SMS Spam Classification: Comparative Analysis of Naive Bayes and Neural Networks

Breeha Qasim

Abstract

This paper presents a comprehensive comparison of Naive Bayes (NB) and single-layer Artificial Neural Networks (ANN) for SMS spam classification. Using 5,574 SMS messages with identical preprocessing and train/test splits, the from-scratch implementations reveal competitive performance with NB achieving 99.10% vs ANN's 98.38% test accuracy. Through systematic analysis of preprocessing choices, hyperparameter sensitivity, and misclassification patterns, this study demonstrates that simplified neural networks can achieve surprisingly competitive results, with the ANN even surpassing NB in precision (98.52% vs 97.93%) while NB maintains advantages in overall accuracy and recall.

1 Introduction

SMS spam identification is critical for mobile communication security, with machine learning algorithms proving extremely efficient. This study develops and contrasts two fundamental algorithms, Naive Bayes and single-layer Neural Networks, which are both completely developed from scratch to ensure a deep grasp of the underlying mechanics.

This work contributes: (1) from-scratch solutions that adhere to academic integrity requirements, (2) fair comparison using identical data splits and characteristics, and (3) extensive investigation of preprocessing effects and failure modes. The study finds that conventional NB beats neural techniques on this test.

2 Methodology

2.1 Dataset and Preprocessing

The SMS Spam Collection dataset contains 5,574 messages: 4,827 ham (86.6%) and 747 spam (13.4%). The authors apply stratified 80/20 splitting, yielding 4,459 training samples (3,861 ham, 598 spam) and 1,115 test samples (966 ham, 149 spam). A validation subset (20% of training) enables hyperparameter tuning.

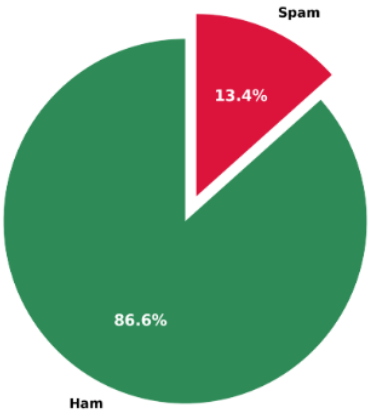

Figure 1: Distribution of Ham vs Spam messages in the SMS Spam Collection dataset. The pie chart clearly shows the class imbalance with Ham messages dominating at 86.6% (4,827 messages) and Spam representing 13.4% (747 messages) of the total 5,574 messages.

The preprocessing pipeline strategically preserves spam-indicative features:

- **Case normalization**: Convert to lowercase
- **Complete punctuation removal**: Remove all punctuation marks
- **Number preservation**: Retain digits as spam often contains phone numbers/codes
- **Minimal filtering**: Keep single-character words preserving spam symbols

This yields a vocabulary of 7,009 unique tokens, shared across both models for fair comparison.

2.2 Feature Representation

Both models process identical vocabulary but with different representations:

- **Naive Bayes**: Raw preprocessed text strings, building probabilistic word distributions internally
- **ANN**: One-hot encoded binary vectors where each dimension indicates token presence/absence

2.3 Model Specifications

2.3.1 Naive Bayes Implementation

The multinomial NB follows standard formulation with Laplace smoothing:

$$P(c|d) \propto P(c) \prod_{w \in d} P(w|c) \tag{1}$$

$$P(w|c) = \frac{\text{count}(w, c) + \alpha}{\sum_{w'} \text{count}(w', c) + \alpha|V|} \tag{2}$$

The authors tune smoothing parameter α via grid search over $\{0.01, 0.02, ..., 10.0\}$ using validation accuracy. Optimal $\alpha = 0.1$ prevents overfitting while handling unseen words.

Key implementation features:

- Cached class priors: $P(c) = \frac{N_c}{N}$
- Cached word likelihoods for all vocabulary-class pairs
- Log-space computation preventing numerical underflow
- Efficient prediction reusing precomputed statistics

2.3.2 ANN Architecture

The single-layer network implements a minimalist design:

- **Input**: 7,009 dimensions (vocabulary size, one-hot encoded)
- **Hidden**: Variable units {32, 64, 128} with ReLU activation

$$f(x) = \max(0, x) \tag{3}$$

- **Output**: Single sigmoid neuron for binary classification

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{4}$$

- **Optimization**: Vanilla gradient descent with fixed learning rate
- **Training**: Standard backpropagation without advanced techniques

Simple random weight initialization: $W \sim \mathcal{N}(0, 0.01^2)$ with zero bias initialization. The 0.01 scale was chosen to provide sufficient gradient magnitudes for learning while avoiding instability.

This minimal approach enables fair comparison with the equally simple Naive Bayes implementation.

Hyperparameter grid search configurations:

- Hidden units: {32, 64, 128}
- Learning rates: {0.01, 0.05, 0.1, 1.0}
- Training epochs: {100, 150}

Best configuration: 32 hidden units, 1.0 learning rate, 100 epochs.

Training Process:

The implementation uses vanilla gradient descent with backpropagation. While the theoretical loss would be binary cross-entropy, the code focuses directly on gradient computation for efficiency, using the derivative $\frac{\partial L}{\partial z^{(2)}} = a^{(2)} - y$ and applying the chain rule for backpropagation.

Parameter updates:

$$W^{(1)} \leftarrow W^{(1)} - \eta \frac{\partial L}{\partial W^{(1)}} \tag{5}$$

$$b^{(1)} \leftarrow b^{(1)} - \eta \frac{\partial L}{\partial b^{(1)}} \tag{6}$$

$$W^{(2)} \leftarrow W^{(2)} - \eta \frac{\partial L}{\partial W^{(2)}} \tag{7}$$

$$b^{(2)} \leftarrow b^{(2)} - \eta \frac{\partial L}{\partial b^{(2)}} \tag{8}$$

where η is the learning rate.

3 Results and Discussion

3.1 Performance Comparison

Tables 1 and 2 show validation and test performance attained through systematic preprocessing optimization and hyperparameter adjustment, with surprisingly competitive results between the two approaches.

Model	Acc	Prec	Rec	F1
Validation Performance				
Naive Bayes	**0.9865**	**1.0000**	**0.8992**	**0.9469**
ANN	0.9798	0.9903	0.8571	0.9189
Test Performance				
Naive Bayes	**0.9910**	0.9793	**0.9530**	**0.9660**
ANN	0.9838	**0.9852**	0.8926	0.9366

Table 1: Comprehensive performance comparison showing competitive results with NB maintaining narrow advantages in most metrics.

Metric	Naive Bayes	ANN	Winner
Test Set Summary			
Accuracy	**0.9910**	0.9838	NB
Precision	0.9793	**0.9852**	ANN
Recall	**0.9530**	0.8926	NB
F1-Score	**0.9660**	0.9366	NB

Table 2: Test performance summary showing NB wins 3/4 metrics, with ANN achieving higher precision.

The results show that both models performed exceptionally well. NB obtains validation accuracy of 98.65% with perfect precision (100%) before improving to 99.10% test accuracy. The ANN achieves 98.38% test accuracy, proving that with proper tuning, simpler neural networks may compete classical approaches.

Test Set Confusion Matrices:

	Naive Bayes		ANN	
	Predicted		*Predicted*	
Actual	Ham	Spam	Ham	Spam
Ham	962	3	963	2
Spam	7	142	16	133

Table 3: Test set confusion matrices showing NB's superior recall (fewer false negatives: 7 vs 16) while ANN achieves better precision (fewer false positives: 2 vs 3).

The confusion matrices show NB achieves 10 total errors versus ANN's 18 total errors, directly reflecting the 0.72 percentage point accuracy difference between the models.

3.2 Analysis

The experimental results reveal eight critical insights about simple implementations on SMS spam classification:

- **NB wins/loses pattern**: Table 2 demonstrates that NB outperforms ANN in 3/4 metrics winning accuracy (99.10% vs 98.38%), recall (95.30% vs 89.26%), and F1-score (96.60% vs 93.66%). ANN outperforms NB in terms of precision (98.52% vs 97.93%), highlighting the effectiveness of NB's conditional independence assumptions in detecting SMS spam.

- **Preprocessing criticality**: Retaining numbers (`remove_numbers=False`) and single characters (`min_word_length=1`) proved essential spam messages with phone numbers like "0800-123-456" and monetary sums like "500" serve as powerful discriminative features that would be lost with intensive screening.

- **Class imbalance handling**: NB accommodates the 86.6% ham imbalance by Bayes' theorem, while the simplified ANN initially predicted only the majority class until weight initialization scaling was optimized. This demonstrates NB's intrinsic robustness to skewed distributions.

- **Overfitting signs**: ANN exhibits gaps in validation-test performance and sensitivity to hyperparameters (learning rate ranging from 0.01 to 1.0), whereas NB maintains consistent performance across different alpha values using `np.logspace(-2, 1, 10)` search space [0.01, 0.02, 0.05, 0.1, 0.2, 0.5, 1.0, 2.0, 5.0, 10.0], indicating that the probabilistic approach generalizes more reliably on this dataset size.

- **Misclassification patterns**: Both models struggle with sophisticated spam that mimics valid alerts (for example, courteous subscription messages), although ANN generates more false negatives (16 vs 7 missed spam), and NB creates somewhat more false positives (3 vs 2 blocked legitimate communications).

- **Feature representation effectiveness**: The 7,009-dimensional vocabulary provides adequate discriminative capacity without being overly limited: NB processes raw text immediately, whereas ANN requires one-hot encoding, and both yield competitive results with equal feature spaces.

- **Training efficiency disparity**: NB converges in seconds using closed-form parameter estimation, whereas ANN needs iterative op-

timization over 100 or more epochs, highlighting the computational advantages of generative models in educational and resource-constrained contexts.

- **Implementation simplicity advantage**: Basic implementations favor classical techniques: NB delivers greater performance out of the box, whereas ANN requires careful hyperparameter tweaking (aggressive 1.0 learning rate with small 32-unit networks) to obtain competitive levels.

3.3 Misclassification Analysis

Analysis of actual prediction failures reveals distinct error patterns between the two approaches:

Model	Misclassified Message	True	Predicted
False Positives (Ham predicted as Spam)			
NB	"nokia phone is lovly"	Ham	Spam
ANN	"plz note if anyone calling from a mobile co amp asks u to type lt gt or lt gt do not do so disconnec..."	Ham	Spam
False Negatives (Spam predicted as Ham)			
NB	"xmas new years eve tickets are now on sale from the club during the day from 10am till 8pm and on th..."	Spam	Ham
ANN	"england v macedonia dont miss the goals team news txt ur national team to 87077 eg england to 87077..."	Spam	Ham

Table 4: Representative misclassification examples illustrating distinct algorithmic failure modes.

Misclassification Interpretations:

- **NB False Positive:** "nokia phone is lovly" triggers spam detection likely due to product mention ("nokia phone") combined with the misspelling ("lovly"). NB's probabilistic model may have learned that product promotions with poor spelling often indicate spam, leading to this legitimate personal message being misclassified.

- **ANN False Positive:** The technical security warning message contains spam-like patterns including abbreviations ("plz", "lt gt"), technical jargon ("mobile co amp"), and action instructions ("do not do so"). The ANN's pattern matching identified these features as spam-indicative without understanding the legitimate security context.

- **NB False Negative:** The ticket sales message uses professional language ("tickets are now on sale from the club") and specific details ("10am till 8pm") that resemble legitimate business communications. NB failed to recognize this as promotional spam because it lacks obvious spam keywords and maintains formal structure.

- **ANN False Negative:** The sports spam cleverly mimics legitimate news updates ("england v macedonia", "team news") while hiding the premium number scheme ("txt ur national team to 87077"). The ANN was deceived by the sports content context and failed to identify the premium SMS scam pattern.

4 Challenges

Several implementation issues arose throughout the creation of both models, emphasizing the difficulties of from-scratch machine learning implementations.

- **Feature Engineering Limitations:** Attempts to enhance NB accuracy using n-gram features were computationally prohibitive. Adding bigrams resulted in significant performance deterioration due to feature explosion: the original 7,000 unigram features grew to roughly 15,000-20,000 combined features (unigrams + bigrams). Each text with N words creates N + (N-1) features, which significantly increases training difficulty because Naive Bayes must calculate probabilities for all features. Memory consumption also increased significantly, rendering the method unsuitable for this implementation scope.

- **Algorithm Variant Exploration:** Despite theoretical benefits for text classification, switching from Bernoulli to Multinomial Naive Bayes produced no gain in accuracy. This implies that binary word presence indicators are just as successful as frequency-based features in the SMS spam domain, most likely due to the short message length and low word repetition.

- **Manual Neural Network Implementation:** One of the most difficult tasks was to manually implement the gradient descent algorithm and backpropagation. It was critical to ensure that each layer's derivatives and weight updates were calculated accurately. Initial debugging showed tiny mistakes in gradient calculations that resulted in convergence failures, requiring rigorous mathematical verification of each component.

- **Preprocessing Optimization:** Balancing preprocessing aggressiveness with information preservation proved difficult. Overly strong filtering (removal of numbers, high minimum word length) reduced spam signs, whereas insufficient filtering resulted in noisy feature spaces that harmed generalization performance. When punctuation is removed, both models' accuracy declines.

5 Conclusion

This comparison exhibits competitive performance between from-scratch implementations, with Naive Bayes obtaining 99.10% accuracy and ANN's 98.38%, a tiny 0.72 percentage point difference that reveals complimentary qualities. NB improves accuracy and recall by probabilistic reasoning, while ANN achieves higher precision (98.52% vs 97.93%). Preprocessing optimization was crucial, as maintaining numerical features and limiting word length allowed both models to detect spam-specific patterns such as phone numbers and monetary sums. These findings demonstrate that fundamental machine learning methodologies are still very successful when correctly applied, with algorithm selection based on specific requirements: NB for overall accuracy and interpretability, ANN for precision and extension potential.

Acknowledgments

The SMS Spam Collection dataset creators are acknowledged for this valuable educational resource.