Prediction of Freezing of Gait Status From Fall Descriptions

Sean Jordan
Department of Biomedical Informatics / Emory University
Sean.Jordan@emory.edu

Abstract

Freezing of Gait (FoG) is a devastating complication of Parkinson's Disease (PD) that increases frequency of falls and leads to a very poor quality of life for those inflicted, and as such it is important that we determine who is likely to develop FoG in their lifetime. A one-year prospective observational study was conducted on patients previously diagnosed with PD in which they were queried for any information (such as a description of what happened) relating to falls they experienced during the one-year period. Through the use of advanced Natural Language Processing (NLP) and feature engineering techniques, this experiment aimed to predict whether a patient suffered from FoG from the descriptive account of their falls. By testing various combinations of five different classifiers and four different feature engineering techniques, we found the best classifier to be an ensemble voting classifier, which yielded a micro-averaged F1 score of 0.8551. This high F1 score indicates that it is possible to predict FoG status from a descriptive account of PD patients' falls.

1 Introduction

One notable symptom that affects Parkinson's Disease (PD) patients, particularly in the latter stages of the disease, is Freezing of Gait (FoG). FoG is defined as a brief absence/reduction of forward progression in the lower body despite the person's full intention to walk (Rahimpour et al., 2021). It is not hard to intuitively understand why this phenomenon can be dangerous and life-threatening in PD patients. Balance, especially in older-age patients (when PD is more likely to strike), is notably diminished and often causes a lot of falls just on its own. With the added effect of unwanted pauses in lower body movement, it can understandably cause a lot of balance issues and unintended falls. These falls can not only lead to a much poorer quality of life, but can be life-threatening as well. Because of this, FoG has played a key part in recent research administered in the context of PD, and one notable aspect of this research involves predicting whether or not a patient has or will have FoG.

One particular research area that has gained lots of traction in academia recently is Natural Language Processing (NLP), which involves the automated extraction and manipulation of speech from textual data. One notable aspect of NLP is that it is not limited to a particular domain of study - countless review articles such as the one authored by Khurana et al. have shown how NLP can be applied to other areas of study, including biomedical and healthcare domains (2023). The study of NLP techniques becomes particularly interesting in the context of FoG and PD, because (since FoG is random and often not observable by professionals) one of the main ways physicians diagnose FoG is through the personal testimony of the patient or a relative when they fall, which often comes in the form of written text that NLP algorithms are trained to analyze. Thus, the question arises as to whether or not we can use this description of the circumstances surrounding a patient's fall to predict whether or not they have or will have FoG.

2 Methods

A one-year prospective observational study was conducted on adults previously diagnosed with PD who had expressed written and informed consent as set forth by the Emory University Institutional Review Board. After an initial study visit in which participants were assessed with a detailed cognitive and motor battery of indicators of potential fall risk, participants were tracked for a nominal one-year period. During this observation period, the study participants were queried about the presence and circumstances of any falls at monthly intervals using mail, email, phone, or text, at the discretion of the participant. Details of subsequent falls were recorded by participants and verified by

study staff. From this study, a dataset was produced that contained a variety of variables, ranging from demographic information to clinical scales of PD progression. Of these, two variables are the most significant - the description of the circumstances surrounding the fall (located in the *fall_description* column), and whether or not the patient was diagnosed with FoG in the baseline study (located in the *fog_q_class* column).

The final dataset resulting from the study contained 370 rows of information. For machine learning purposes, an 80/20 train-test split was performed, which ended up putting 299 samples in the training data and the remaining 71 samples in the test dataset. After loading the dataset, it was discovered that three samples in the training dataset and two additional samples in the testing dataset did not have descriptions of any falls. Since the model we are trying to build relies on the fall description, we opted to remove those rows entirely, leaving 296 training samples and 69 testing samples.

In our experiment, we tested six different classifiers and five different feature sets, and aimed to find the combination of those two categories that resulted in the highest micro-averaged F1 score for predicting FoG. The feature engineering methods tested, along with the descriptions of how they were used in the experiment, are listed in Table 1. The classifiers tested on these engineered features (and their `scikit-learn` class names) are below:

- Gaussian Naive Bayes (`GaussianNB`)
- Support Vector Machine (`SVC`)
- K-Nearest Neighbors (`KNeighborsClassifier`)
- Logistic Regression (`LogisticRegression`)
- Multi-Layer Perceptron (`MLPClassifier`)
- Random Forest (`RandomForestClassifier`)
- Ensemble Soft Voting (`VotingClassifier`)

In order to tune the hyperparameters of the six individual classifiers tested, a grid search was performed with five-fold cross validation (CV). One important thing to note is that if we did the TF-IDF and N-gram vectorizing on the entire training dataset before splitting the data into CV folds, then the features subjected to the validation fold will have already been seen and trained on by the model, causing target leakage and an overestimate of the model's prediction ability. To solve this problem, we used `scikit-learn`'s *pipeline* functionality, which works seamlessly with their cross-validation hyperparameter search such that the fitting of the TF-IDF and N-gram models are only done on the train data and not the validation data for each CV fold. This involved creating a custom "classifier" called `FreezingOfGaitDatasetBuilder` that would come right before the actual classifier in the pipeline.

Feature	Description
N-grams	Continuous sequences of n words in document
Number of Characters	Number of characters in a given text
Average Length of Words	Average length of words in a given text
TF-IDF	Term Frequency - Inverse Document Frequency
Part-of-Speech (POS) Tagging	Number of each type of part of speech in text
Word2Vec	Vector embedding of each post

Table 1: List of engineered features and how they are used in the classifier models

One decision we had to make in the feature engineering process was whether or not to preprocess the data (i.e. remove stopwords and unnecessary punctuation). For the features such as number of characters, average length of words, and part-of-speech tagging, preprocessing is never done because the result inherently lies on all of the text being included in the analysis. For TF-IDF and N-gram vectorization though, pre-processing would help vastly decrease the dimensionality of the dataset. This said however, pre-processing does not always guarantee that the models will improve in their performance - in fact, it could even make the models worse, which when dealing with descriptions of peoples' falls (that have lots of meaning in stopwords), could very well be the case. Therefore, we gave the user a parameter option in the `FreezingOfGaitDatasetBuilder` initialization to turn preprocessing on/off depending on their resulting performance.

Finally, the last modification we made to the data was standardization. After the dataset was built using the custom classifier, all of the features were transformed such that the mean for each feature was 0, and the standard deviation for each feature was 1.

After building the required dataset and running it through each of the six models we tested, we aggregated the micro-averaged F1 scores and chose the model that had the highest value. Using this best classifier, we generated a training-set size versus

performance graph to estimate how much data we actually need to maintain a reasonable performance with our model. We also performed a feature ablation study with the best classifier by removing each feature set on its own and re-running the classifier to see how much that feature impacted the overall performance of the model.

3 Results

Surprisingly, the baseline Naive Bayes Model was not too bad, with an accuracy and micro-F1 score of 0.7971. Three of the other models tested (K-Nearest Neighbors, Multi-Layer Perceptron, and Random Forest) did not compare to the Naive Bayes model and gave much worse performance results. Two models though - Support Vector Machines and Logistic Regression - performed slightly better than the baseline Naive Bayes model. None of these individual models, however, were able to compare to the ensemble method that combined all of them, which yielded an accuracy and micro-F1 score of 0.8551 on the held-out test data.

Additionally, when we compared the performance of the models *with* data pre-processing versus *without* data pre-processing, we found that every model actually performed significantly worse when we pre-processed the data (i.e. removed stop words and punctuation). The results of this pre-processing analysis are in Table 3.

The training set size versus performance graph is shown below in Figure 1. We see a logarithmic-like curve for the micro-averaged F1 score that rises up quickly at the start but gradually evens out as the training set size increases.

Table 4 shows the results of the feature ablation study, in which each individual feature set was removed from the model to see how the performance fared. We can see that removing the vectorized TF-IDF feature set yielded the highest decrease in model accuracy (down to 0.7826), whereas removing the Word2Vec feature set yielded the *lowest* decrease in model accuracy (down to 0.8261).

4 Discussion

The comparison of the performance of the models with and without preprocessing is interesting, because usually it goes the other way around, where the model performs much better after preprocessing. This could show that a lot of the predictive value that indicates FoG status lies in stopwords and not in the other text, which is an important note in it of itself.

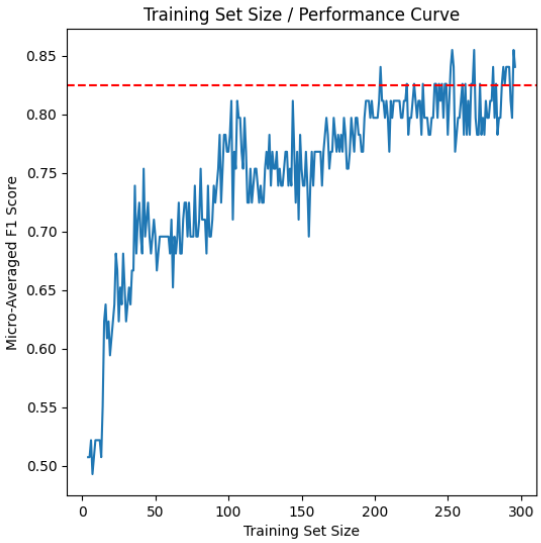

Figure 1: The training data becomes saturated at around 200 samples and will no longer yield a higher micro-F1 score if more training samples are provided to the model. The area that the model levels out at (red line) is at an micro-F1 score of 0.825.

Surprisingly, the individual classifiers by themselves had a very high variance in their performance scores. In fact, the Naive Bayes classifier, which was supposed to be the baseline in this experiment, had a better performance than both NKK, MLP, *and* Random Forest Classifiers. What is more interesting is that including all six models - even the ones with poor performance like K-Nearest Neighbors - led to a much higher performance when they were aggregated together in a soft-voting ensemble classifier. This shows that while these models' performance is subpar, they still give useful information to the classification of FoG.

As expected, the training set size versus performance graph follows a logarithmic curve. The fact that the graph evens out pretty nicely at the end indicates that we *do* actually have enough training data for our model (and that adding more data won't really make it better). In other words, the training set of our model is saturated and satisfies the information requirements for the models being trained on it. In fact, we could decrease our training-set size to just 200 samples, and the model would still theoretically work just the same when it comes to micro-averaged F1 score.

Model	Hyperparameters	Accuracy	Macro F1	Micro F1
Gaussian Naive Bayes	NA	0.7971	0.7936	0.7971
Support Vector Machine	C = 10, kernel = "sigmoid", probability = True	0.8116	0.8116	0.8116
K-Nearest Neighbors	n_neighbors = 3, weights = "uniform"	0.5072	0.3365	0.5072
Logistic Regression	C = 0.01, max_iter = 5000, penalty = 'l2', solver = 'saga'	0.8261	0.8231	0.8261
Multi-Layer Perceptron	alpha = 0.0001, learning_rate = "adaptive", learning_rate_init = 0.001, max_iter = 4000	0.8116	0.8102	0.8116
Random Forest	criterion = "entropy", max_depth = 7, max_features = "sqrt", n_estimators = 150	0.6957	0.6862	0.6957
Ensemble Soft Voting	NA	0.8551	0.8543	0.8551

Table 2: The model with the best micro-averaged F1 score is the Ensemble Soft Voting Classifier, highlighted in yellow.

Model	Micro-Averaged F1 Score With Preprocessing	Micro-Averaged F1 Score Without Preprocessing
Gaussian Naive Bayes	0.7391	0.7971
Support Vector Machine	0.7391	0.8116
K-Nearest Neighbors	0.5072	0.5072
Logistic Regression	0.7536	0.8261
Multi-Layer Perceptron	0.6957	0.7681
Random Forest	0.6667	0.6957
Ensemble Soft Voting	0.7971	0.8551

Table 3: All seven models, including the ensemble voting classifier, performed significantly better when there was no preprocessing done (highlighted in yellow).

Feature Removed	Micro-Averaged F1 Score
N-Grams	0.7971
Number of Characters	0.8260
Average Length of Words	0.8116
TF-IDF	0.7826
POS Tagging	0.8116
Word2Vec	0.8261

Table 4: The TF-IDF feature contributed the most to the predictive capabilities of the classifier, while Word2Vec contributed the least.

5 Conclusion

The classifier that we found to best predict FoG status from the textual description of patients' falls was an ensemble soft voting classifier that aggregated the performance measures from Gaussian Naive Bayes, Support Vector Machine, K-Nearest Neighbors, Logistic Regression, Multi-Layer Perceptron, and Random Forest Classifiers. The reported micro-averaged F1 score was 0.8551 showing that it is highly effective at predicting the correct FoG status. We also proved that our model has enough training data in its repertoire by looking at the training set versus performance curve, and that the feature engineering method that does the best job by itself of predicting FoG status are TF-IDF vectors. In all, this experiment proves that a textual description of the circumstances surrounding a PD patients' fall can in fact be used to predict FoG status, which opens up a variety of doors in the field of NLP. Future experiments on this topic could explore other feature engineering techniques or other classification models that could potentially drive the performance of FoG classification even higher.

References

D. Khurana, A. Koli, K. Khatter, and S. Singh. 2023. Natural language processing: state of the art, current trends and challenges. *Multimed Tools Appl*, 82(3):3713–3744.

S. Rahimpour, W. Gaztanaga, A. P. Yadav, S. J. Chang, M. O. Krucoff, I. Cajigas, D. A. Turner, and D. D. Wang. 2021. Freezing of Gait in Parkinson's Disease: Invasive and Noninvasive Neuromodulation. *Neuromodulation*, 24(5):829–842.

Github Link:
https://github.com/sjordan2/BMI550-Assignment2