NEURAL NETWORK- TEXT CLASSIFICATION

Akhila Singaraju

**INTRODUCTION:**

In the quest to adopt the most effective model for handwriting recognition for the school, I compared the K-Nearest Neighbors (KNN) model with a neural network model using several benchmarking metrics. This comparison helps us ascertain which model performs better in terms of accuracy, precision, recall, and F1-score.

**MODEL 1 EVALUATION: KNN MODEL:**

The K-Nearest Neighbors (KNN) model is often used for handwriting prediction because it effectively classifies data based on the similarity of input features to samples in the training set.

For the first model, I utilized the letters.csv dataset to develop a K-Nearest Neighbors (KNN) model for handwriting prediction. I configured the model to use 20 neighbors with a weighting based on the Euclidean distance to potentially enhance the accuracy of classifying handwritten characters.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | | **Class** | **Precision** | **Recall** | **F1-Score** | | 0 | 0.84 | 0.88 | 0.86 | | 1 | 0.77 | 0.96 | 0.85 | | 2 | 0.69 | 0.61 | 0.65 | | 3 | 0.62 | 0.56 | 0.59 | | 4 | 0.72 | 0.52 | 0.60 | | 5 | 0.66 | 0.56 | 0.60 | | 6 | 0.82 | 0.87 | 0.85 | | 7 | 0.48 | 0.61 | 0.54 | | 8 | 0.62 | 0.53 | 0.57 | | 9 | 0.46 | 0.49 | 0.48 | | **Accuracy** |  |  | **0.67** | |

The performance of the model on the test data yielded an accuracy of about 67%.

* **Precision** varied across classes, from a low of 0.46 for class 9 to a high of 0.84 for class 0, demonstrating differences in the model's accuracy in predicting positive instances across classes.
* **Recall** was highest for class 1 at 0.96, showing it correctly identified most of its actual instances, while it was lowest for class 3 at 0.56.
* The **F1-Score**, which considers both precision and recall, ranged from 0.48 for class 9 to 0.86 for class 0, indicating effectiveness varied significantly among classes.

**Encountered Challenges:**

1. **Optimizing Parameters**: Selecting the appropriate 'k' value and the distance metric was critical. A small 'k' can make the model overly sensitive to noise, while a large 'k' can be computationally intensive and may skew results towards the more frequent classes.
2. **Handling Dimensionality**: The high dimensionality of the data was a challenge due to increased computational demands and the potential issues with the curse of dimensionality, where distances lose meaning.

**CONFUSION MATRIX: \*\* see appendix picture 1 for visualization and table of confusion matrix.**

The confusion matrix presented above details the performance of my KNN model in classifying handwritten characters, with each row representing the actual class and each column representing the predicted class. The numbers along the diagonal, such as 754 for class 0 and 932 for class 1, indicate the instances where the model's predictions matched the actual labels, highlighting the model's accuracy for those classes. Misclassifications are noted in off-diagonal cells, like the 56 misclassifications of class 2 as class 3, and the 100 misclassifications of class 3 as class 1, suggesting visual or stylistic similarities that confused the model.

**MODEL 2 EVALUATION: NEURAL NETWORK**

For 2nd model, I developed a neural network model using the letters dataset to predict handwriting accurately. This model consists of a series of dense layers with ReLU activation functions, which introduce non-linearity to the model, helping it learn complex patterns in the data. The output layer uses a SoftMax activation because this is a multi-class classification problem, ensuring the output values are probabilities that sum to one across all classes. The model achieved an accuracy of about 70% on the test data.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | | **Class** | **Precision** | **Recall** | **F1-Score** | | 0 | 0.88 | 0.90 | 0.89 | | 1 | 0.83 | 0.96 | 0.89 | | 2 | 0.76 | 0.64 | 0.70 | | 3 | 0.67 | 0.58 | 0.63 | | 4 | 0.78 | 0.58 | 0.67 | | 5 | 0.65 | 0.67 | 0.66 | | 6 | 0.88 | 0.89 | 0.89 | | 7 | 0.49 | 0.71 | 0.58 | | 8 | 0.65 | 0.58 | 0.61 | | 9 | 0.49 | 0.45 | 0.47 | | **Accuracy** |  |  | **0.70** | |

The classification report for my neural network model provides key metrics that detail the model's performance across different classes of handwritten characters. The **precision** metric, which indicates how many selected items are relevant, shows values such as 0.88 for class 0 and lower values like 0.49 for class 9, suggesting that the model is more reliable in identifying certain classes than others. The **recall** metric, which measures how many relevant items are selected, is highest for class 1 at 0.96, indicating that the model is particularly effective at identifying nearly all instances of this class. However, for class 9, the recall is only 0.45, highlighting a struggle to detect all instances of this class correctly. The **F1-score**, which balances precision and recall, reflects this mixed accuracy, with higher scores around 0.89 for classes 0 and 1, and lower scores, like 0.47 for class 9.

**CONFUSION MATRIX: \*\* see appendix picture 2 for visualization and table of confusion matrix.**

The confusion matrix shown provides a detailed breakdown of the predictions made by my neural network model for handwriting classification. The diagonal values, such as 776 for class 0 and 928 for class 1, represent the correctly predicted instances for each class, indicating where the model performed well. Off-diagonal values show misclassifications, such as class 2 being mistaken 51 times for class 3 and class 7 being confused 155 times with class 9, highlighting specific areas where the model confused one class for another. These misclassifications are particularly noteworthy between classes that may have similar handwriting features, leading to higher error rates. The matrix is crucial for pinpointing which classes are more challenging for the model to distinguish, thereby guiding potential improvements in the model’s training or architecture.

**Encountered Challenges:**

 **Class Imbalance and Overfitting**: Handling class imbalance was crucial, as disproportionate data volumes led to overfitting, where the model excelled at recognizing over-represented classes but faltered with under-represented ones. Strategies like adjusting the loss function or rebalancing the dataset were necessary to enhance model generalization.

 **Hyperparameter Tuning**: Optimizing the neural network's configuration, such as the number of layers, neurons, dropout rates, and learning rates, was challenging. These parameters significantly impact learning and generalization, requiring careful adjustment and validation to prevent overfitting and underfitting, often at the expense of computational resources and time.

**MODEL COMPARISON:**

To determine the best model for handwriting recognition, I compared the performance of both the K-Nearest Neighbors (KNN) model and the neural network model using various benchmarking metrics. Below is a summary table showing the accuracy, precision, recall, and F1-score for both models:

| **Metric** | **KNN Model** | **Neural Network Model** |
| --- | --- | --- |
| Accuracy | 0.67 | 0.70 |
| Precision | 0.67 | 0.71 |
| Recall | 0.66 | 0.70 |
| F1-Score | 0.66 | 0.70 |

### Summary of Findings:

* **Accuracy**: The neural network model shows a slight improvement in overall accuracy at 70%, compared to 67% for the KNN model. This suggests that the neural network is better at correctly identifying all classes.
* **Precision**: The average precision of the neural network model is higher at 71%, indicating it has a lower rate of false positives compared to the KNN model which is at 67%.
* **Recall**: The neural network also leads in recall with 70%, meaning it successfully identifies a higher percentage of relevant instances across all classes compared to the KNN model's 66%.
* **F1-Score**: Reflecting a balance between precision and recall, the F1-score is also higher for the neural network at 70%, compared to 66% for the KNN model, suggesting better overall performance.

### Recommendation:

Based on these metrics, I recommend the school adopt the **neural network model** for handwriting recognition. The neural network not only provides higher accuracy, precision, recall, and F1-score but also offers better handling of complex patterns and class differentiation, crucial for effective handwriting recognition. Furthermore, with its ability to improve through additional layers and training adjustments, the neural network holds potential for further enhancements with more data or refined training techniques.

**CONCLUSION:**

The neural network model outperforms the KNN model across all key performance metrics, making it the preferred choice for handwriting recognition at our school. Its superior performance in handling complex data patterns and potential for further optimization makes it highly suitable for educational applications where accuracy and adaptability are crucial.

**APPENDIX:**

**FIGURE 1: CONFUSION MATRIX: KNN MODEL**

| **True Label \ Predicted Label** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 754 | 1 | 17 | 4 | 16 | 9 | 30 | 14 | 3 | 13 |
| 1 | 0 | 932 | 4 | 2 | 6 | 6 | 3 | 4 | 9 | 5 |
| 2 | 45 | 41 | 498 | 56 | 19 | 7 | 12 | 41 | 66 | 32 |
| 3 | 7 | 23 | 100 | 463 | 7 | 51 | 13 | 28 | 102 | 40 |
| 4 | 18 | 63 | 1 | 9 | 415 | 28 | 27 | 150 | 13 | 78 |
| 5 | 9 | 33 | 19 | 90 | 23 | 415 | 44 | 40 | 24 | 47 |
| 6 | 24 | 21 | 15 | 12 | 16 | 11 | 718 | 0 | 4 | 0 |
| 7 | 8 | 8 | 10 | 14 | 31 | 47 | 0 | 554 | 14 | 228 |
| 8 | 17 | 78 | 53 | 78 | 14 | 37 | 27 | 24 | 422 | 39 |
| 9 | 20 | 11 | 10 | 19 | 28 | 21 | 0 | 297 | 25 | 416 |

A diagram of a confusion matrix

Description automatically generated

**FIGURE 2: CONFUSION MATRIX: NEURAL NETWORK**

| **True Label \ Predicted Label** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 776 | 1 | 10 | 7 | 16 | 7 | 19 | 11 | 9 | 5 |
| 1 | 0 | 928 | 2 | 5 | 6 | 4 | 2 | 1 | 13 | 10 |
| 2 | 42 | 22 | 526 | 51 | 15 | 15 | 15 | 41 | 63 | 27 |
| 3 | 3 | 15 | 83 | 487 | 9 | 64 | 4 | 32 | 91 | 46 |
| 4 | 10 | 43 | 3 | 3 | 469 | 21 | 18 | 156 | 14 | 65 |
| 5 | 3 | 13 | 5 | 72 | 20 | 496 | 24 | 48 | 19 | 44 |
| 6 | 15 | 10 | 5 | 6 | 13 | 36 | 734 | 0 | 2 | 0 |
| 7 | 7 | 9 | 3 | 10 | 23 | 46 | 0 | 649 | 12 | 155 |
| 8 | 10 | 63 | 51 | 70 | 16 | 41 | 17 | 28 | 457 | 36 |
| 9 | 17 | 10 | 8 | 13 | 14 | 29 | 0 | 351 | 27 | 378 |

A diagram of a confusion matrix

Description automatically generated