$$r_t = \sigma(W_r[h_{t-1}, x_t]) \quad (1)$$

$$z_t = \sigma(W_z[h_{t-1}, x_t]) \quad (2)$$

$$\bar{h}_t = tanh(W_{\bar{h}} \times [r_t \times h_{t-1}, x_t]) \quad (3)$$

$$h_t = (1 - z_t) \times h_{t-1} + z_t \times \bar{h}_t \quad (4)$$

$$y_r = \sigma(W_\sigma \times h_t) \quad (5)$$

Where σ represents sigmod function, z_t is the update gate of the unit, sigmod function converges the value of the update gate to 0 or 1, i.e., whether the value of the previous step is remembered or discarded. r_t is the reset gate, the smaller r_t, the more information about the previous state needs to be ignored, W is the weight value, h_t and $\bar{h}_t$ are the output and temporary hidden states in the module.

The GRU model has a lower computational cost with faster training, so the model is extensively used in various fields of deep learning. The structure of a single GRU module is shown in Fig. 2.

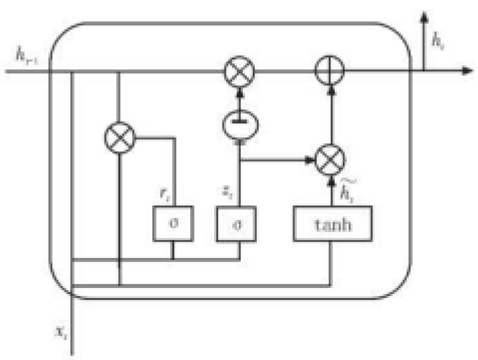


Figure 2. Structure of a single GRU model.

The GRU neural network in this work had a total of 6 layers and 1700 learnable properties. Table III illustrated the structure of the GRU neural network.

Table I
THE STRUCTURE OF GRU NEURAL NETWORKS

Name	Activations	Learnable Properties
Sequence Input	$88(C) \times 1(B) \times 1(T)$	
GRU	$6(C) \times 1(B)$	InputWeights 18×88 RecurrentWeights 18×6 Bias 18×1
ReLU	$6(C) \times 1(B)$	
Fully Connected	$2(C) \times 1(B)$	Weights 2×6 Bias 2×1
Softmax	$2(C) \times 1(B)$	
Classification Output	$2(C) \times 1(B)$	

E. Public Voice Datasets Used

This paper used public datasets [11] collected from 188 PD patients (107 men, 81 women) aged 33-87 at Istanbul University. The control group includes 64 healthy individuals (23 men, 41 women) aged 41-82. Participants were asked to sustainably pronounce the vowel /a/ while a microphone set at 44.1 KHz recorded their voice three times.

DementiaBank [12] is a resource that collects voice, video, and text data from older adults and patients with AD. It contains two groups of participants; the elderly group includes 60 healthy older adults from New York City who ranged in age from 60 to 91 years, while the AD group includes 64 patients from Pittsburgh who ranged in age from 60 to 95 years. Each participant was asked to answer a series of questions. Data were collected using a specialized recording device, with recorded voice data at a sampling rate of 44.1 kHz.

Visualization of voice can help to extract feature information. The voice waveform and spectrum of AD are shown in Fig. 3.

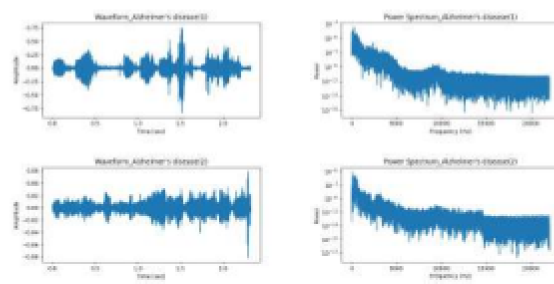


Figure 3. Alzheimer data voice waveforms and spectrograms.

F.Training and Testing Process

After noise removal and signal segmentation of all the voice data in the dataset, we extracted 88 voice signal features for each voice window signal, and then created a neural network for training and learning. In this paper, a 6-layer GRU model was adopted. It's model structure is shown in Fig. 4.

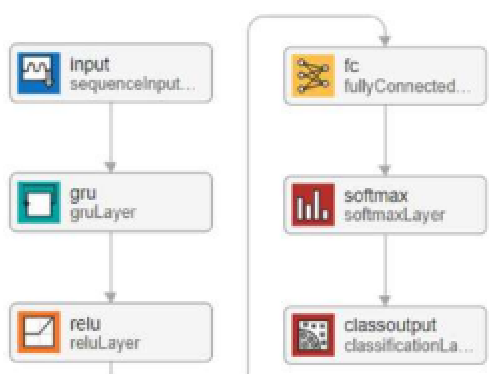


Figure 4. Structure of 6-layer GRU model.

To avoid overfitting, we added a Relu layer after the GRU layer and output the probability of class labels by defining a Softmax layer to vectorize the labels ontehot before calculating the correctness. To accelerate the model convergence, the training took a batch gradient descent approach for weight update, and each batch contained 64 features. A flow chart of the overall model structure of the experiment is shown in Fig. 5.

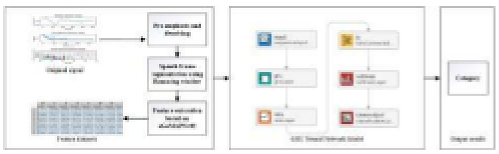


Figure 5. Flow chart of the overall model structure of the experiment

G. Deploying the Model to the Thingspeak Platform

Thingspeak is an open source IoT application platform that allows users to conveniently collect, process and analyze data from IoT devices. The platform provides a way for developers and manufacturers to collect, store, analyze and visualize data from the center of IoT devices and use that data for real-time decision-making and operations. To upload the sensor data from the mobile phone to the Thingspeak IoT platform and read the results of the data analysis using an application developed by ourselves, we followed the following steps:

1) Registering a Thingspeak account and creating new channel.
2) Getting our channel write/read API Key, which we can find in our Thingspeak account.
3) Adding network authority and sensor permission in our application.
4) Adding the Thingspeak API library, we can get the source code of the library from the Thingspeak website
5) Implementing the code to upload data in our application, the code should use HTTP protocol to upload our sensor data to our Thingspeak channel, providing the channel write API Key to authenticate our identity
6) After uploaded data, we can analyze the data using Thingspeak's analytics tool. Once we have uploaded the data, we could use Thingspeak's analytics tool to analyze the data. To get the result of data analysis by using HTTP GET request.
7) Implementing the code to read the analysis results in our application. We need to get data analysis results using HTTP GET request and read API key to parse the results into JSON format so that we can process and display the data in our application.

In summary, to upload the sensor data from the phone to the Thingspeak IoT platform and read the data analysis results, we need to register an account and create a new channel, get the channel write/read API Key, add network permissions and sensor permissions, add the Thingspeak API library, implement the code to upload the data, use the platform's analysis tool to analyze the data, implement the code to read the analysis results, and parse the results into JSON format to process as well as display the data in the application.

Deployment of the GRU model to the Thingspeak IoT platform for data analysis Data analysis on Thingspeak using the GRU model involves the following steps:

1) Creating a new channel on Thingspeak to store the data to be analyzed. We can use Thingspeak's REST API or MQTT API to add the sensor data to the channel.
2) Training a GRU model on our local computer and exporting the model to a format that can be used on.
3) Uploading the exported KNN model to the Thingspeak platform. We can use Thingspeak's REST API or MQTT API to upload the model to the channel.
4) Once the model is uploaded successfully, we can use Thingspeak's MATLAB analysis toolbox or matlab scripts to load the model and classify the uploaded data. In MATLAB, we can read the uploaded data using the thingSpeakRead function, load the GRU model using the load function, and classify the data using the predict function.
5) Displaying the classification results on the user interface of Thingspeak or sending the results to our cell phone as well as to an email for easy viewing of the identification results.

IV. EXPERIMENTS AND RESULTS

A. Experimental Setup

In this paper, the feature datasets were divided into training datasets and test datasets in the ratio of 9:1. The training datasets were trained and validated using the 5-fold cross-validation method, which was repeated five times. The test datasets were then used to test the final results. And We evaluated the experiment using the confusion matrix [13]. The Table II showed the GRU neural network model hyperparameter setting table in this experiment.

B. Experiment Results and Evaluation

The Fig. 6 showed the process of training the GRU model in 1000 epochs based on the Parkinson's public voice datasets. As seen in the Fig. 6, the GRU neural network model based on the Parkinson's public voice dataset can converge substantially in a short time. The model uses stochastic gradient descent and variable learning rate in solving the minimization loss function, so there was

Table II
GRU NEURAL NETWORK MODEL HYPERPARAMETER SETTING

Number	Parameter Name	Parameter Value
1	Mini Batch Size	64
2	Max Epochs	1000
3	Initial Learn Rate	0.01
4	Learn Rate Drop Factor	0.1
5	Learn Rate Drop Period	700
6	Shuffle	every-epoch
7	optimization	adam

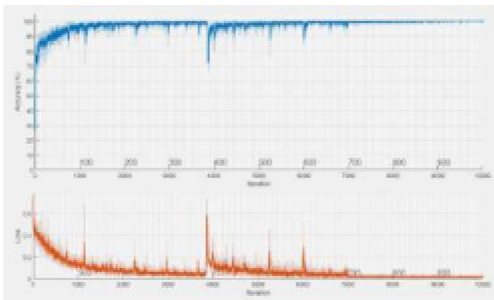
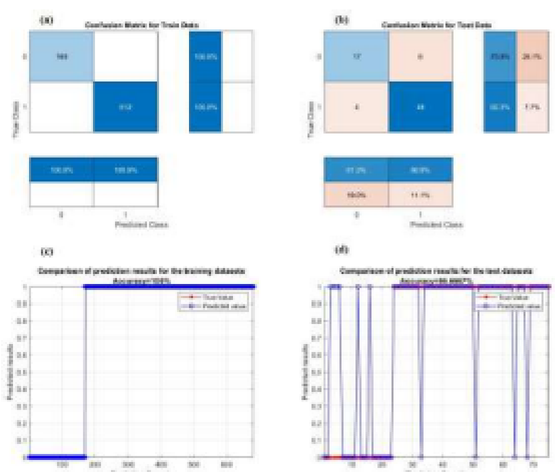


Figure 6. The process of training the GRU model based on Parkinson's datasets in 1000 epochs.

some jitter in the convergence process of the model, but the general trend of the model accuracy was improved, the loss function of the model corresponded to a decreasing trend. The final training accuracy of the model reached 100%.



c

As can be seen from Fig. 7, the accuracy of the model on the test set was 86.66 %, while the accuracy on the training set was much better than the accuracy on the test set, the model may have been overfitted. The overfitting phenomenon may arise because of the small amount of data in the public voice dataset of Parkinson's, coupled with the uneven distribution of samples in this public dataset, so the model's performance was degraded.

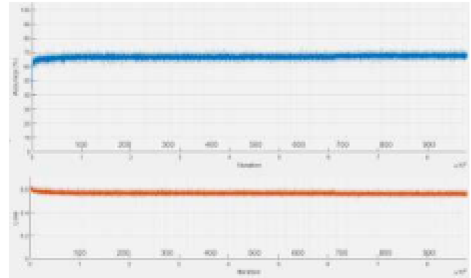


Figure 8. The process of training the GRU model based on Alzheimer's datasets in 1000 epochs.

As can be seen in Fig. 8, the model converged after the Alzheimer's voice training dataset was fed into the model and entered 2000 training cycles. The Fig. 9 showed a comparison of the prediction results and the confusion matrix of the training and testing datasets for AD.

Figure 9. a - confusion matrix of testing datasets; b - confusion matrix of training datasets.

Table III showed the experimental results of PD recognition and AD using GRU based on the test datasets.

Table III
THE EXPERIMENTAL RESULTS OF PD RECOGNITION AND AD USING GRU BASED ON THE TEST DATASETS

Public Datasets	Average Precision	Average Sensitivity	Average F1 score	Test Accuracy
Parkinson's	84.95 %	83.10 %	84.01 %	86.66 %
Alzheimer's	67.60 %	67.50 %	67.55 %	68.27 %

In summary, the accuracy of the GRU-based PD model could reach 86.67 % on the test dataset and 100 % on the training dataset. On the testing datasets, the average precision was 84.95 %, the average sensitivity was 83.10 %, and the average F1 score was 84.01 %. This experimental result showed that the recognition of PD using GRU algorithm based on freezing of gait data was effective.

However, the test results of the model on Alzheimer's data were not satisfactory, which may be due to the fact that Alzheimer's data were more complex and harder to find feature points compared to Parkinson's data, after which we will try new models or improve the model in a