority voting (91.2%) and uniform averaging (92.0%) on IMDB, with corresponding gains on Rotten Tomatoes (Table 5).

Table 5: Fusion strategy comparison.

Fusion Strategy	IMDB (%)	Rotten Tomatoes (%)
Majority Voting	91.2	88.7
Uniform Averaging	92.0	89.4
Learned Weighted (Proposed)	93.1	90.6

5.3 Comparison with State-of-the-Art

Our ensemble (93.1%) outperforms BERT-base (91.5%) and DistilBERT (92.1%), and is competitive with larger models like RoBERTa (93.8%) and XLNet (94.5%), while using fewer resources. On Rotten Tomatoes, our method achieves 90.6% accuracy.

5.4 Generalization Across Datasets

On Rotten Tomatoes, the ensemble achieves 90.6% accuracy, 0.89 macro F1, and 0.93 AUC, consistently outperforming standalone models (Table 6).

Table 6: Performance on Rotten Tomatoes.

Model	Accuracy (%)	Macro F1	AUC
LDA+LR	78.4	0.77	0.81
SVM	80.2	0.80	0.84
BiLSTM	85.1	0.83	0.87
BERT	87.3	0.86	0.89
Ensemble	90.6	0.89	0.93

5.5 Error Analysis

Analysis of 200 misclassified IMDB samples reveals error categories: negation (12%), sarcasm (10%), mixed sentiment (14%), ambiguity (8%), and length/intensifiers (56%) for the ensemble. These proportions are lower than for individual models, indicating improved robustness.

5.6 Computational Analysis

Experiments used an Intel Xeon Gold 6248 CPU, 256 GB RAM, and NVIDIA Tesla V100 GPU. Training times: LDA+LR (5 min), SVM (12 min), BiLSTM (45 min), BERT (2 h/epoch). Inference latency: ≈ 33 ms per sample. Parameter counts: LDA+LR (~ 100), SVM (~ 10 k support vectors), BiLSTM (2.5M), BERT (110M).

5.7 Reproducibility

All hyperparameters, random seeds, and learned weights are provided in Sections 2–4. These details, combined with the methodology, enable independent reproduction.

5.8 Summary

The ensemble approach yields consistent improvements across metrics and datasets, while offering resilience to the linguistic and structural variance typical of user-generated content.

6. Conclusion and Future Directions

In this study, we proposed a hybrid ensemble framework that integrates generative, discriminative, and embedding-based models for sentiment polarity detection in movie reviews. By leveraging the complementary strengths of these diverse model families, our system achieves superior accuracy and robustness compared to standalone classifiers.

We demonstrated that topic-based models such as LDA capture high-level thematic content, traditional machine learning models like SVM offer interpretable and efficient classification, while deep contextual models like BERT and LSTM enable fine-grained sentiment interpretation. Through a weighted voting mechanism, the ensemble aggregates model predictions, yielding consistent performance improvements across datasets.

Empirical evaluation on benchmark datasets such as IMDB and Rotten Tomatoes confirms that the ensemble model outperforms individual components in terms of accuracy, macro F1, and AUC. Moreover, the architecture supports extensibility and modularity—key requirements for real-world NLP deployments. We acknowledge the limitations of this work: the experiments are limited to binary classification of

English movie reviews, the ensemble incurs computational overhead due to running multiple models, and the contribution of individual components varies across datasets. Future work will extend the approach to multi-class sentiment classification, explore cross-domain and cross-lingual generalization, and investigate efficient distillation techniques to reduce the computational cost of the ensemble.

Limitation and bias awareness. Movie reviews on IMDB and Rotten Tomatoes predominantly reflect English-language, Western cultural biases. The ensemble may inadvertently amplify such biases if base models inherit skewed representations. Future work should evaluate the framework on diverse, multilingual, and balanced datasets, and investigate bias-mitigation strategies within the weighted voting mechanism. The approach to formalizing model integrity through typing systems, as explored in software product line design [17], could inform bias detection and mitigation strategies.

6.1 Limitations

This work has several limitations that should be acknowledged:

1. **Dataset Scope:** The evaluation is limited to two English movie-review datasets (IMDB and Rotten Tomatoes). The approach has not been validated on other sentiment domains (e.g., product reviews, social media, news articles) or in other languages, which may exhibit different sentiment expression patterns and vocabulary distributions.
2. **Binary Classification:** The experimental task is limited to binary sentiment classification. The approach has not been extended or evaluated for multi-class (positive/negative/neutral) sentiment classification.
3. **Dependence on Pretrained Representations:** The BiLSTM and BERT components rely on pretrained embeddings (GloVe and BERT pretraining) that were trained on large corpora. The ensemble's performance may vary with different pretraining sources or when applied to domain-specific text with vocabulary not well represented in the pretraining data.
4. **Computational Cost:** The ensemble requires running four independent models, resulting in increased training time (approximately 3 hours on a V100 GPU) and inference latency (≈ 33 ms per sample) compared to any single model. This may limit practical deployment in resource-constrained or real-time applications.
5. **No Multilingual Evaluation:** The current study does not include evaluation on non-English sentiment datasets. The performance of the ensemble in multilingual settings remains unexplored.
6. **Weight Transferability:** The learned weights are dataset-specific. Whether these weights can be transferred to other datasets without re-optimization is not examined.

6.2 Key Contributions

- A multi-paradigm ensemble integrating topic models, linear classifiers, and deep neural networks.
- A weighted voting strategy optimized via cross-validation for robust prediction fusion.
- Empirical validation showing performance gains across multiple benchmark datasets.

6.3 Future Work

While our results are promising, several avenues remain for exploration:

- **Domain Adaptation:** Extend the ensemble to new domains (e.g., product reviews, political discourse) using transfer learning or domain adaptation techniques [32].
- **Multilingual Sentiment Analysis:** Incorporate multilingual versions of BERT (e.g., mBERT, XLM-R) to analyze non-English content [33].
- **Explainability Integration:** Embed explainable AI components (e.g., SHAP, LIME) to highlight decision rationale at the token or sentence level [34].
- **Online Learning:** Adapt the ensemble to handle streaming data or evolving sentiment patterns through incremental model updates [35].
- **Confidence Calibration:** Explore ensemble uncertainty quantification to flag ambiguous or low-confidence predictions [36].
- **Efficient Nearest Neighbor Search:** Investigate efficient nearest neighbor search techniques [37] for scaling the ensemble to large datasets.
- **Dialogue Systems Integration:** Explore the application of our ensemble approach to conversational AI systems, drawing insights from attentive encoder-decoder chatbots trained on scripted dialogue [38].
- **Database Query Interfaces:** Investigate the integration of natural language database interfaces [39] with sentiment analysis ensembles for enhanced information retrieval.

6.4 Closing Remarks

As public discourse continues to shift online, the demand for accurate, interpretable, and robust sentiment classification systems grows. Ensemble architectures like the one presented here offer a scalable and modular approach to capturing sentiment across heterogeneous text sources. By combining models rooted in linguistics, probability, and deep learning, this work lays a foundation for future sentiment analysis systems that are both high-performing and trustworthy.

References

- [1] Bing Liu. *Sentiment analysis and opinion mining*. Morgan & Claypool Publishers, 2012.
- [2] Bo Pang and Lillian Lee. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1–2):1–135, 2008.
- [3] Deepit Sapru. Analyzing crime discourse in u.s. metropolitan communities on reddit: Trends, influences, and insights. 2025. doi: 10.48550/arXiv.2512.18227.
- [4] Christopher D Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.
- [5] Ranjith Kumar Ramakrishnan. Hybrid decision tree and genetic algorithm for enhanced classification accuracy. *TechRxiv*. URL <https://doi.org/10.36227/techrxiv.174163792.23092397/v1>.

- [6] Lei Zhang, Shuai Wang, and Bing Liu. Deep learning for sentiment analysis: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(4), 2018.
- [7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186. Association for Computational Linguistics, 2019. doi: 10.18653/v1/N19-1423.
- [8] Wajiha Abdul Shakir. Enhancing named entity recognition through neural architectures. In *2024 21st International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP)*, pages 1–5, 2024. doi: 10.1109/ICCWAMTIP64812.2024.10873731.
- [9] Tanish Gupta. High-performance deep learning for large-scale image recognition: A gpu-accelerated convolutional approach. 2026. URL <https://ieeexplore.ieee.org/document/11409295>.
- [10] Ashish Vaswani et al. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.
- [11] Nagajayant Nagamani. Scaling reasoning in ai: Challenges of long-context understanding in emerging models. *Research and Reviews: Advancement in Robotics*, 9(1):21–30, 2025. doi: 10.5281/zenodo.17231097.
- [12] Lili Xu et al. Bert-based ensemble model for sentiment analysis. In *IEEE Access*, 2019.
- [13] Sumit Mamtani. Enhancing transformer-based vision models: Addressing feature map anomalies through novel optimization strategies, 2025.
- [14] Abhi Desai. Active learning strategies for efficient text classification. 2025. doi: 10.2139/ssrn.5209772.
- [15] Krishna Karan Ghantasala. Pediatric t1d copilot: Lstm-driven glucose forecasting and conversational decision support from real-world cgm, insulin, and meal streams. 2026. doi: 10.1109/ICCCES62661.2026.11436160.
- [16] Gnanendra Bachu. Empowering data-driven decision making: The rise of self-service data wrangling in the age of automation. 2026. URL <https://ieeexplore.ieee.org/document/11280864/>.
- [17] Rakesh Reddy Thalakanti. Formalizing feature model integrity: A typing system and refactoring approaches for improving software product line design. 2025. doi: 10.1049/icp.2025.4792.
- [18] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.
- [19] Ivan Titov and Ryan McDonald. Modeling online reviews with multi-grain topic models. In *Proceedings of the 17th International Conference on World Wide Web*, 2008.
- [20] Marx Prathap Singh Shanmugam. Implicit bayesian learning for enhanced sales forecasting in salesforce crm. 2025. doi: 10.1109/ICERECT65215.2025.11378062.
- [21] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

- [22] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *EMNLP*, 2014.
- [23] Aakash Gadh. Robust autonomous manipulation of articulated objects. *International Journal of Mechanical and Mechatronics Engineering*. doi: 10.7763/ijmo.
- [24] Kaushal Thaker. Covert wireless deception: Unmasking a protocol vulnerability in authenticated wi-fi links. doi: 10.31224/5117. URL <https://doi.org/10.31224/5117>.
- [25] Thomas G Dietterich. Ensemble methods in machine learning. *International Workshop on Multiple Classifier Systems*, pages 1–15, 2000.
- [26] Lior Rokach. Ensemble-based classifiers. *Artificial Intelligence Review*, 33(1):1–39, 2010.
- [27] David Opitz and Richard Maclin. Popular ensemble methods: An empirical study. *Journal of Artificial Intelligence Research*, 11:169–198, 1999.
- [28] Ludmila I Kuncheva. *Combining Pattern Classifiers: Methods and Algorithms*. Wiley, 2004.
- [29] Shantanu Gujar. Exploring device fingerprinting for password-less authentication systems. doi: 10.1109/10862062. URL <https://ieeexplore.ieee.org/document/10862062>.
- [30] John C. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *Advances in Large Margin Classifiers*, pages 61–74. MIT Press, 1999.
- [31] Aditya Saxena. Portfolio optimization with stochastic return functions: An algorithmic approach. URL <https://ieeexplore.ieee.org/document/11156675>.
- [32] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *ICML*, 2011.
- [33] Alexis Conneau et al. Unsupervised cross-lingual representation learning at scale. *ACL*, 2020.
- [34] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should i trust you?": Explaining the predictions of any classifier. In *KDD*, 2016.
- [35] Steven CH Hoi et al. Online learning: A comprehensive survey. *Neurocomputing*, 2018.
- [36] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *NeurIPS*, 2017.
- [37] Shreya G. Tendulkar. Efficient nearest neighbor search in large-scale text data using optimized locality sensitive hashing in distributed computing environments. In *2025 International Conference on Advanced Computing Technologies (ICoACT)*, pages 1–6, 2025. doi: 10.1109/ICoACT63339.2025.11005204.
- [38] Sourabh Rajput. Conversations from make-believe: An attentive encoder-decoder chatbot trained on scripted dialogue. doi: 10.21203/rs.3.rs-8716036/v1.
- [39] Pramath Parashar. A prolog-based natural language database with a procedural interface. 2026. doi: 10.1109/ICCTWC68241.2026.11583360.