Sharif University of Technology
Electrical Engineering School

Computational Intelligence Final Project

BCI CLASSIFICATION TASK USING MLP, RBF & PSO

Armin Panjehpour
arminpp1379@gmail.com

Supervisor(s): Dr. Sepideh Hajipour
Sharif University, Tehran, Iran

24/07/2022

Part.1 - Implemented Features

The list of features implemented can be found below:

Features List	
Feature Name	Implemented Function Name
Variance	Var_Feature
Amplitude Histogram	amp_hist_Feature
AR coefficients	AR_Coeffs
Form Facto	FF_Feature
Covariance between Channels	cov_Feature
Kurtosis	kurtosis_Feature
Skewness	skewness_Feature
Lyapunov Exponent	lyapunov_Feature
Entropy	entropy_Feature
Correlation Dimension	cordir_Feature
Maximum Frequency	max_freq_Feature
Mean Frequency	mean_freq_Feature
Median Frequency	med_freq_Feature
Energy band for 5 ranges	band_energy_Feature

Energy bands are calculate for 5 bands as you can see below:

Energy Bands	
Band Name	Band Range
All Range	All Range
Delta	[0.5 4] Hz
Theta	[4 8] Hz
Alpha	[8 12] Hz
Beta	[12 35] Hz

From all of these features, 12 features that I think are better were selected and calculated for all train and test data and normalized and all of them normalized with respect to training data. So There are a total of 12×30 (channels), 360 features that we have to select using Fisher score.

The selected features are:
1. Kurtosis
2. Skewness
3. Lyapunov Exponent
4. Entropy
5. Correlation Dimension
6. Mean Frequency
7. Median Frequency
8-12. Energy band for 5 ranges

Part.2 - 1D Fisher Score For Each of 360 Features

Using the 1D Fisher score, we calculate a score for each feature and keep the ones with a score of more than 0.01 which leaves us with 179 features.

$$J = \frac{|S_b|}{|S_w|} = \frac{|\mu_0-\mu_1|^2+|\mu_0-\mu_2|^2}{\sigma_1^2+\sigma_2^2}$$

Part.3 - nD Fisher Score For Selected Features

Now, to select the best combinations of features between these 179 features, we select n (here 12) features for lots of times (here 10000 times) and calculate J for them, and then select the combination of features that have the maximum score. The scores are calculated with nD Fisher Score as below:

$$S_1 = \frac{1}{N_1}\sum_{i\in C_1}(\mathbf{x}_i-\boldsymbol{\mu}_1)(\mathbf{x}_i-\boldsymbol{\mu}_1)^T \qquad S_W = S_1 + S_2$$

$$S_2 = \frac{1}{N_2}\sum_{i\in C_2}(\mathbf{x}_i-\boldsymbol{\mu}_2)(\mathbf{x}_i-\boldsymbol{\mu}_2)^T$$

$$S_b = \sum_{i=1}^{2}(\mu_i-\mu_0)(\mu_i-\mu_0)^T$$

$$J = \frac{trace(S_b)}{trace(S_W)}$$

Figure 1: nD Fisher Score

There are lot of good combination with high J score and each time we select features, the features could change. For Instance, here's the result for one of my selections with a J score of 0.081:

1. "kurt_feature for channel 14"
2. "energy_feature for band Delta channel 28"
3. "meanfreq_feature for channel 18"
4. "kurt_feature for channel 2"
5. "lyapunov_feature for channel 10"
6. "lyapunov_feature for channel 12"
7. "meanfreq_feature for channel 1"
8. "meanfreq_feature for channel 20"
9. "skew_feature for channel 2"
10. "meanfreq_feature for channel 27"
11. "entropy_feature for channel 10"
12. "corrDimension_feature for channel 6"

Keeping these features, we then have a 120×12 and 40×12 matrices for training and testing, respectively.

Part.4 - Train a MLP Network

My network has two hidden layers with size of 5 and 4, input layer having neurons equal to number of features and an output layer. Using 5-fold cross validation, each time a part of data is separated and others are used for training. Averaging over 100 runs, the accuracy of training and validation can be seen below:

Average Training Accuracy: 80% - Standard Deviation of Training Accuracy: 5%
Average Validation Accuracy: 73 % - Standard Deviation of Validation Accuracy: 4%

Test labels are estimated using the above network and are attached in a .mat file named "predicted_y_test_MLP".

Part.5 - Train a RBF Network

Doing the same things just with RBF network with 5 neurons and a spread size of 0.4, We get these accuracies:

Average Training Accuracy: 70 % - Standard Deviation of Training Accuracy: 0.5%
Average Validation Accuracy: 68 % - Standard Deviation of Validation Accuracy: 0.8%

Test labels are estimated using the above network and are attached in a .mat file named "predicted_y_test_rbf".

Part.6 - Consistency Between MLP and RBF

The predicted test labels of these two networks, have 80 to 85 % consistency which is a high consistency according to the validation accuracies and indicate that the test labels are predicated nearly correct.

Part.7 - Particle Swarm Optimization for Feature Selection

PSO_feature_selection function is implemented for feature selection according to the PSO algorithm as below

```
Algorithm 13.10 (Particle-Swarm-Optimization)
function pso (m: int, a, b, c: real) : array of real;
begin                                      (* m: number of particles *)
  t ← 0; q ← −∞;                           (* a, b, c: update parameters *)
  for i ∈ {1, ..., m} do begin             (* initialize the particles *)
    v_i ← 0;                               (* initialize velocity and position *)
    x_i ← choose a random point of Ω = ℝ^n;
    x_i^(local) ← x_i;                     (* initialize the local memory *)
    if f(x_i) ≥ q then begin x^(global) ← x_i; q ← f(x_i); end
  end                                      (* compute initial global memory *)
  repeat                                   (* update the swarm *)
    t ← t + 1;                             (* count the update step *)
    for i ∈ {1, ..., m} do begin           (* update the local and global memory *)
      if f(x_i) ≥ f(x_i^(local))  then x_i^(local) ← x_i; end
      if f(x_i) ≥ f(x^(global)) then x^(global) ← x_i; end
    end
    for i ∈ {1, ..., m} do begin           (* update the particles *)
      β_1 ← sample random number uniformly from [0, a);
      β_2 ← sample random number uniformly from [0, a);
      α ← b/t^c;                           (* example of a time dependent α *)
      v_i(t + 1) ← α · v_i(t) + β_1(x_i^(local)(t) − x_i(t)) + β_2(x^(global)(t) − x_i(t));
      x_i(t + 1) ← x_i(t) + v_i(t);        (* update velocity and position *)
    end
  until termination criterion is fulfilled;
  return x^(global);                       (* return the best point found *)
end
```

Figure 2: PSO Algorithm

Having 20 particles, which each of them are a vector of length of the number of features that we want (here 12) and the values are ranging from 1 to 179 which is the feature number (I used the 179 features after fisher score threshold for a smaller area of search and better result). The selected features will give us a J score of 0.09 which is a little better than the previous feature selection method.

Training MLP and RBF networks with these selected features, the accuracies will be (MLP has 5 neurons in two hidden layers, RBF has 20 neurons with a spread size of 0.6):

- MLP:

Average Training Accuracy: 82 % - Standard Deviation of Training Accuracy: 3%
Average Validation Accuracy: 76 % - Standard Deviation of Validation Accuracy: 4%

- RBF:

Average Training Accuracy: 83 % - Standard Deviation of Training Accuracy: 0%
Average Validation Accuracy: 73 % - Standard Deviation of Validation Accuracy: 0%

The test labels are attached with a "pso" in the end of their names in zip file.

Part.8 - Consistency Between MLP and RBF using PSO

The predicted test labels of these two networks, have 82 to 85 % consistency which is better than fisher score feature selection and is a high consistency.

Part.9 - Consistency Between Fisher and PSO Feature Selection

There is also 85 % consistency between the test labels of the two feature selection methods which is indicating that we have predicated test labels with a high accuracy and consistency.

Part.10 - Comparing

According to the results, MLP is better than RBF in both feature selection methods and also, using PSO (or probabily other genetic algorithms), we will have better features and higher accuracies.