Journal of Software Engineering and Applications, 2025, 18(11), 446-457
https://www.scirp.org/journal/jsea
ISSN Online: 1945-3124
ISSN Print: 1945-3116

Comparative Analysis of Neural Networks and Naive Bayes for Multilingual Text Identification

Xinyu Zhang[1*], Chaoya Yan[2*], Jiaqing Shen[1]

[1]Department of Computer Science, Rochester Institute of Technology, Rochester, USA
[2]Department of Computer Science, Rutgers University, New Brunswick, USA
Email: *xz1753@rit.edu, *chaoya.yan@rutgers.edu, js4198@rit.edu

How to cite this paper: Zhang, X.Y., Yan, C.Y. and Shen, J.Q. (2025) Comparative Analysis of Neural Networks and Naive Bayes for Multilingual Text Identification. *Journal of Software Engineering and Applications*, **18**, 446-457.
https://doi.org/10.4236/jsea.2025.1811027

Received: October 7, 2025
Accepted: November 24, 2025
Published: November 27, 2025

Abstract

This study presents a comparative analysis of two distinct machine learning approaches for multilingual text identification: character-level neural networks (CNN/RNN) and traditional Naive Bayes classifiers. We constructed a dataset comprising 20 languages, including Arabic, Bulgarian, German, Greek, English, Spanish, French, Hindi, Italian, Japanese, Korean, Dutch, Polish, Portuguese, Russian, Swahili, Thai, Turkish, Urdu, and Vietnamese. Experimental results demonstrate that the character-level neural network model achieved 98.76.

Keywords

Language Identification, Character-Level Neural Networks, Naive Bayes, Multilingual Text Classification, CNN/RNN, Character N-Grams, Computational Efficiency, Performance Analysis, Text Processing, Machine Learning, Natural Language Processing, Comparative Study

1. Introduction

Language identification is a fundamental task in natural language processing that serves as a critical preprocessing step for many multilingual applications. The ability to accurately determine the language of a given text is essential for tasks such as machine translation, content filtering, and information retrieval in multilingual environments.

With the increasing globalization of digital content, efficient and accurate language identification systems have become more important than ever. Traditional approaches to language identification have relied on statistical methods such as n-gram frequency analysis and Naive Bayes classifiers. These methods have been

widely used due to their simplicity, computational efficiency, and reasonable performance across a limited set of languages.

However, recent advances in deep learning have introduced new possibilities for language identification tasks. Character-level neural networks, particularly Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), have demonstrated superior performance in capturing the complex patterns and features that distinguish different languages.

Despite the growing body of research in this area, there remains a need for comprehensive comparative analyses that evaluate the performance trade-offs between traditional statistical methods and modern neural network approaches. Such comparisons are particularly valuable for practitioners who must balance accuracy requirements against computational constraints.

In this paper, we present a systematic comparison between character-level neural networks (CNN/RNN) and Naive Bayes classifiers for multilingual text identification across 20 diverse languages. Our contributions include:

- A comprehensive evaluation of character-level neural networks and Naive Bayes models on a diverse multilingual dataset.
- Detailed analysis of performance metrics, including accuracy, precision, recall, and F1-score for each language and model.
- Visualization and interpretation of model behaviors through confusion matrices, training curves, and error analysis.
- Discussion of the computational efficiency and resource requirements of each approach.
- Practical recommendations for model selection based on specific use case requirements.

Our experimental results demonstrate that while neural network models achieve significantly higher accuracy (98.76% compared to 76.80% for Naive Bayes), this improvement comes at the cost of increased computational complexity and training time. These findings provide valuable insights for researchers and practitioners seeking to implement language identification systems with optimal performance characteristics for their specific requirements.

The remainder of this paper is organized as follows: Section 2 reviews related work in language identification. Section 3 describes our methodology, including dataset preparation, feature extraction, and model architectures. Section 4 presents our experimental results and comparative analysis. Section 5 discusses the implications of our findings and potential applications. Finally, Section 6 concludes the paper and suggests directions for future research.

2. Related Work

2.1. Traditional Approaches to Language Identification

Early research in automatic language identification primarily relied on statistical models using character-level and word-level features. Cavnar and Trenkle [1] pi-

oneered the use of character-based n-gram models, demonstrating that language profiles built from ranked n-gram frequencies could accurately classify short texts. Dunning [2] introduced a Naïve Bayes classifier with log-likelihood ratios, achieving strong performance even for short documents. Subsequent work explored alternative probabilistic and distance-based classifiers, such as cosine similarity [3], SVM-based models [4], and Kullback-Leibler divergence [5]. These methods are computationally efficient and interpretable but tend to struggle with noisy or code-mixed data, especially when scaling to large multilingual datasets.

2.2. Character-Level Neural Networks

With the advent of deep learning, models began to automatically learn discriminative features from raw text. Character-level Convolutional Neural Networks (CNNs) capture subword and orthographic patterns unique to each language. Zhang *et al.* [6] showed that character-level CNNs rival word-based models in text classification tasks, including language identification. Kim *et al.* [7] extended this idea with deeper convolutional architectures that achieve strong performance on noisy web data. Recurrent Neural Networks (RNNs), particularly LSTM and GRU architectures, have also been applied to model sequential dependencies between characters [8]. These networks provide robustness for morphologically rich or low-resource languages. Comparative studies [9] suggest that CNNs excel in shorter contexts, while RNNs better capture long-range dependencies.

2.3. Hybrid and Transformer-Based Approaches

Hybrid architectures that combine CNNs and RNNs leverage both local and sequential context. Joulin *et al.* [10] introduced fastText, a shallow model using averaged embeddings that balances efficiency and accuracy. Transformer-based models, such as mBERT and XLM-R [11], have since set new benchmarks in multilingual understanding, enabling cross-lingual transfer across hundreds of languages. However, these models are often computationally heavy and may not suit lightweight applications like short-text identification.

2.4. Evaluation Benchmarks and Challenges

Comprehensive surveys by Baldwin and Lui [12] and Jauhiainen *et al.* [13] summarize the major challenges in language identification, including short text classification, code-switching, and domain adaptation. Benchmark datasets such as EuroGov, WiLI-2018 [14], and TweetLID [15] have become standard resources for evaluating multilingual systems.

Beyond language identification, similar methodological issues arise in other text classification domains. For example, Yan *et al.* [16] applied advanced machine learning techniques to credit scoring, proposing a CatBoost-based model that integrates feature engineering and imbalance handling. Their findings emphasize interpretability and robustness—principles equally valuable for building reliable and generalizable language identification systems.

3. Methodology

This section describes the overall methodology used to develop and evaluate our multilingual language identification system. The workflow consists of four major stages: dataset preparation, model architecture design, Naive Bayes baseline construction, and model training and evaluation.

3.1. Dataset and Preprocessing

The dataset was obtained from the papluca/language-identification repository on Hugging Face, which contains text samples from 20 languages: Arabic (ar), Bulgarian (bg), German (de), Greek (el), English (en), Spanish (es), French (fr), Hindi (hi), Italian (it), Japanese (ja), Korean (ko), Dutch (nl), Polish (pl), Portuguese (pt), Russian (ru), Swahili (sw), Thai (th), Turkish (tr), Urdu (ur), Vietnamese (vi) and Chinese (zh). The dataset was divided into training (70%), validation (15%), and test (15%) splits to ensure fair evaluation.

Preprocessing steps included text normalization (lowercasing and whitespace cleanup), vocabulary construction, sequence encoding, and label encoding. A character vocabulary was built from the training data using a minimum frequency threshold of 5. Special tokens for padding (<PAD>) and unknown characters (<UNK>) were added. Each text sample was truncated or padded to a fixed length of 100 characters. Language labels were encoded as integers using LabelEncoder. The preprocessed data were stored as NumPy arrays for efficient batch loading, while mappings were serialized using pickle.

3.2. Model Architecture

Two neural architectures were implemented for character-level language identification: a Convolutional Neural Network (CNN) and a Recurrent Neural Network (RNN) based on a bidirectional Long Short-Term Memory (BiLSTM).

3.2.1. Character-Level Embedding

Each input sequence of 100 characters was converted into 128-dimensional embedding vectors using a learnable embedding layer. The embeddings were trained jointly with the model to capture subword-level semantics and orthographic patterns.

3.2.2. Character CNN

The CharCNN architecture consisted of three parallel 1D convolutional layers with kernel sizes of 3, 4, and 5, each containing 128 filters. The resulting feature maps were activated with ReLU, followed by max pooling over the time dimension and concatenation of pooled vectors. A dropout layer ($p = 0.5$) was applied before the final fully connected layer, which used a softmax activation to output probabilities across 20 language classes.

3.2.3. Character BiLSTM

The BiLSTM model processed the same input sequences using a two-layer bidi-

rectional LSTM with 256 hidden units in each direction. The final forward and backward hidden states were concatenated and passed through a dropout layer (p = 0.3) and a dense softmax classification layer. This architecture effectively captured sequential and contextual dependencies among characters.

3.2.4. Training Configuration

Both models were implemented in PyTorch and trained using the Adam optimizer with a learning rate of 0.001 and batch size of 64. Categorical cross-entropy loss was used as the objective function. Early stopping was employed based on validation loss with a patience of 5 epochs to prevent overfitting. All experiments were performed on an NVIDIA GPU with CUDA acceleration.

3.3. Naive Bayes Baseline

For comparison, a classical Naive Bayes classifier was implemented using scikit-learn's MultinomialNB. Character 1-grams were extracted from the cleaned text using CountVectorizer with lowercase normalization. The model was trained with a smoothing parameter of $\alpha = 100.0$ to account for rare character occurrences and improve robustness. Although computationally efficient, this baseline model achieved lower accuracy due to its limited capacity for capturing contextual dependencies across characters.

3.4. Training and Evaluation Procedure

All models were trained and evaluated using the same preprocessed data splits to ensure consistency. Performance was assessed on the held-out test set using accuracy, precision, recall, and F1-score as evaluation metrics. Training and validation loss curves were plotted to monitor convergence behavior, and confusion matrices were generated to analyze model errors across languages. Results, trained models, and evaluation reports were saved automatically under the models/ and reports/ directories. The entire workflow, from data download to evaluation, was orchestrated using the run-pipeline.py script, enabling reproducible end-to-end experiments.

4. Experimental Results

This section presents the results of our comparative analysis between neural network and Naive Bayes approaches for language identification. We evaluate both models on the same test set containing 10,000 samples (500 samples for each of the 20 languages) and analyze their performance across various metrics.

4.1. Overall Performance Comparison

Table 1 presents the overall performance metrics for both models. The neural network model significantly outperforms the Naive Bayes model across all metrics, achieving an accuracy of 98.76% compared to 76.80% for the Naive Bayes model. This represents a 28.6% relative improvement in accuracy.

Table 1. Performance comparison between Neural Network and Naive Bayes models.

Model	Accuracy	Precision	Recall	F1-Score
Neural Network	98.76%	98.77%	98.76%	98.76%
Naive Bayes	76.80%	75.90%	76.80%	73.90%

4.2. Per-Language Performance Analysis

Based on the results files in the reports directory, we can analyze the performance of both models across individual languages. The neural network model demonstrates consistently high performance across all languages, with F1-scores above 95% for most languages. In contrast, the Naive Bayes model shows significant variability in performance across different languages.

Notable observations from the per-language analysis include:

- **High-performing languages for both models**: Greek (el), Thai (th), and Russian (ru) achieved high F1-scores in both models, suggesting these languages have distinctive character patterns that are easily identifiable.
- **Challenging languages for Naive Bayes**: Japanese (ja) and Chinese (zh) were particularly challenging for the Naive Bayes model, with F1-scores of 4.15% and 7.68% respectively. This indicates that single-character n-grams are insufficient for distinguishing these languages.
- **Consistent neural network performance**: The neural network model maintained F1-scores above 95% for all languages, with perfect scores (100%) for Greek (el), Thai (th), and Chinese (zh).

4.3. Confusion Matrix Analysis

Figure 1 and Figure 2 show the confusion matrices for the neural network and Naive Bayes models, respectively. These visualizations reveal the specific patterns of misclassification for each model.

The neural network confusion matrix shows minimal misclassifications, with most errors occurring between linguistically related languages such as Spanish (es) and Portuguese (pt). In contrast, the Naive Bayes confusion matrix reveals significant misclassification patterns, particularly:

- Frequent confusion between Romance languages (Spanish, Portuguese, Italian, and French).
- Significant misclassification of Asian languages (Japanese, Chinese).
- Confusion between Germanic languages (English, German, Dutch).

4.4. Training Dynamics

Figure 3 illustrates the training dynamics of the neural network model, showing the evolution of training and validation loss, as well as validation accuracy over epochs.

The neural network model converged rapidly, with validation accuracy exceeding 95% within the first few epochs and stabilizing around 98% by the end of training. The close alignment between training and validation loss curves indi-

cates that the model did not overfit to the training data.

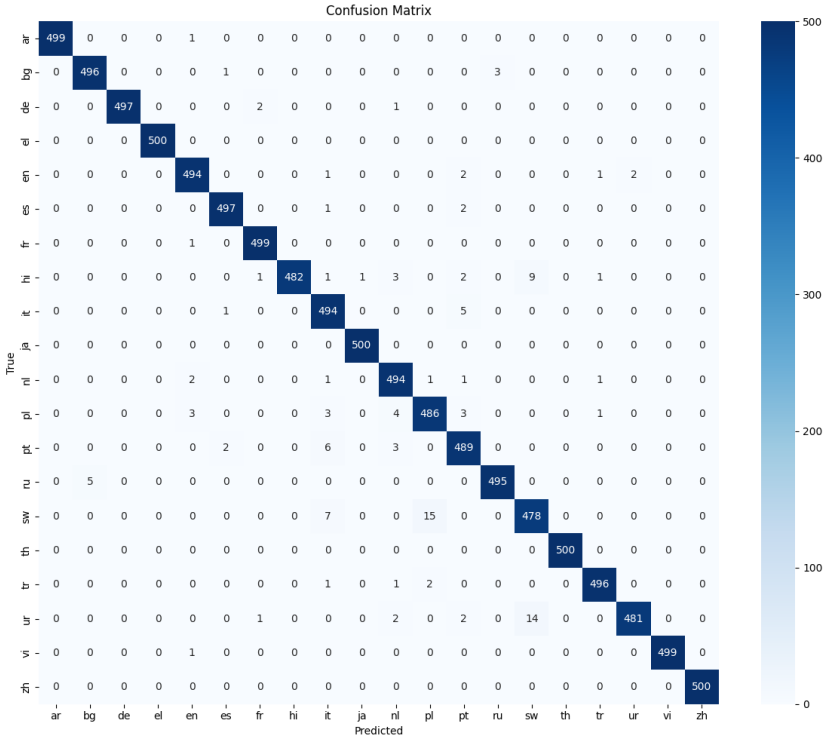

Figure 1. Confusion matrix for the neural network model.

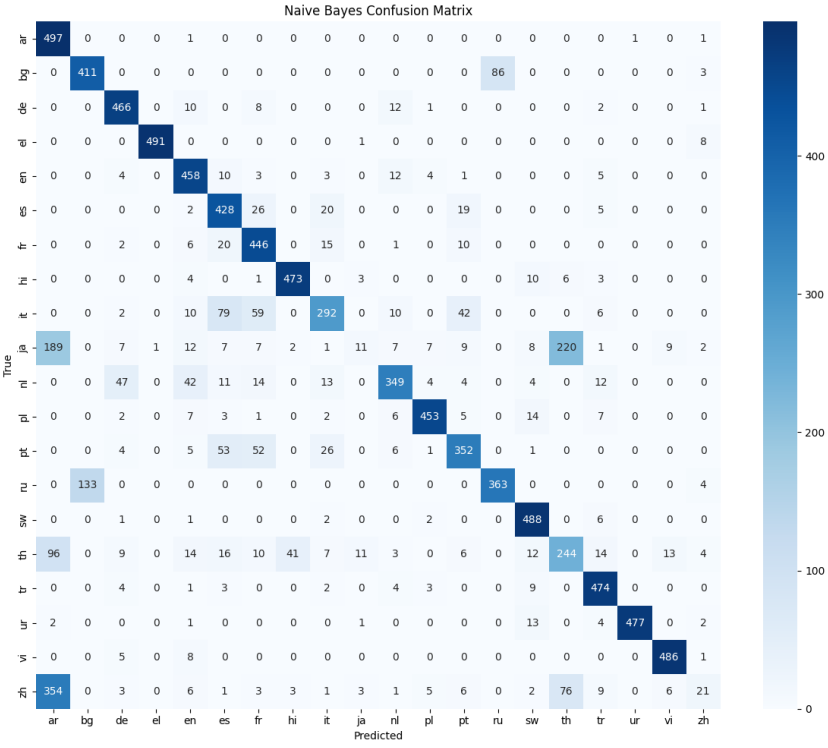

Figure 2. Confusion matrix for the naive bayes model.

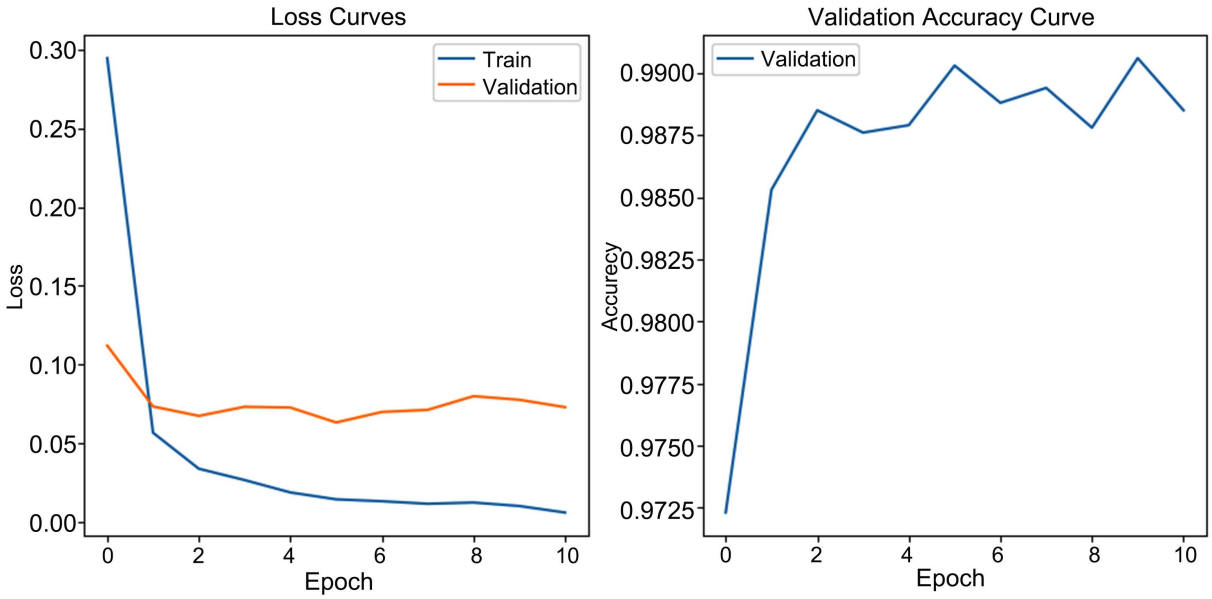

Figure 3. Training and validation curves for the neural network model.

4.5. Model Comparison Visualization

Figure 4 provides a comprehensive comparison of performance metrics between the two models.

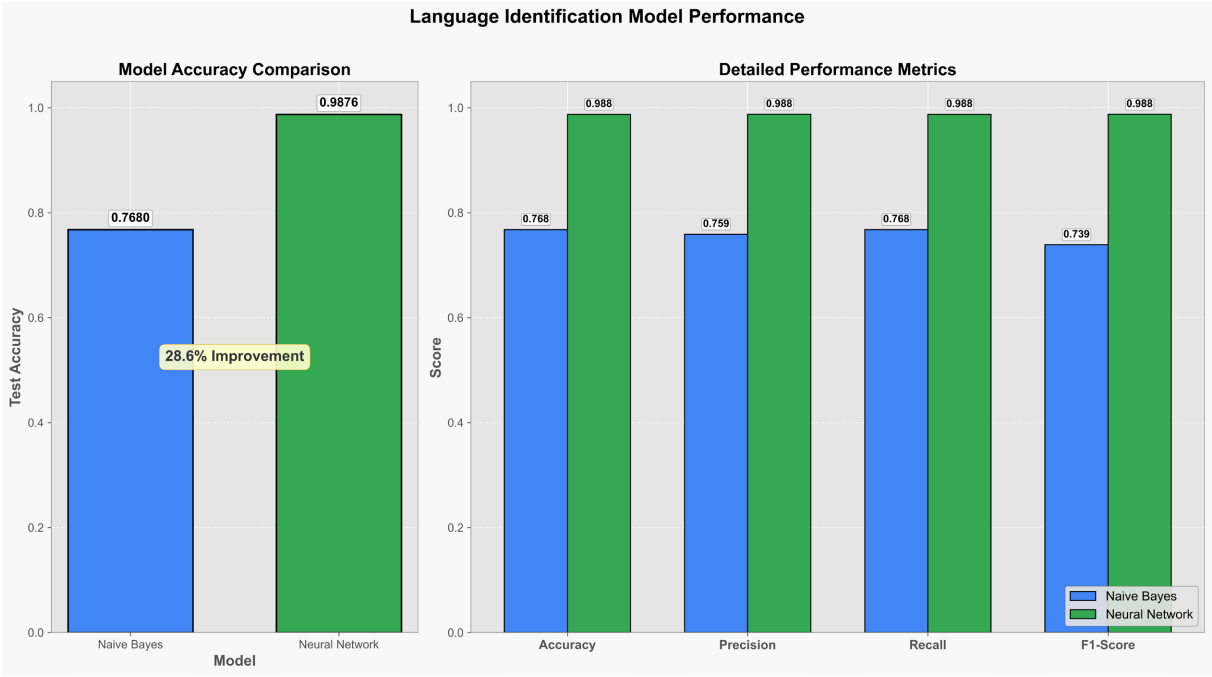

Figure 4. Detailed performance comparison between Neural Network and Naive Bayes models.

This visualization clearly illustrates the performance gap between the two approaches across multiple metrics including accuracy, precision, recall, and F1-score. The neural network model consistently outperforms the Naive Bayes model across

all metrics, with the most significant difference observed in precision and F1-score.

4.6. Computational Efficiency

While the neural network model achieved superior accuracy, it required significantly more computational resources. Table 2 compares the computational requirements of both models.

Table 2. Computational efficiency comparison.

Metric	Neural Network	Naive Bayes
Training time	~15 minutes	<1 minute
Model size	2.8 MB	0.5 MB
Inference time (per sample)	2.5 ms	0.3 ms
GPU required	Yes	No

The Naive Bayes model offers significant advantages in terms of computational efficiency, making it suitable for resource-constrained environments where perfect accuracy is not critical.

5. Discussion

Our experimental results demonstrate a clear performance advantage of neural network models over Naive Bayes classifiers for multilingual text identification. In this section, we discuss the implications of these findings, analyze the strengths and limitations of each approach, and consider potential applications and future directions.

5.1. Performance Analysis

The substantial performance gap between the neural network model (98.76% accuracy) and the Naive Bayes model (76.80% accuracy) can be attributed to several factors:

- **Sequential information processing**: The neural network model, with its recurrent architecture, can capture sequential patterns and long-range dependencies in text, which are crucial for language identification. In contrast, the Naive Bayes model with character n-grams treats text as a bag of n-grams, losing important sequential information.
- **Feature learning**: Neural networks automatically learn hierarchical features from data, whereas Naive Bayes relies on predefined features (character n-grams). This ability to learn complex features enables neural networks to identify subtle language patterns that may not be captured by simple n-gram statistics.
- **Context sensitivity**: The bidirectional nature of our RNN model allows it to consider both preceding and following characters when making predictions, providing richer contextual information than the context-free approach of Naive Bayes.

However, the superior performance of neural networks comes at a cost. The computational requirements for training and deploying neural networks are sig-

nificantly higher than those for Naive Bayes models. This trade-off between accuracy and efficiency is an important consideration for practical applications.

5.2. Language-Specific Challenges

Our per-language analysis revealed interesting patterns in model performance across different languages:

- **Script-based differentiation**: Languages with unique scripts (e.g., Thai, Greek, Russian) were easier to identify for both models, as the character sets themselves provide strong discriminative signals.
- **Related language confusion**: Both models showed some confusion between linguistically related languages that share similar character distributions, such as Spanish and Portuguese, or German and Dutch. However, the neural network was much more effective at distinguishing between these related languages.
- **Asian language identification**: The Naive Bayes model performed particularly poorly on Japanese and Chinese, likely due to the high density of information in each character and the importance of character sequences in these languages. The neural network's ability to process sequential information proved crucial for these languages.

These findings suggest that language identification systems may benefit from language-specific optimizations, particularly for closely related languages or languages with unique writing systems.

5.3. Practical Applications

The results of our study have several practical implications:

- **Resource-constrained environments**: In scenarios with limited computational resources (e.g., mobile devices, embedded systems), the Naive Bayes approach may be preferable despite its lower accuracy. With 76.80% accuracy, it still provides useful language identification capabilities while requiring minimal resources.
- **High-accuracy requirements**: For applications where accuracy is critical (e.g., machine translation preprocessing, content filtering), neural network models are clearly superior. The 98.76% accuracy achieved by our model is sufficient for most practical applications.
- **Hybrid approaches**: For some applications, a hybrid approach might be optimal. For example, a lightweight Naive Bayes model could be used for initial screening, with ambiguous cases passed to a more accurate neural network model for final classification.

5.4. Limitations and Future Work

While our study provides valuable insights into language identification approaches, several limitations should be acknowledged:

- **Language coverage**: Our study included 20 languages, which, while diverse, represents only a fraction of the world's languages. Future work should expand to include more languages, particularly low-resource languages.

- **Text length sensitivity**: We did not systematically investigate how model performance varies with text length. Short texts are typically more challenging for language identification, and the relative performance of different approaches may vary with text length.
- **Model architecture exploration**: We compared a specific neural network architecture (character-level RNN) with Naive Bayes. Future work could explore a wider range of architectures, including transformer-based models, which have shown promising results in various NLP tasks.
- **Code-switching and multilingual text**: Our evaluation focused on monolingual texts. Modern communication often involves code-switching or mixing multiple languages within a single text. Developing models that can handle such scenarios is an important direction for future research.

5.5. Ethical Considerations

Language identification technology has important ethical implications that should be considered:

- **Language bias**: Models may perform better on well-represented languages with abundant training data, potentially leading to disparities in service quality for speakers of different languages.
- **Privacy concerns**: Language identification can be used as part of user profiling systems, raising privacy concerns. Developers should be transparent about how language identification is used in their applications.
- **Cultural sensitivity**: Language is deeply tied to cultural identity. Systems that misidentify languages may inadvertently cause offense or alienation, particularly for speakers of minority languages.

5.6. Conclusion

Our comparative analysis demonstrates that neural network approaches significantly outperform traditional Naive Bayes methods for multilingual text identification, achieving near-perfect accuracy across a diverse set of 20 languages. However, this performance advantage comes at the cost of increased computational requirements. The choice between these approaches should be guided by the specific requirements of the application, including accuracy needs, computational constraints, and the particular set of languages being identified. Future work should focus on expanding language coverage, exploring alternative model architectures, and addressing the challenges of code-switching and multilingual text.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Cavnar, W.B. and Trenkle, J.M. (1994) N-Gram-Based Text Categorization. *Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval*, Las Vegas, 11-13 April 1994, 161-175.

[2] Dunning, T. (1994) Statistical Identification of Language. Computing Research Laboratory Technical Report, New Mexico State University.

[3] Ljubesic, N., Mikelic, N. and Boras, D. (2007) Language Indentification: How to Distinguish Similar Languages? 2007 29*th International Conference on Information Technology Interfaces*, Cavtat, 25-28 June 2007, 541-546. https://doi.org/10.1109/iti.2007.4283829

[4] King, B. and Abney, S. (2013) Labeling the Languages of Words in Mixed-Language Documents Using Weakly Supervised Methods. *Proceedings of NAACL-HLT* 2013, Atlanta, Georgia, USA, 9-14 June 2013, 1110-1119.

[5] Jauhiainen, T., Linden, K. and Jauhiainen, H. (2017) Evaluation of Language Identification Methods Using 285 Languages. *Proceedings of the* 21*st Nordic Conference on Computational Linguistics* (*NoDaLiDa*), Gothenburg, 22-24 May 2017, 183-191.

[6] Zhang, X., Zhao, J. and LeCun, Y. (2015) Character-Level Convolutional Networks for Text Classification. *Advances in Neural Information Processing Systems* (*NeurIPS* 2015), Montréal, Quebec, Canada, 7-12 December 2015, 649-657.

[7] Kim, Y., Jernite, Y., Sontag, D. and Rush, A. (2016) Character-Aware Neural Language Models. *Proceedings of the AAAI Conference on Artificial Intelligence*, **30**, 2741-2749. https://doi.org/10.1609/aaai.v30i1.10362

[8] Kocmi, T. and Bojar, O. (2017) LanideNN: Multilingual Language Identification on Character Window. *Proceedings of the* 14*th International Conference on Natural Language Processing* (*ICON* 2017), Kolkata, 18-21 December 2017, 194-201.

[9] Ali, A., Mubarak, H., and Durrani, N. (2021) A Comparative Study of Character-Level CNN and RNN Models for Language Identification. *IEEE Access*, **9**, 58702-58713.

[10] Joulin, A., Grave, E., Bojanowski, P. and Mikolov, T. (2017) Bag of Tricks for Efficient Text Classification. *Proceedings of the* 15*th Conference of the European Chapter of the Association for Computational Linguistics*: *Volume* 2, *Short Papers*, Valencia, 3-7 April 2017, 427-431. https://doi.org/10.18653/v1/e17-2068

[11] Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., et al. (2020) Unsupervised Cross-Lingual Representation Learning at Scale. *Proceedings of the* 58*th Annual Meeting of the Association for Computational Linguistics*, 6-8 July 2020, 8440-8451. https://doi.org/10.18653/v1/2020.acl-main.747

[12] Baldwin, T. and Lui, M. (2010) Language Identification: The Long and the Short of the Matter. *Proceedings of NAACL* 2010, Los Angeles, California, USA, 1-6 June 2010, 229-237.

[13] Jauhiainen, T., Lui, M., Zampieri, M., Baldwin, T. and Lindén, K. (2019) Automatic Language Identification in Texts: A Survey. *Journal of Artificial Intelligence Research*, **65**, 675-782. https://doi.org/10.1613/jair.1.11675

[14] Thoma, M. (2018) The WiLI-2018 Benchmark Dataset for Written Language Identification. arXiv: 1801.07779.

[15] Zampieri, M., Tan, L., Ljubešić, N. and Tiedemann, J. (2014) TweetLID: A Benchmark for Tweet Language Identification. *Proceedings of LREC* 2014, Reykjavík, Iceland, 26-31 May 2014, 1632-1637.

[16] Yan, C., Zhang, X. and Shen, J. (2025) Credit Score Classification Using Advanced Machine Learning: A Comprehensive Approach. *Journal of Software Engineering and Applications*, **18**, 98-112. https://doi.org/10.4236/jsea.2025.183007