Projection Transfer Learning using Phase Aware LSTM

Help File for MATLAB code-base | GitHub Repository

Author: Vishnu KN

This document provides a concise overview of the '**Projection Transfer Learning'** codebase. It describes the project structure, key scripts, configuration workflow, and the logic behind the projection-based transfer learning pipeline.

Project Overview

Projection Transfer Learning is a MATLAB implementation of end-to-end EEG model training (source domain) and projection-based transfer learning (target domain). The central objective is to map ***low-dimensional EEG inputs*** into the latent space learned by a ***high-dimensional source model***. This is especially useful for heterogeneous EEG setups (e.g., medical-grade → consumer-grade).

The pipeline is implemented in two phases:

1. **Source Domain training:** High-dimensional source models (*High-Dimensional EEG*)
2. **Target Domain Projection:** Low-channel projection models (*Low-Dimensional EEG*)

Supported source architectures include **DeepRNN**, **EEGNet**, and **Transformer** models. Supported projection layers include **FFNN**, **LSTM**, **GRU**, and the custom **Phase-Aware LSTM**. The goal is to map low-density EEG inputs into the latent space learned by a high-density source model.

Main Script

This script in the root-directory is the recommended starting point. One can explore the rest of the codebase from this starting point. It offers an interface to run all the experiments described in the original manuscript.

It performs:

1. Experiment configuration via **configureExperiment** class.
    - Domain / Model / Projection choices
    - Dataset choices (fold-specific)
2. Cross-validation training using **kFold_crossValidation**
3. Automatic result storage inside **cvalresults/**

Users should modify this file when running new experiments.

Repository Structure

The project is modular and organized for clarity and reproducibility:

Directory	Description
models/	Contains source architectures and projection models (FFNN, LSTM, GRU, PhaseAwareLSTM)
utils/	Dataset loaders, preprocessing, evaluation utilities, fold management, confusion matrices
experimentscripts/	Optional higher-level experiment pipelines or prototypes
datasets/folds/	Includes a **demo dataset (~100 trials)** to verify pipeline correctness *(You should mirror this folder structure for your own datasets.)*

This separation ensures that the core logic, utilities, models, and datasets remain independent. Users bringing external datasets must follow the same folder structure as the demo.

How to Configure Experiments

All configuration logic is isolated inside the **configureExperiment** class. Once the repository is downloaded, point to it within this class. This step is essential for making sure the project has access to all the required files to run the main script. Additionally, this class offers parameters and functionality that can be utilized for configuring all the different experiments described in the manuscript. It shall be initialized as:

config = configureExperiment()

<u>The main knobs of interest are:</u>

1. Dataset Type (CWL | MI)
2. Source Model Type (DeepRNN | EEGNet | Transformer)
3. Projection Layer (FFNN | LSTM | phase-LSTM)
   Domain (source-domain | target-domain)
4. Number of Classes (2 | 3)

Please refer to Fig. 1 for more information about how to configure different expriments.

Cognitive Workload Dataset (CWL)

Source-Domain training with DeepRNN Model

```
configs.typeNet      = 'DeepRNN';
configs.layerName    = 'DeepRNN';
configs.domain       = 'source-domain';
configs.datasetType  = 'CWL';
configs.ncats        = 2;
```

Target-Domain training with LSTM Projection

```
configs.typeNet      = 'DeepRNN';
configs.layerName    = 'LSTM';
configs.domain       = 'target-domain';
configs.datasetType  = 'CWL';
configs.ncats        = 2;
```

Projection with Phase-Aware LSTM

```
configs.typeNet      = 'DeepRNN';
configs.layerName    = 'phase-LSTM';
configs.domain       = 'target-domain';
configs.datasetType  = 'CWL';
configs.ncats        = 2;
```

Motor Imagery Dataset (MI)

Source-Domain training with a transformer model

```
configs.typeNet      = 'transformer';
configs.layerName    = 'transformer';
configs.domain       = 'source-domain';
configs.datasetType  = 'MI';
configs.ncats        = 2;
```

Target-Domain training with LSTM Projection

```
configs.typeNet      = 'transformer';
configs.layerName    = 'phase-LSTM';
configs.domain       = 'target-domain';
configs.datasetType  = 'MI';
configs.ncats        = 2;
```

Source-Domain training Multi-category Classification

```
configs.typeNet      = 'EEGNet';
configs.layerName    = 'EEGNet';
configs.domain       = 'source-domain';
configs.datasetType  = 'CWL';
configs.ncats        = 3;
```

Fig. 1: Different combinations of the configureExperiment class are described here. The class is initialized as *config = configureExperiment()* for this example.

One the experiment is performed, the pretrained source models are saved in '*projectRoot/pretrainedModels/*', while the results are stored in '*projectRoot/cvalresults/*'.

Projection Models

Target-domain learning uses projection modules to transform low-dimensional EEG into the latent representation of a pretrained high-dimensional source model. The proposed Phase-Aware LSTM is provided as a MATLAB deep learning layer class ("PhaseAwareLSTM.m"). Three baselines are provided which are standard layers available in MATLAB (version 2022a onwards). These layers can be chosen from the parameter combination described in Fig. 1.

Custom MATLAB Layer: Phase-Aware LSTM

The custom layer is implemented in: (*utils/modelstuff/layerdefinitions/phaseAwareLSTM.m*).

In MATLAB syntax, similar to an LSTM layer, the phaseAwareLSTM can be instantiated as: *layer = phaseAwareLSTM(hiddenSize, Name="phase_lstm", OutputMode="sequence");*

This layer implements a magnitude–phase recurrent state using:

- LSTM-style gating for magnitude
- Half-angle tangent parameterization for phase rotations
- Fully real-valued operations (no complex datatype needed)

This module is the central contribution of the manuscript.

Demo Dataset

A small test fold (~100 trials) is provided under: projectRoot/*datasets/folds/CWL/....* Its purpose is only pipeline verification, not meaningful evaluation.

To use your own dataset:

1. Download or preprocess your EEG dataset
2. Create folds externally
3. Mirror the same directory structure as the demo dataset
4. Update datasetType and fold paths in configureExperiment

This ensures full reproducibility with the training/testing pipeline.

Example Experiment Instance

To run an experiment:

1. Open **projectionTL_mainScript.m**
2. Adjust the configuration parameters
   (datasetType, layerName, domain, network architecture, etc.)
3. Ensure your dataset matches the demo fold structure
4. Run the script

The trained models and evaluation files will automatically appear in:
cvalresults/<dataset>/<model>/

Manuscript / Github / Citation

The manuscript describing Phase-Aware LSTM can be accessed at:
https://iopscience.iop.org/article/10.1088/1402-4896/ae229d

Please cite the manuscript as:

V. K N, S. Sundaram, and C. N. Gupta, "*Phase-aware LSTM projections for learning from heterogeneous electroencephalogram montages,*" Phys. Scr., 2025, doi: 10.1088/1402-4896/ae229d.

The codes can be download from GitHub via: https://github.com/your-username/PhaseAwareLSTM