

# Train Topographic Map Network

Michael Caiola ([Michael.Caiola@fda.hhs.gov](mailto:Michael.Caiola@fda.hhs.gov)) and Meijun Ye ([Meijun.Ye@fda.hhs.gov](mailto:Meijun.Ye@fda.hhs.gov))

Live script used to train topographic map network

```
tic
path = pwd;
cd ../../eeglab/ % replace with EEGLab path
eeglab
```

```
Some menus items hidden. Use Preference menu to show them all.
eeglab: options file is C:\Users\nonun\eeg_options.m
Retrieving plugin versions from server...
Retrieving download statistics...
EEGLAB: adding "Biosig" to the path; subfolders (if any) might be missing from the path
EEGLAB: adding "ICLabel" v1.3 (see >> help eegplugin_iclabel) - new version 1.4 available
EEGLAB: adding "clean_rawdata" v2.6 (see >> help eegplugin_clean_rawdata) - new version 2.7 available
EEGLAB: adding "dipfit" v4.3 (see >> help eegplugin_dipfit)
EEGLAB: adding "firfilt" v2.4 (see >> help eegplugin_firfilt) - new version 2.6 available
EEGLAB: adding "xdfimport" v1.18 (see >> help eegplugin_xdfimport)
Warning:
A newer revision of EEGLAB (v2022.1) is available HERE.
```

See Release notes for more information  
You may disable this message in the File > Preferences menu.

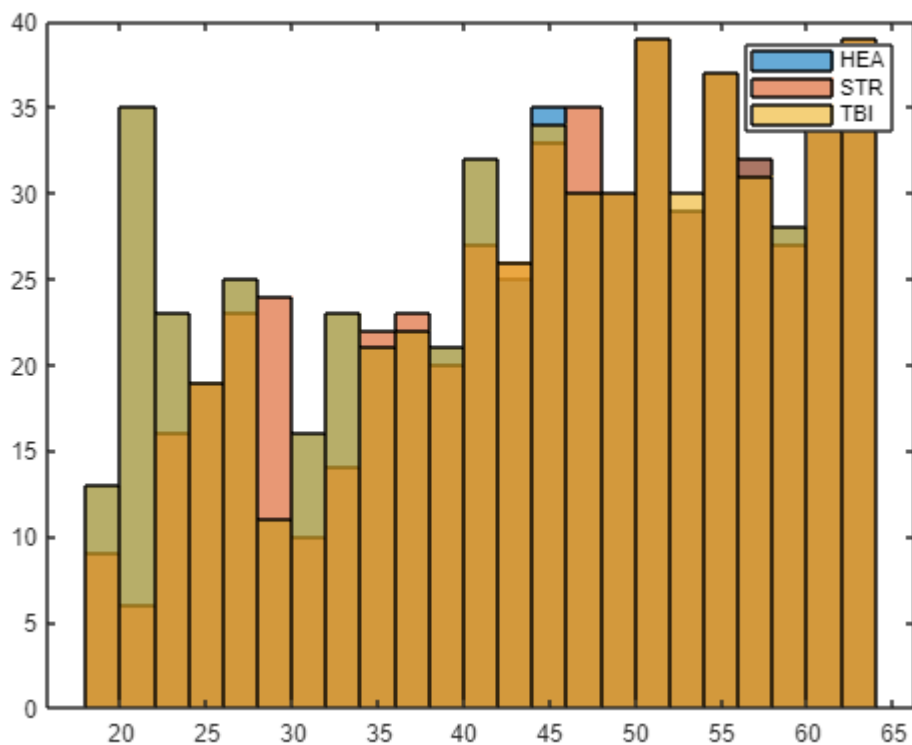
```
close
cd (pwd)
load("chlocs2.mat")
%Make TopoDatastore
fol = "D:\3minSept22";
ds_all = TopoDatastore(fol,[],channel_locations);
toc
```

Elapsed time is 256.766027 seconds.

Used Matched Age and Gender subjects/sessions of HEA and STR

```
[HEA,STR,TBI] = MatchSubjects();
```

```
# HEA subjects/sessions: 595/629
# STR subjects/sessions: 527/585
# TBI subjects/sessions: 552/629
```



```

s = split(HEA.Location, '\');
s = string(s(:,9));
s = split(s, '.');
HEA = s(:,1);
s = split(STR.Location, '\');
s = string(s(:,9));
s = split(s, '.');
STR = s(:,1);
s = split(TBI.Location, '\');
s = string(s(:,9));
s = split(s, '.');
TBI = s(:,1);

```

Check for overlaps

```

f = string(ds_all.Datastore.Files);
in_HEA = [];
in_STR = [];
in_TBI = [];
for i = 1:length(f)
    for j = 1:length(HEA)
        if contains(f(i), HEA(j))
            in_HEA = [in_HEA i];
            break
        end
    end
end

```

```

for j = 1:length(STR)
    if contains(f(i),STR(j))
        in_STR = [in_STR i];
        break
    end
end
for j = 1:length(TBI)
    if contains(f(i),TBI(j))
        in_TBI = [in_TBI i];
        break
    end
end
end
%in_TBI = find(ds_all.Labels == "TBI");
in = [in_HEA, in_STR, in_TBI];
ds_match = subset(ds_all,in);
%ds_match_test = subset(ds_test,in);

```

Set aside  $\geq 50$  samples for IV

```

c = string(categories(ds_match.Labels));
numsub = max(50,floor(min(countcats(ds_match.Labels)).2));
disp("Using " + numsub + " for IV.")

```

Using 569 for IV.

```

in_iv = [];
in_cv = [];
for i = 1:length(c)
    in = find(ds_match.Labels == c(i));
    r = randperm(length(in));
    in_iv = [in_iv; in(r(1:numsub))];
    in_cv = [in_cv; in(r(numsub+1:end))];
end
X_Train=subset(ds_match,in_cv);
X_Test=subset(ds_match,in_iv);

```

Layers

```

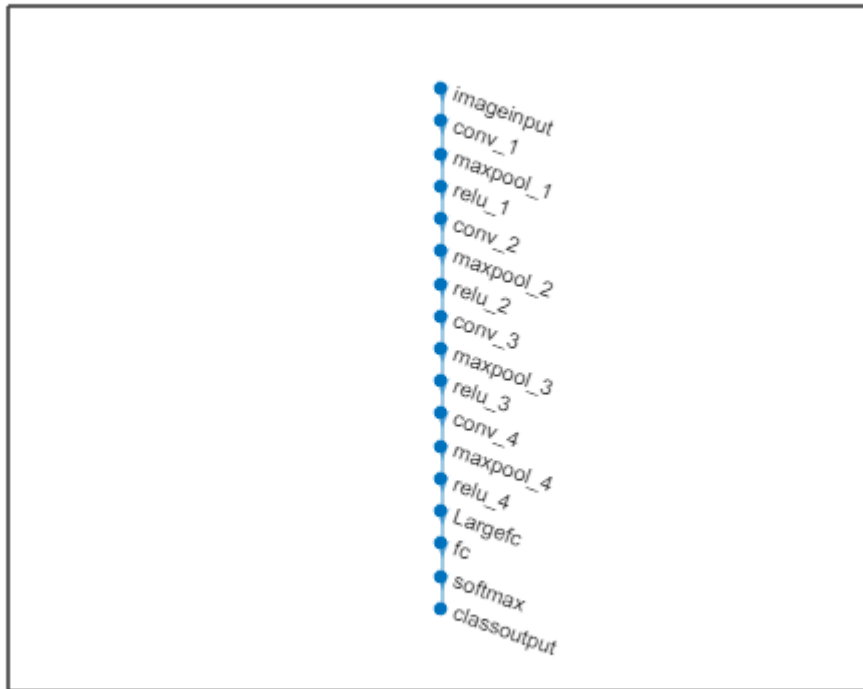
numclasses = X_Train.NumClasses;
layers = [
    imageInputLayer([134 134 6],"Name","imageinput","Normalization","none")
    convolution2dLayer([3 3],32,"Name","conv_1")
    maxPooling2dLayer([3 3],"Name","maxpool_1","Stride",[1 1])
    reluLayer("Name","relu_1")
    convolution2dLayer([3 3],32,"Name","conv_2")
    maxPooling2dLayer([3 3],"Name","maxpool_2","Stride",[2 2])
    reluLayer("Name","relu_2")
    convolution2dLayer([3 3],64,"Name","conv_3")
    maxPooling2dLayer([3 3],"Name","maxpool_3","Stride",[1 1])

```

```

reluLayer("Name","relu_3")
convolution2dLayer([3 3],128,"Name","conv_4")
maxPooling2dLayer([3 3],"Name","maxpool_4","Stride",[2 2])
reluLayer("Name","relu_4")
fullyConnectedLayer(100,"Name","Largefc")
fullyConnectedLayer(numclasses,"Name","fc")
softmaxLayer("Name","softmax")
classificationLayer("Name","classoutput");
figure;
plot(layerGraph(layers));

```



## Generate Local Data

6GB RAM needed. Only need to run one time.

```

% y = zeros(134,134,6,length(X_Train.Labels)); %~6Gb
% X_Train.MiniBatchSize = 1;
% reset(X_Train)
% for i = 1:length(X_Train.Labels)
%     temp = read(X_Train);
%     y(:,:, :,i) = temp.Predictors{1};
% end

```

Warning: For increased performance, remaining outputs are not shown. Consider reducing the number of outputs.

```

% v = zeros(134,134,6,length(X_Test.Labels)); %~6Gb
% X_Test.MiniBatchSize = 1;

```

```
% reset(X_Test)
% for i = 1:length(X_Test.Labels)
%     temp = read(X_Test);
%     v(:, :, :, i) = temp.Predictors{1};
% end
% save('TopoData.mat', "v", "y", "X_Train", "X_Test", '-v7.3');
```

Due to the large file size, an example file is not included for this dataset. To reproduce use the code above.

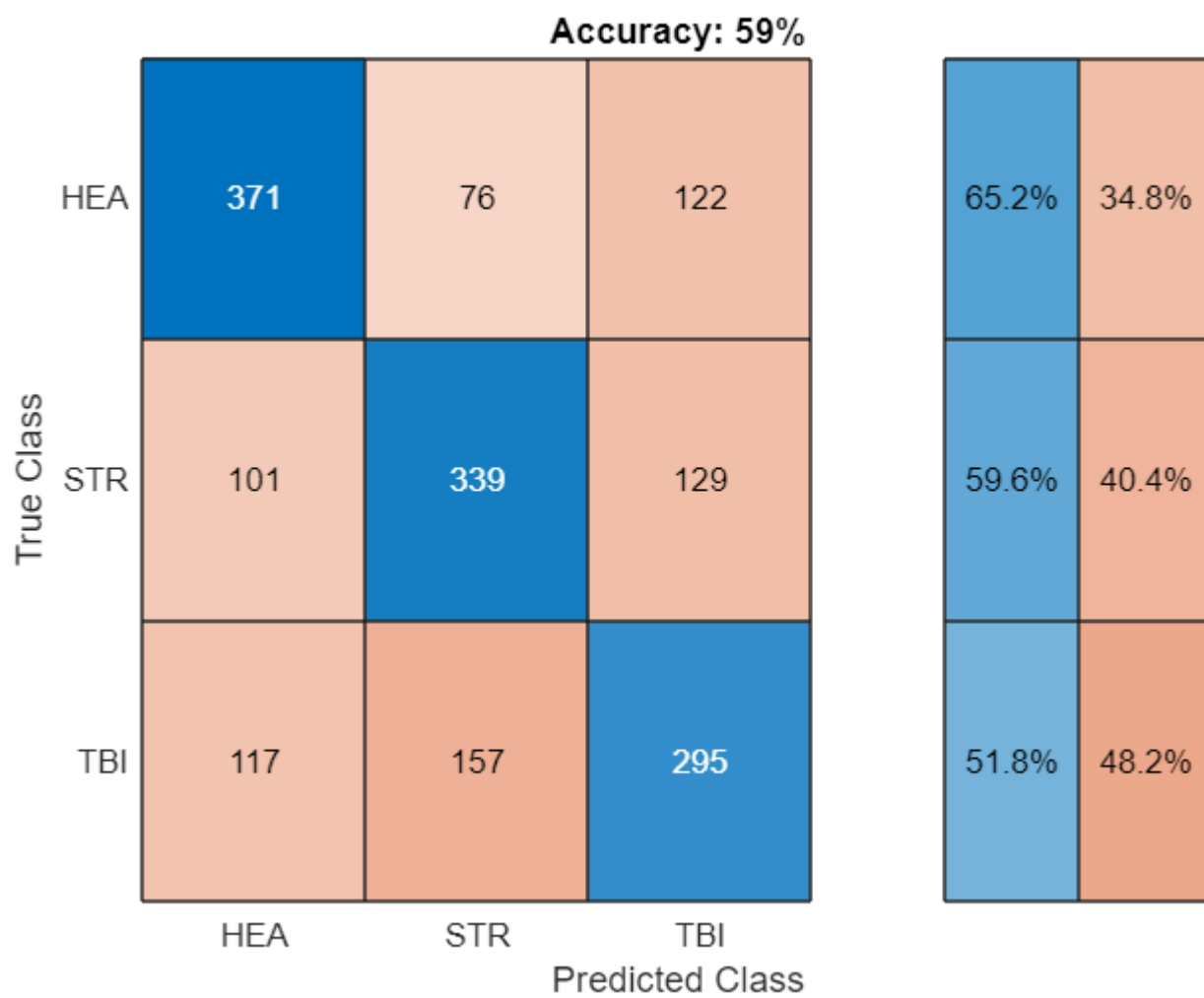
```
load('TopoData.mat');
```

## Run Network

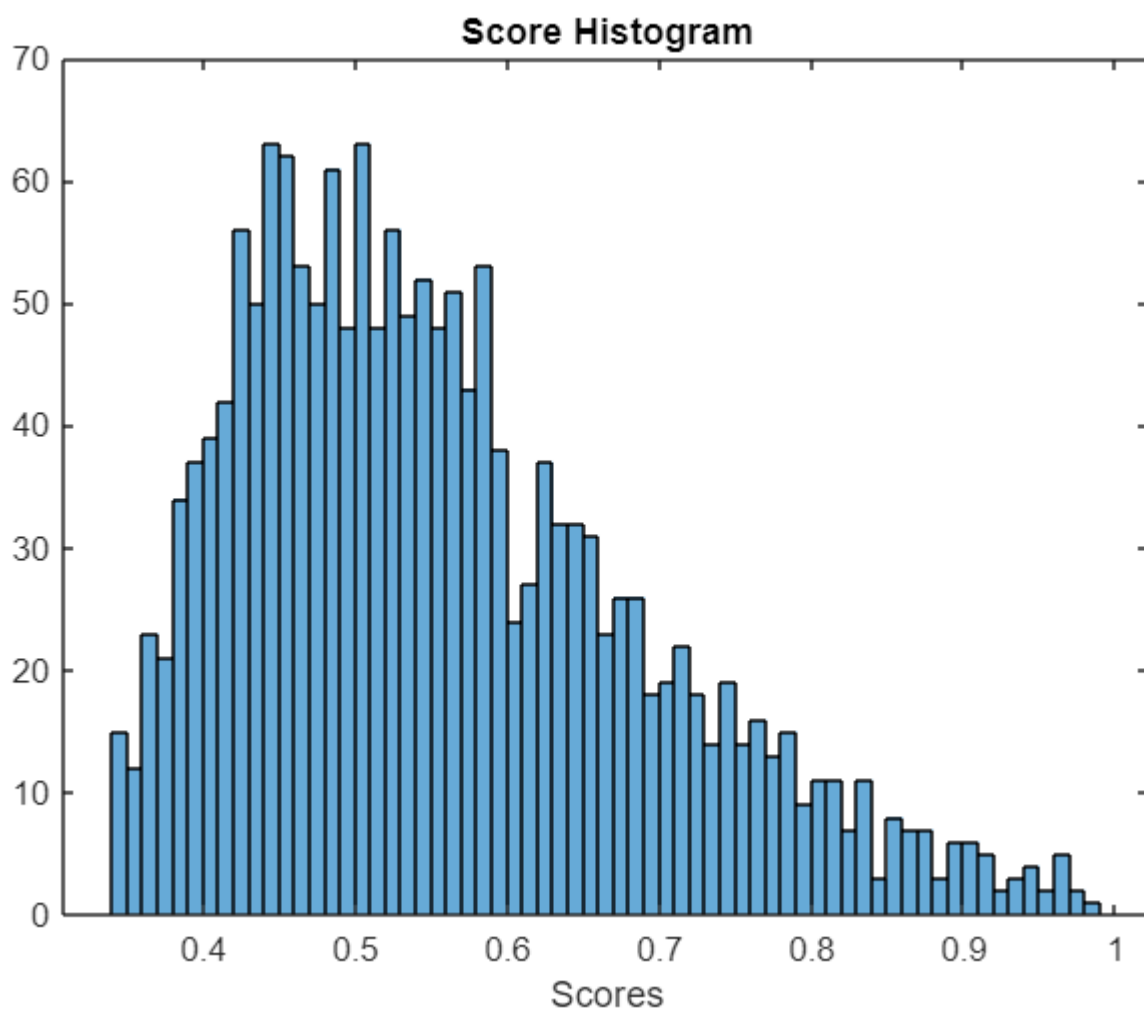
```
% gpuDevice(1);
% miniBatchSize=200;
% opts=trainingOptions("adam", "Plots", "training-
progress", "ExecutionEnvironment", "gpu", ...
%     "Shuffle", "every-epoch", "MiniBatchSize", miniBatchSize, "MaxEpochs", 300, ...
%
% "InitialLearnRate", 0.001, "LearnRateSchedule", "piecewise", "LearnRateDropPeriod", 100, .
% .. "GradientThreshold", 2, ...
%     "OutputNetwork", "best-validation-loss", "ValidationData",
% {v, X_Test.Labels}, "ValidationFrequency", 100, ...
%     "ValidationPatience", 20, "L2Regularization", .001);
% net=trainNetwork(y, X_Train.Labels, lgraph_1, opts);
```

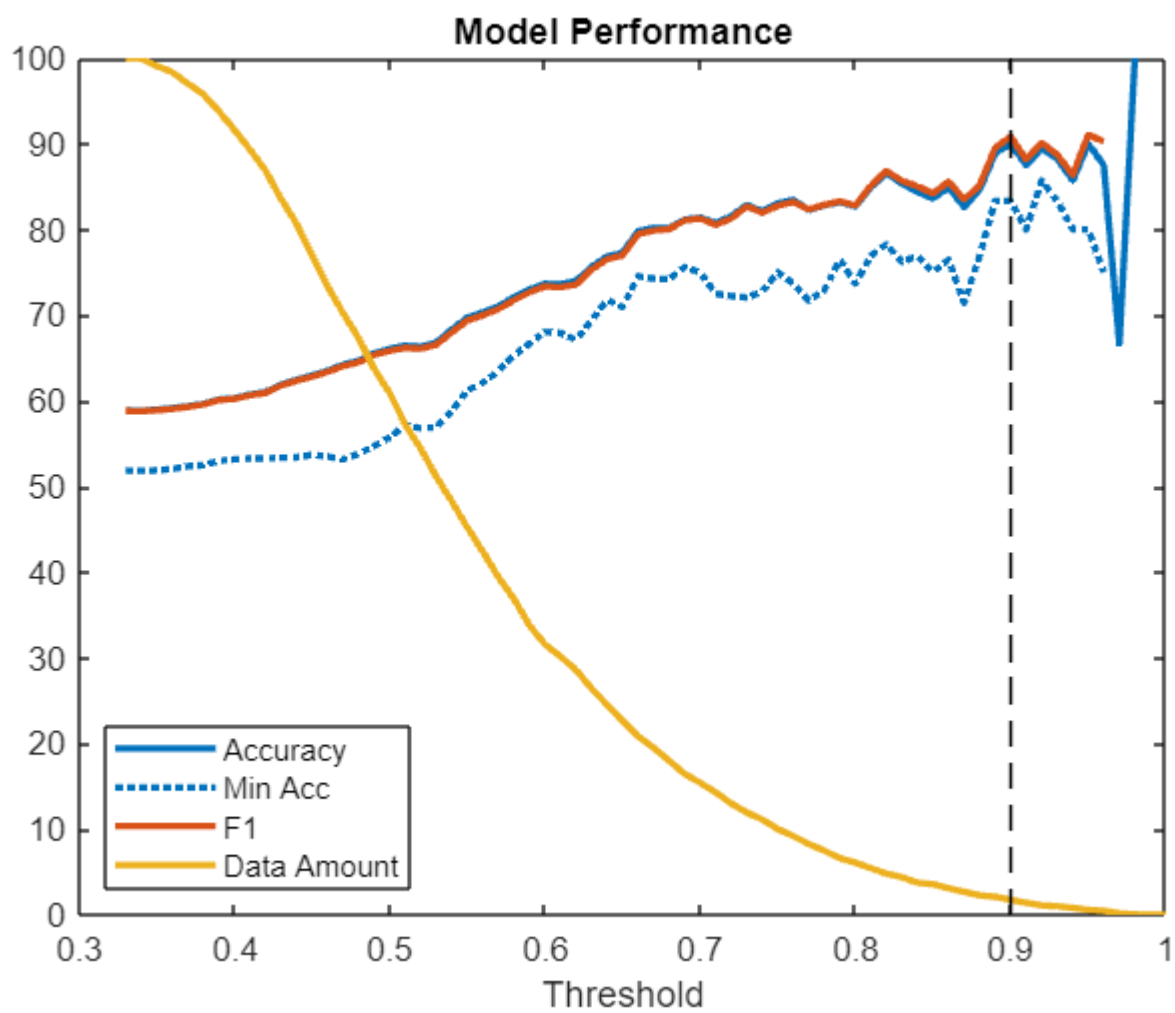
## Basic Architecture

```
load("Topo_BasicNet.mat", "net3")
basic3 = MdlResults(net3, v, X_Test.Labels);
classify(basic3);
```



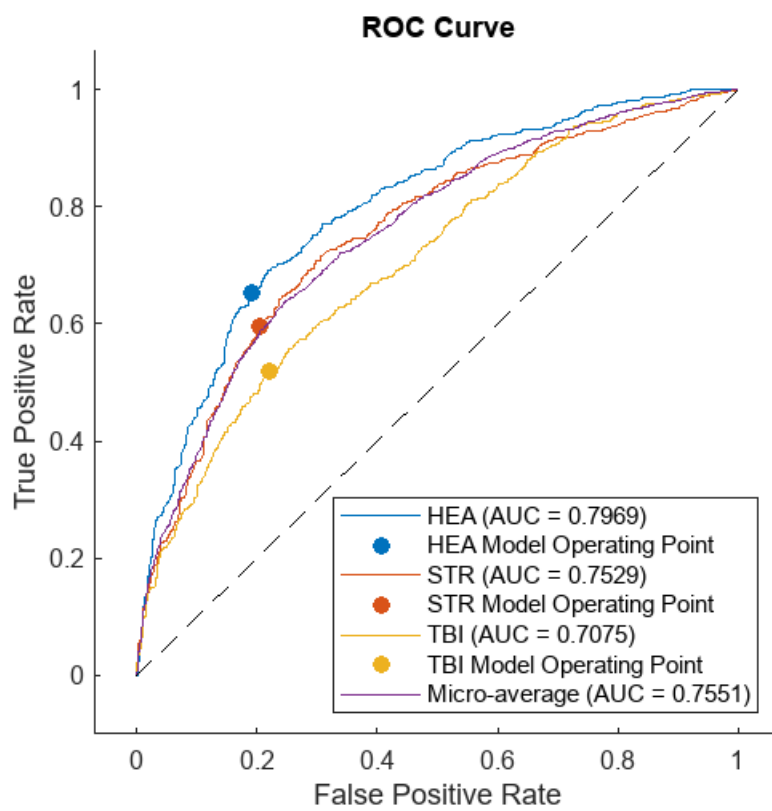
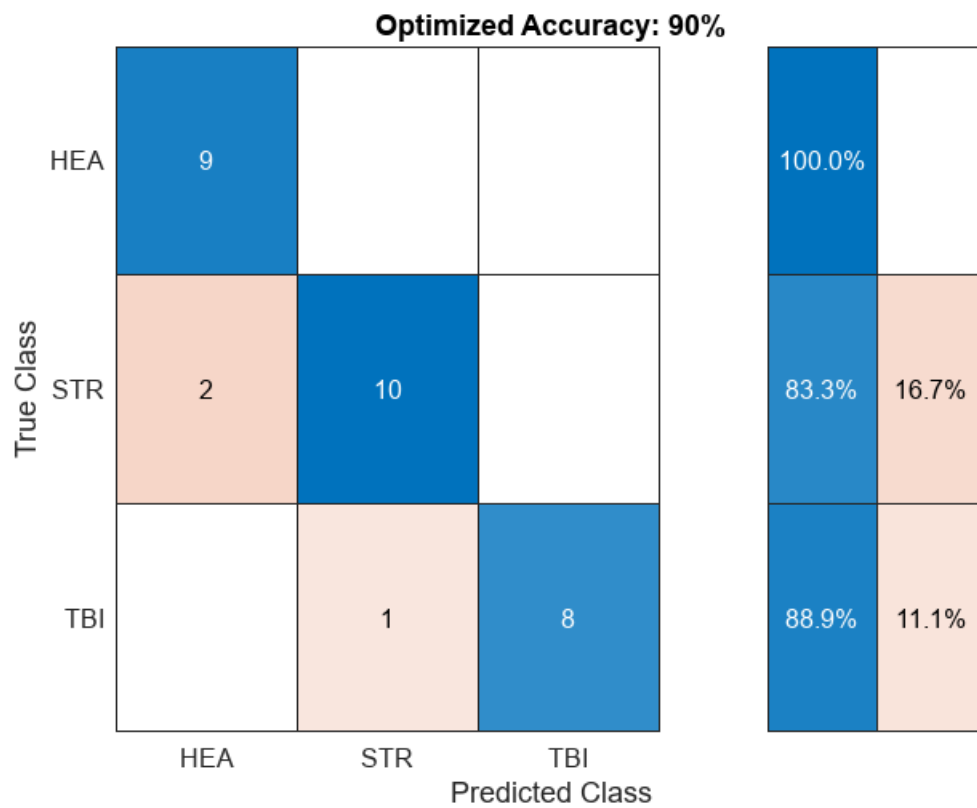
```
metrics(basic3);
```





Data Remaining: 0.017575





**Disclaimer**

This software and documentation (the "Software") were developed at the Food and Drug Administration (FDA) by employees of the Federal Government in the course of their official duties. Pursuant to Title 17, Section 105 of the United States Code, this work is not subject to copyright protection and is in the public domain. Permission is hereby granted, free of charge, to any person obtaining a copy of the Software, to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, or sell copies of the Software or derivatives, and to permit persons to whom the Software is furnished to do so. FDA assumes no responsibility whatsoever for use by other parties of the Software, its source code, documentation or compiled executables, and makes no guarantees, expressed or implied, about its quality, reliability, or any other characteristic. Further, use of this code in no way implies endorsement by the FDA or confers any advantage in regulatory decisions. Although this software can be redistributed and/or modified freely, we ask that any derivative works bear some notice that they are derived from it, and any modified versions bear some notice that they have been modified. The Software is not intended to make clinical diagnoses or to be used in any way to diagnose or treat subjects for whom the EEG is taken.