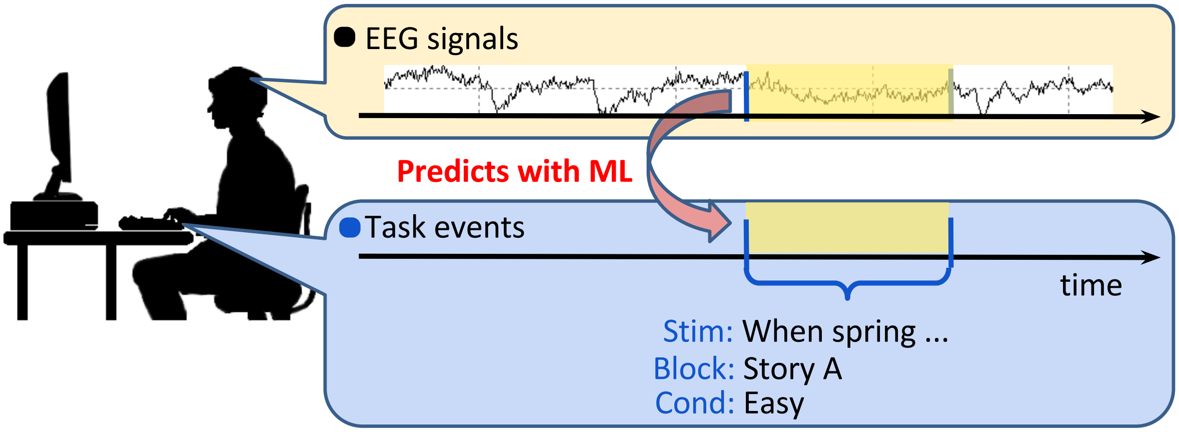
EEG-ML

1. What is EEG-ML?

EEG for Machine Learning (EEG-ML) is an effort to facilitate the use of portable electroencephalography (EEG) devices in the intelligent tutoring systems and education community. The EEG signal is a voltage signal that can be measured on the surface of the scalp, arising from large areas of coordinated neural activity manifested as synchronization (groups of neurons firing at the same rate). This neural activity varies as a function of development, mental state, and cognitive activity, and the EEG signal can measurably detect such variation. Machine learning methods recognize the complex patterns in the EEG signals and construct classifiers that predicts students' brain states that are relevant to learning. Many general-purpose EEG processing software and machine learning packages have been implemented and distributed; however, combining the two often involves complicated coding effort. To assist researchers who are new to this topic, EEG-ML helps to simplify the process so that the researchers can focus on experiment design. EEG-ML inputs a EEG data set, a behavioral task data set, and some specification of a machine learning experiment that a researcher likes to run. EEG-ML generates and executes the code to train and test the machine learning classifiers.



Open *Use Case*

2. How to download and use EEG-ML?

Download Page

EEG-ML is implemented in Matlab, so you need to have Matlab installed and running. Commercial and student license of Matlab can be purchased from Mathworks. By the way, for those of you who are not familiar with Matlab, Mark S. Gockenbach has an excellent, introductory tutorial to Matlab.
Here's a quick 'Hello World' example:

1. Open Matlab
2. Navigate to the `classifier/base/src` folder
3. Add this folder to the path (you can do this by typing 『`addpath(pwd)`』 into the Command Window)
4. Navigate to the `classifier/example/src` folder
5. Run the `example_script.m` file (you can do this by typing 『example_script' into the Command Window)
6. You should receive the output 「`n=58 accuracy=0.79* p=0.00`」 after the algorithm runs for about a minute.

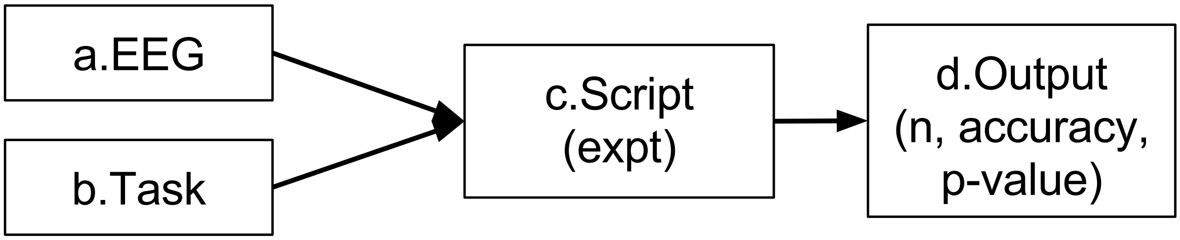
3. Step-by-step explanation of the EEG-ML toolkit

To set up an experiment, you should create a folder with two folders: 『`src`' and 『`data`'.

The 『`data`' folder should contain the files used for the experiment. The data you want to analyze should be separated into two files (1) an EEG file which holds the data collected from the EEG and (2) a task file that holds the behavioral labels i.e. the label that the classifier is trying to predict. All data files are CSV files with tabs as delimiters (meaning that there is a 『\t' separating every column and a 『\n' separating every line)

The 『`src`' folder should contain a .m file (you can copy the `example_script.m` file from the `classifier/example/src` folder as a reference). The contents of this file will be described in the 『script' subsection.

The following figure gives an architectural overview of the different components.



Open *Architecture*

a. EEG File

This is a file that contains the EEG data collected over the course of the experiment. It is recommended (for efficiency reasons) to break an EEG recording session into several segments (represented by different rows). Some of the columns could be left blank if no data is available. The columns are as follows (see `classifier/example/data/eeg.xls` for an example):

Column	**Description**	**Example**
machine	The name of the machine that the data is collected on (could be blank)	RT11-DEMETER
subject	The subject id of the participant whose EEG data is recorded in this segment.	52
start_time	The start time of this segment in the format 「year/month/day hour:minute:second.millisecond」.	2014/01/01 20:39:50
end_time	The end time of this segment in the format 「year/month/day hour:minute:second.millisecond」.	2014/01/01 20:39:50
stim	The stimulus shown to the subject during this segment (could be blank).	There was an Old Man with a nose,
block	The experimental block that this segment is in (could be blank).	..\\data\\stories\\2011VocabExp\\Old Man With a Nose
sigqual	The signal quality of the EEG signal (on a scale of 0 to 200, with 0 being best and 200 being worst).	25
rawwave	The raw signal from the EEG during collected between the start time and end time of this segment. The signal should be space delimited.	「0 7 12 17 33 28 ...」

b.Task File

This file contains the behavioral data i.e. the output we want to predict. Some of the columns could be left blank if no data is available. The columns are as follows (see `classifier/example/data/task.xls` for an example):

Column	**Description**	**Example**
machine	The name of the machine that the data is collected on (could be blank)	RT11-DEMETER

subject	The subject id of the participant whose behavior is recorded in this segment	52
start_time	The start time of this segment in the format 「year/month/day hour:minute:second.millisecond」	2014/01/01 20:39:50
end_time	The end time of this segment in the format 「year/month/day hour:minute:second.millisecond」	2014/01/01 20:39:50
stim	The stimulus shown to the subject during this segment (could be blank)	There was an Old Man with a nose,
block	The experimental block that this segment is in (could be blank)	..\\data\\stories\\2011VocabExp\\Old Man With a Nose
cond	The dependent variable of this experiment. The variable we want to predict.	1

c. Script

The experiment script creates a new `expt` struct which holds the parameters for our experiment. This object is passed into the function 『`run_experiment`' which runs the whole experiment.

See the comments in 『`classifier/example/src/example_script.m`' for a description of each of the parameters. Some important parameters are covered below:

Field	Description
task_file	this is the file location of the task file1
eeg_file	this is the file location of the eeg file
cv_subject	you can choose 『within' or 『between' to run within or between subject experiments
classifier	you can choose 『svm' for the SVM classifier or 『nbayesPooled' for the Gaussian naïve Bayes classifier
sampling_rate	this is the sampling rate of the EEG device you used in your experiment

Here's a pseudo-code that gives high level overview of the code structure:

```
[data, results] = run_experiment(expt)

    run_setup

    data = run_prepare_data(expt)
        task_data = read_task(expt.task_file)

        eeg_data = read_eeg(expt.eeg_file)
        eeg_data = smooth_eeg(eeg_data)

        data = align_data(task_data, eeg_data)

        data = gen_epochs(data)
        data = gen_epoch_features(data)
        data = gen_higher_order_features(data)

        data = gen_feature_matrix(data)
        data = filter_data(data)

    cv_splits = gen_cv_splits(data)

    cv_results = run_all_classification(data, cv_splits)
        gen_cv_data
        balance_data

        train_feature_selector
        apply_feature_selector

        train_classifier
        apply_classifier

    gold = data.task_M(:,COND);
    results = aggregate_results(data, cv_splits, cv_results)
    [n, accuracy, p] = evaluate_results(gold, results)
```

d. Output

The output of the experiment will be written to the Command Window in the following format:

```
n=[number of trials] accuracy=[accuracy of the
classifier] p=[p value of chi squared test against 50:50
accuracy]
```

4. How to cite?

Chang, K.M., Nelson, J., Pant, U., & Mostow, J. (2013). Toward Exploiting EEG Input in a Reading Tutor. *International Journal of Artificial Intelligence in Education*, 22 (1-2), 19-38.

Yueran Yuan

Last modified: Sat, Mar 01, 2014 8:58:19 PM

ADD FILES

Comments