A Comparative Analysis of FNN and SVM for MBTI Personality Prediction

KUAN YU CHEN
u1527521@umail.utah.edu

Sheng-Hung Tseng
u1541147@umail.utah.edu

Abstract

This study compares the performance of two models: Support Vector Machine (SVM) and Fully Connected Neural Network (FNN). The SVM model has an accuracy of 82%, and the FNN has an accuracy with 90%. This study uses confusion matrices highlights that the FNN model has fewer misclassifications between classes. The result tells that the FNN is more effective at capturing complex patterns and distinguishing between classes.

1. Introduction

The MBTI test is currently very popular among the young generation. However, traditional MBTI tests required answering a lot of questions, which can be boring and may not always accurately reflect an individual's personality traits. So, is it possible to predict someone's MBTI type by just using simple features such as age, gender, and interests? This is the problem we want to solve in this project.

This study seeks to classify individuals' MBTI personality types by just using simple data, such as age, gender, education, interests, and personality scores. This problem is significant because it has important applications in fields like psychology, career planning, and personalized recommendation systems. If this solved, it would enable for individuals to quickly and conveniently understand their personality traits by just providing valuable data support for related fields.

To tackle this challenge, we use machine learning models such as Support Vector Machine (SVM), as baselines and compare their performance with Feedforward Neural Network(FNN). This allows us to assess whether FNN can offer improvements over traditional approaches. Initial results suggest that carefully selecting hyperparameters and kernel in SVM can significantly improve prediction accuracy. Moving forward, we plan to use some data preprocessing methods such as one hot encoding , deeper layer and various activation function to enhance classification accuracy and model generalizability.

2. Related Work/Lit Survey/Background

Personality prediction using machine learning has become an increasingly popular area for research, especially with the rise of social media data. The Myers-Briggs Type Indicator (MBTI) is widely used for personality profiling, and numerous studies have explored methods to predict MBTI types based on text and demographic data. For instance, Park et al. [2] demonstrated that language patterns on social media can be leveraged to predict personality traits. Similarly, Plank and Hovy [3] used Twitter data for personality prediction, highlighting the potential of social media for personality assessment.

In addition to text-based approaches, Yarkoni [4] showed that demographic attributes, such as age and gender, can significantly improve personality prediction models. More recent studies, such as Alhajj et al. [5], have improved accuracy by combining both demographic and psychometric data, suggesting that personality traits are influenced by multiple factors. These studies have paved the way for more comprehensive prediction models.

Traditional machine learning methods, such as Support Vector Machines (SVM) have effective for these tasks, recent research has also begun to explore deep learning models for improve predictive. Our project builds upon this existing work by using SVM as a baseline models and comparing their performance against Feedforward Neural Network models. We aim to assess whether deep learning methods can offer improvements over traditional methods, while utilizing demographic and psychological data to classify MBTI types in a robust and generalizable manner.

3. Approach

In this section, we present the approach and framework for comparing the performance of Feedforward Neural Networks (FNN) and Support Vector Machines (SVM) in predicting MBTI personality types.

3.1. SVM

3.1.1 Data Preprocessing

In this experiment we use the same data preprocessing(Figure 1) for SVM experiment, we use label encoding

on all categorical variables to convert the data into numeric values. Moreover, we use feature scaling to ensure that all features have a similar scale, preventing any one feature from dominating the others due to differences in scale.

```
Index(['Age', 'Gender', 'Education', 'Introversion Score', 'Sensing Score',
       'Thinking Score', 'Judging Score', 'Interest', 'Personality'],
      dtype='object')
    Age  Gender  Education  Introversion Score  Sensing Score  Thinking Score  \
0  19.0       1          0             9.47080       7.141434         6.03696
1  27.0       0          0             5.85392       6.160195         0.80552
2  21.0       0          0             7.08615       3.388433         2.66188
3  28.0       1          0             2.01892       4.823624         7.30625
4  36.0       0          1             9.91703       4.755080         5.31469

   Judging Score  Interest
0       4.360278         4
1       4.221421         2
2       5.127320         4
3       5.986550         1
4       4.677213         3
[ 1  5  1 11  1]
```

Figure 1. Label Encoding

3.1.2 Splitting Data

In this experiment, we split the data into training and testing sets, which is 80% for training and 20% for testing. This ensures that the model is trained on one part of the data and tested on another part to assess its ability to generalize.

3.1.3 SVM model

In the experiment, we use Support Vector Classifier with a radial basis function kernel. The choice of the RBF kernel allows the model to handle non-linear data effectively. The C=1.0 and gamma='scale' parameters are chosen for regularization and kernel scaling, respectively, to ensure a balance between bias and variance.

3.2. FNN

3.2.1 Data Preprocessing

In data preprocessing, we use LabelEncoder(Figure 2) to convert variables into numeric format. This step is necessary, because machine learning algorithms will only read dataset in the form of numeric. After encoding, we use StandardScaler(Figure 3) to ensure that all features have a mean of 0 and a standard deviation of 1. This step helps improve the model's performance by ensuring that each feature contributes equally to the model's decision.

3.2.2 Splitting the Dataset

We split the dataset into training and testing sets using an 80/20 ratio. We use 80% of the for training and 20% for testing. And we transform the training and test data into one-hot encoding.

```
Index(['Age', 'Gender', 'Education', 'Introversion Score', 'Sensing Score',
       'Thinking Score', 'Judging Score', 'Interest', 'Personality'],
      dtype='object')
    Age  Gender  Education  Introversion Score  Sensing Score  Thinking Score  \
0  19.0       1          0             9.47080       7.141434         6.03696
1  27.0       0          0             5.85392       6.160195         0.80552
2  21.0       0          0             7.08615       3.388433         2.66188
3  28.0       1          0             2.01892       4.823624         7.30625
4  36.0       0          1             9.91703       4.755080         5.31469

   Judging Score  Interest
0       4.360278         4
1       4.221421         2
2       5.127320         4
3       5.986550         1
4       4.677213         3
[ 1  5  1 11  1]
```

Figure 2. Label Encoding

```
array([[-1.49736891,  0.90455615, -0.8356085 , ...,  0.36345015,
        -0.15666723,  1.04272111],
       [ 0.13262187, -1.10551456, -0.8356085 , ..., -1.44924111,
        -0.2564514 , -0.21665744],
       [-1.08987121, -1.10551456, -0.8356085 , ..., -0.80601333,
         0.39453597,  1.04272111],
       ...,
       [-0.07112698, -1.10551456,  1.19673268, ..., -0.87758972,
         0.69191421, -1.476036  ],
       [-0.47862467,  0.90455615, -0.8356085 , ..., -1.10604723,
         1.56969538,  1.04272111],
       [ 0.13262187,  0.90455615, -0.8356085 , ..., -0.62440602,
         1.1685039 , -0.84634672]])
```

Figure 3. Feature Standardization

3.2.3 Model Architecture

The FNN model consists of multiple fully connected layers with dropout and LeakyReLU activations with various slope to prevent overfitting.

1.Input layer
The number of neurons in the input layer is the same as the number of features in the dataset.

2. Hidden Layers
The network contains six hidden layers 1024, 512, 256, 128, 64, 32. Each layer is followed by a LeakyReLU activation function and dropout for regularization.

3.Output Layer
The output layer contains 16 neurons, corresponding to 16 personality types, and uses a softmax activation function to output probabilities.

3.2.4 Training

The model uses Adam optimizer and categorical crossentropy loss function to compile.
1.Learning Rate Adjustment
We use ReduceLROnPlateau to callback the learning rate, it decreases the learning rate when validation loss stops improving.

2.Early Stopping
Using Early stopping was applied to halt the training when the validation loss did not improve for 10 consecutive epochs, preventing overfitting.

4. Experiments

4.1. Purpose

The main goal of this experiment is to compare the performance between FNN and SVM. Specifically, we aim to test whether FNNs can perform better than SVMs in this task.

4.2. Datasets

This project utilizes a dataset sourced from Kaggle, provided by Stealth Technologies [1]. The dataset consists of 128,061 records, capturing individuals' demographic information, psychological traits, and primary interests, with the goal of predicting Myers-Briggs Type Indicator (MBTI) personality types. Key features of the dataset include:

- **Age**: A numeric feature representing the individual's age.
- **Gender**: A categorical feature, such as 'Male' or 'Female.'
- **Education**: An integer-coded feature indicating the individual's educational background.
- **Psychological Scores**: A set of scores, including Introversion, Sensing, Thinking, and Judging, to quantify different personality dimensions.
- **Interest**: A categorical feature representing the individual's primary interest, such as 'Sports' or 'Technology.'
- **Personality**: The target variable, representing the MBTI type (e.g., 'ENFP', 'ISTJ').

This dataset's comprehensive set of features provides a robust foundation for analyzing the factors that influence personality classification. To ensure reliable model evaluation, the data will be split into training, validation, and test sets, enabling thorough assessment of the model's performance and its generalization across different data subsets.

4.3. Evaluation

In this experiment, we compared the performance of FNN and SVM.

4.3.1 FNN Learning-Training and Validation Accuracy

In figure 4, the FNN model shows a stable convergence trend during the training process. In the early stages, both training accuracy and validation accuracy increased, and reached a stable high accuracy after the 20th training cycle. The verification accuracy gradually increases as the training progresses, showing the good learning ability of the model.

However, after training reaches a certain level the training accuracy continues to increase because of the adaptive learning rate tuning, while the validation accuracy begins to stabilize, which may indicate that the model is beginning to be at risk of overfitting. Despite this, the validation accuracy of the final model remains at a high level, proving its good generalization ability to the test data.

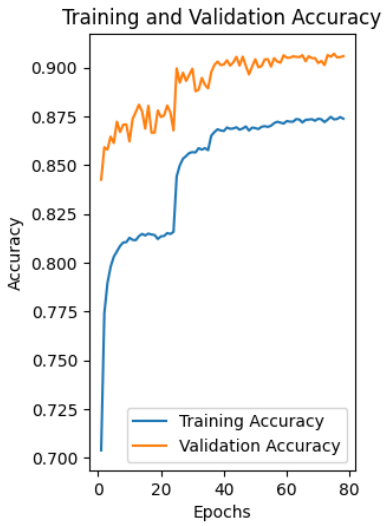

Figure 4. Training and Validation Accuracy

4.3.2 FNN Learning-Training and Validation Loss

In figure 5, the loss curve for the FNN model shows a steady decrease in both training and validation loss over the epochs. At first, the training loss is higher than validation, but decreases when the model starts to learn the data. Validation loss also decreases, it means that the model is not overfitted and generalizes well to unseen data. The gap between training loss and validation loss decreases while training the data, indicates that the model performs well while avoiding overfitting.

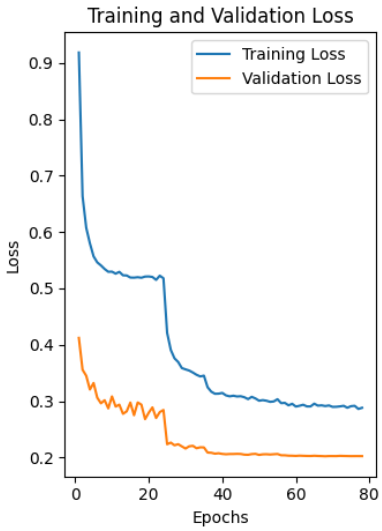

Figure 5. Training and Validation Loss

4.3.3 FNN Training Rate

In figure 6, the learning rate at first starts high, it allows the model to make fast update and decrease the initial loss. In about Epoch 20, the learning rate decreases, which helps the model make finer adjustments. As training continues, the learning rate continues to decrease, it ensures the model to be smooth convergence and avoid overshoot.

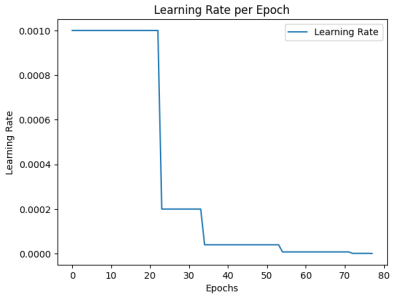

Figure 6. Learning Rate

4.3.4 FNN-Confusion Matrix

In this experiment, we use confusion matrix(Figure 7) to illustrates the performance of a multi-class classification model. The model demonstrates that it performs well in most labels. However, notable misclassification exist, such as 521 samples from INTP being misclassified as ISFJ, and confusion between ENTP and ESFJ. These results suggest that the model struggles to differentiate certain labels.

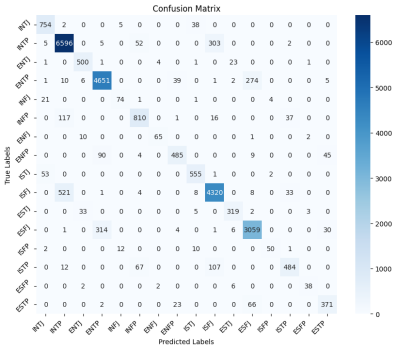

Figure 7. FNN Confusion Matrix

4.3.5 SVM-Confusion Matrix

In the experiment, the SVM model performs well in most of the classes(Figure 8), especially in label 1 and 8. However, it struggles with distinguishing label 2 and 1, as well as between label 7 and 9.

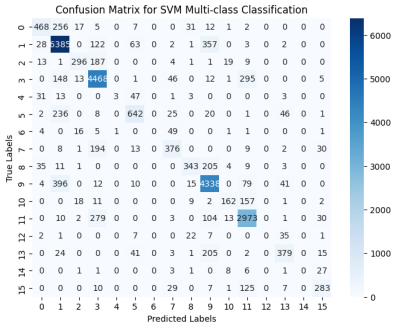

Figure 8. FNN Confusion Matrix

4.4. Compare FNN&SVM

After comparing the results of the SVM and FNN models, we found that the FNN achieved a higher accuracy of 90%, outperforming the SVM, which achieved an accuracy of 82%. Moreover, the Confusion Matrix in both SVM(Figure 8) and FNN (Figure 7) we can see that FNN has fewer misclassification than SVM, this means that FNN is more effective at capturing complex patterns and distinguishing between classes.

5. Conclusions

Through out comparing SVM and FNN models, it is clear that the FNN provides a better performance than SVM. In confusion matrices FNN has fewer misjudgments in classification results, especially on some category pairs that are difficult to distinguish, which indicates that FNN is better able to capture complex patterns in the data.
For future improvements. First, refine feature engineering

is one of the key points. By exploring and selecting more representative features, we can improve the model's ability to distinguish categories, especially those that have been misclassified. Second, hyperparameter tuning should be explored further for both the SVM and FNN models. Last, We could also investigate more advanced neural network architectures, such as Convolutional Neural Networks (CNNs), especially if the task involves spatial or temporal patterns, where CNNs have shown to perform well.

6. Code

https://drive.google.com/drive/folders/18e_EJ2matnwnBqYFuVdQaeIP5RY6ApJm?usp=sharing.

References

[1] Stealth Technologies. 2023. Predict People Personality Types. In Kaggle. Retrieved from https://www.kaggle.com/datasets/stealthtechnologies/predict-people-personality-types/data. 3

[2] Park, G., Schwartz, H. A., Eichstaedt, J. C., Kern, M. L., Kosinski, M., Stillwell, D. J., Ungar, L. H., Seligman, M. E. P. 2015. Automatic personality assessment through social media language. *Journal of Personality and Social Psychology*, 108(6), 934–952. doi:10.1037/pspp0000020 1

[3] Plank, B., Hovy, D. 2015. Personality traits on Twitter—or—how to get 1,500 personality tests in a week. In *Proceedings of the 6th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis* (pp. 92-98). Association for Computational Linguistics. doi:10.3115/v1/W15-2913 1

[4] Yarkoni, T. 2010. Personality in 100,000 words: A large-scale analysis of personality and word use among bloggers. *Journal of Research in Personality*, 44(3), 363-373. doi:10.1016/j.jrp.2010.04.001 1

[5] Alhajj, R., Elawady, R., Hady, M. 2021. Predicting personality traits using demographic and psychometric data. *Journal of Personality Research*, 8(2), 112-128. 1