Comparative Analysis of Logistic Regression, Neural Networks, and SVM for Urdu Text Classification

Natiq Khan
25110027

Muhammad Nafees
26100029

Danish Athar
25100174

Usama Habib
25110235

Waleed Ali
24100200

Abstract

This study investigates the classification of Urdu news articles into predefined categories using machine learning models: Logistic Regression, Neural Networks, and Support Vector Machines (SVM). The dataset, scraped from prominent Urdu news websites, was preprocessed with TF-IDF vectorization. Results show that while Neural Networks provided the most balanced performance across categories, the SVM struggled with imbalanced data. This research highlights the challenges and potential improvements for Urdu text classification, contributing to personalized news systems for underserved audiences.

ACM Reference Format:
Natiq Khan, Muhammad Nafees, Danish Athar, Usama Habib, and Waleed Ali. 2024. Comparative Analysis of Logistic Regression, Neural Networks, and SVM for Urdu Text Classification. In . ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

Automatically classifying news articles is important in today's world, where we generate large amounts of information every day. While we have made great progress in processing English text, languages like Urdu, which millions of people speak, are still not well represented in research. Creating models for classifying Urdu news is essential for improving access to local content, personalizing news delivery, and enhancing media monitoring for Urdu-speaking audiences.

This project aims to solve these challenges by scraping and analyzing Urdu news data, conducting exploratory data analysis, and building classification models from scratch. By not relying on prebuilt libraries, we intend to gain a better understanding of the issue and help develop resources specifically for Urdu.

2 Methodology

2.1 Data Preparation

The dataset included Urdu text from Samaaa, Jang, Express, and Geo news websites which was scraped using Python b4Soup and selenium. It was collected according to the following categories: Business, Entertainment, Sports, Science-Technology, and World. A basic cleaning was done on this data before further pre-processing could be done. This involved removing null entries and checking for duplicates etc.

Preprocessing Urdu text data posed unique challenges due to the language's morphological richness and script complexity. Unlike English, Urdu often uses contextual phrases, where multiple words combine to convey specific meanings, making normalization and tokenization essential. Text normalization, implemented using the UrduHack library, addressed inconsistencies such as diacritics, spacing errors, and varying representations of characters across sources like newspapers. Without normalization, comparing words with a predefined stop word list would have been unreliable due to script discrepancies, as the same word could appear in multiple forms.

After normalization, the text was tokenized using the SpaCy library, effectively segmenting the text into word tokens while accommodating Urdu's linguistic nuances. Stop words—common words that do not add significant meaning—were removed at this stage. While the TF-IDF method inherently reduces the impact of commonly occurring words, explicit stop word removal provided an additional layer of refinement, ensuring better results by eliminating unnecessary noise from the data.

The final step involved transforming the preprocessed text into numerical features using TF-IDF, incorporating n-grams to capture the contextual relationships between words. This was especially crucial for Urdu, as its reliance on multi-word phrases often carries significant semantic weight (for e.g "ڈونلڈ ٹرمپ" and "اے آئی" are two-worded phrases that make sense together). These preprocessing steps collectively ensured a consistent, clean, and meaningful dataset, optimizing it for machine learning model training.

In the end, the dataset comprised 2281 entries divided as such :

- 450 - Business
- 455 - Entertainment
- 457 - Science-Technology
- 458 - Sports
- 461 - International

For TF-IDF, the minimum document frequency was set to 2 while the maximum was set to 0.95. The ngram range varied from model to model. The logistic regression used (1, 4) range to capture unigram to 4-gram features for richer contextual representation while filtering out overly rare or common terms. For the Neural Network, the ngram range was (1,2) to prevent over-fitting and limiting computational time. The data was then randomized to remove any unintended patterns that might bias the model.For the Neural Network Model and Support Vector Machine Model, the dataset was

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
Conference'17, July 2017, Washington, DC, USA
© 2024 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

preprocessed into numerical feature vectors of size 56,800. The train test split was 80:20.

2.2 Logistic Regression

The logistic regression model was chosen for multi-class classification due to its interpretability, efficiency, and ability to estimate probabilities using the softmax function. The model was optimized with gradient descent and used a stopping criterion based on convergence. This approach allowed for effective classification, leveraging the extracted features while balancing computational efficiency and predictive power.

2.3 Neural Network

The initial Neural Network architecture comprised an input layer, the first hidden layer that had 128 neurons, and the second that had 64 neurons. Each uses ReLU activation. To prevent overfitting, dropout layers with a rate of 0.5 were added after each hidden layer. The final output layer had five neurons, one for each class, and the model used cross-entropy loss to optimize classification performance. The learning rate of 0.001 and 20 epochs were used during which the model's training and validation accuracy, as well as loss, were monitored. Early stopping with a patience of five epochs was implemented to halt training if validation performance ceased to improve. precision, recall, and F1-score were computed for each class, along with macro and weighted averages.

The improved model increased the size of the hidden layers to 256 and 128 neurons, respectively, and adjusted the dropout rate to 0.3 for a better balance between regularization and information retention. The learning rate was also reduced to 0.0005. This model was trained for 25 epochs, with early stopping applied using the same criteria as the baseline model. Evaluation of the enhanced model followed the same approach as the baseline.

2.4 Neural Network

The SVM model was implemented as a custom PyTorch module designed for multi-class classification. SVM is a powerful supervised machine learning algorithm that excels at classification tasks by identifying an optimal hyperplane that maximizes the margin between data points belonging to different classes. In this implementation, the one-vs-rest (OvR) strategy was used, where a separate binary classifier was trained for each class, enabling the model to handle multi-class classification effectively. It features a single linear layer to calculate the raw class scores, with the regularization parameter (C = 5.0) incorporated for controlling the margin-size trade-off. The training utilized a hinge loss function, which penalizes misclassifications and ensures that correctly classified instances are adequately separated from the decision boundary. Balanced class weights were also calculated and applied to account for imbalances in the dataset categories, further refining the model's predictions.

The optimization process used stochastic gradient descent (SGD) with a learning rate of 0.0001, a momentum value of 0.95, and weight decay of 0.001 for L2 regularization to simplify decision boundaries. Training spanned 1000 epochs, during which early stopping was employed with a patience of 200 epochs to prevent overfitting. The model incorporated regularization by adding penalties for large weights and biases, promoting simpler decision boundaries. The early stopping mechanism monitored validation accuracy, saving the best-performing model weights. This tailored implementation ensures that the SVM effectively manages the sparse and high-dimensional nature of the dataset, leveraging its strengths in linear separability for text classification.

3 Discussion

The comparison between the performance of the Logistic Regression (LR), Neural Network (NN) and Support Vector Machine (SVM) models demonstrates the varying strengths and nuances of these three approaches in text classification tasks. The aim of this discussion is to analyze their performance, highlight their key differences, and provide insights into the implications of model choice for real-world applications.

The model for the improved Neural Network did not perform comparatively better than the initial Neural Network as can be seen by the confusion matrix of each model (figure 1 and figure 2 respectively) The noted efficiencies were 0.9452 (for the initial model) and 0.9474(for the improved model) despite doubling the number of nodes in each layer and tweaking the learning rate and epochs. This may be due to the small size of the dataset. A third hidden layer may help given the dataset is improved. Therefore it suffices to use the evaluations of the improved Neural Network for comparison.

The Logistic Regression Model and Neural Network, both had an accuracy of 94%. This could be due to the different ngram ranges of the models. A unigram to quadgram range for the regression model provided enough feature sets bypassing the linearity limitation. The unigram-bigram range was ideal for the Neural Network model as it uses non-linear learning capabilities. Using a larger range would not however be ideal as it may lead to potential overfitting unless the dataset is also proportionally increased.

The SVM model, in comparison, achieved an accuracy of 64.25%, which is significantly lower than the Logistic Regression and Neural Network models. This lower accuracy can be attributed to the sparse nature of TF-IDF features and SVM's reliance on linear separability. The model used a unigram to trigram range for TF-IDF vectorization, which was chosen to strike a balance between capturing contextual information and minimizing the sparsity of features, given the linear nature of the SVM algorithm.

Observing individual categories, The Regression model had 100% precision in the sports category while the NN had 99%. However, The recall and f1-scores are significantly higher for the NN showing that it performed better overall in capturing relevant and non-relevant articles.

For the Science-Technology category, the LR showed a significantly higher precision than the Neural Network with similar recall and f1-scores. This could be due to terms like "اے آئی" and "گوگل" which mark the category allowing accurate classifications.

The confusion matrix for the Logistic Regression (figure 3) shows that none of the predictions were mislabeled as entertainment. This may have a similar reason as the Science-Technology category. However, it has significantly lower precision and also outputs a lower f1-score than the Neural Network. The confusion matrix

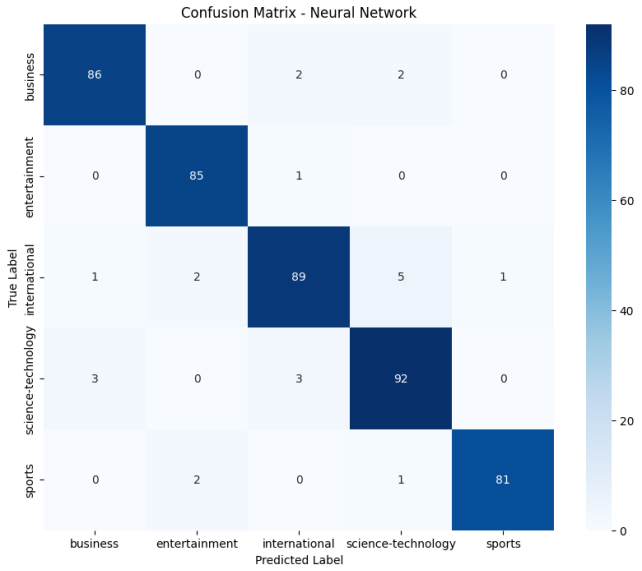

Figure 1: Initial Neural Network Confusion Matrix

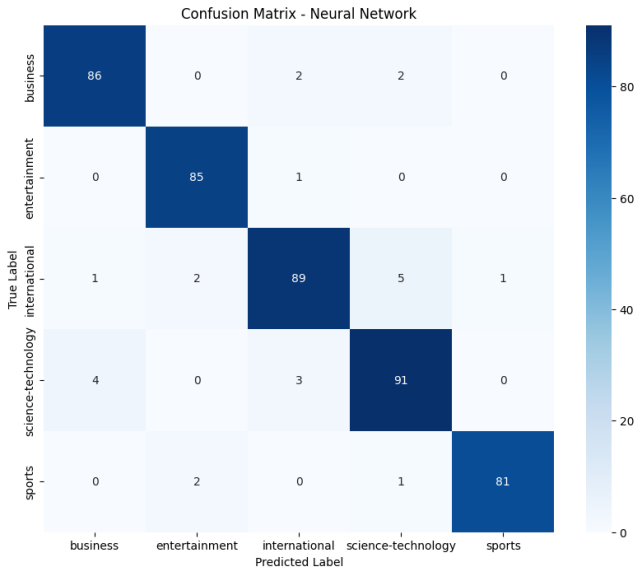

Figure 2: Improved Neural Network Confusion Matrix

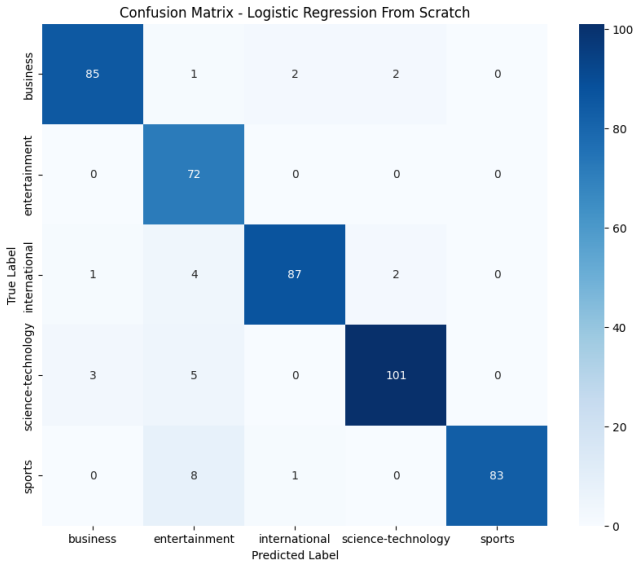

Figure 3: Logistic Regression Confusion Matrix

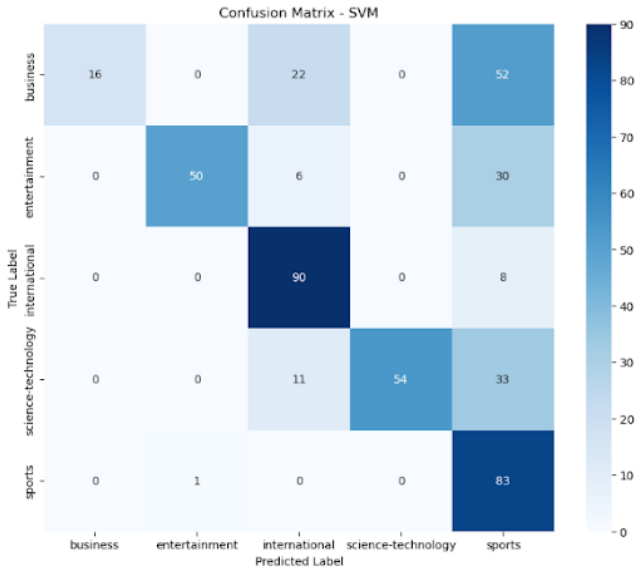

Figure 4: SVM Confusion Matrix

(Figure 4) reveals that the model performed exceptionally well in some categories but struggled significantly in others.

For the business category, the SVM achieved a precision of 100% but with a very low recall of 18%, leading to an F1-score of 0.30. This indicates that while the model identified true positives with perfect precision, it missed a large number of relevant articles, many of which were misclassified into the sports category.

In the entertainment category, the model performed better, achieving a precision of 98% and a recall of 58%, resulting in an F1-score of 0.73. Despite some misclassifications, this suggests that the SVM managed to balance identifying true positives while minimizing false positives reasonably well.

The international category saw robust performance, with 90 correct predictions and only 8 instances misclassified as sports, resulting in a recall of 92%, a precision of 70%, and an F1-score of 0.79. This highlights the SVM's ability to effectively identify patterns specific to this category.

For the science-technology category, the model achieved a perfect precision of 100% but struggled with recall at 55%, leading to an F1-score of 0.71. While the model avoided false positives, it misclassified 33 articles from this category into sports, indicating

overlapping features between these two classes in the TF-IDF vector space.

The sports category had the lowest precision at 40% but an exceptionally high recall of 99%, resulting in an F1-score of 0.57. This indicates that the model successfully captured most relevant articles in this category but misclassified many irrelevant articles, especially from business and science-technology, as sports.

4 Conclusion

Overall, the SVM model demonstrated inconsistent performance, excelling in categories but struggling in others. In contrast, the Neural Network performed in a more balanced manner across all categories compared to both the Logistic Regression and SVM models

5 Limitations & Future Considerations

The dataset's small size and imbalanced category distribution likely impacted the performance of all models, particularly the SVM. Increasing the dataset size, refining feature engineering techniques, and experimenting with ensemble models could enhance classification accuracy. Additionally, incorporating domain-specific Urdu embeddings and leveraging pre-trained models may further improve performance for underrepresented categories.