Guo *et al. Journal of Big Data* (2023) 10:79
https://doi.org/10.1186/s40537-023-00758-9

Journal of Big Data

METHODOLOGY Open Access

Gaussian transformation enhanced semi-supervised learning for sleep stage classification

Yifan Guo[1], Helen X. Mao[2], Jijun Yin[1] and Zhi-Hong Mao[1,3*]

*Correspondence: zhm4@pitt.edu

[1] Department of Electrical and Computer Engineering, University of Pittsburgh, Pittsburgh, PA 15261, USA
[2] North Allegheny Senior High School, Perry Hwy, Wexford, PA 15090, USA
[3] Department of Bioengineering, University of Pittsburgh, Pittsburgh, PA 15261, USA

Abstract

Sleep disorders are significant health concerns affecting a large population. Related clinical studies face the deficiency in sleep data and challenges in data analysis, which requires enormous human expertise and labor. Moreover, in current clinical practice, sleep data acquisition processes usually cover only one night's sleep history, which is too short to recognize long-term sleep patterns. To address these challenges, we propose a semi-supervised learning (cluster-then-label) approach for sleep stage classification, integrating clustering algorithms into the supervised learning pipeline. We test the effectiveness of the proposed semi-supervised learning approach on two architectures: an advanced architecture using deep learning for classification and *k*-means for clustering, and a relatively naive Gaussian-based architecture. Also, we introduce two novel Gaussian transformations to improve the robustness and accuracy of the Gaussian-based architecture: assembled-fixed transformation and neural network based transformation. We reveal the effectiveness of the proposed algorithm via experiments on whole-night electroencephalogram (EEG) data. The experiments demonstrate that the proposed learning strategy improves the accuracy and F1 score over the state-of-the-art baseline on out-of-distribution human subjects. The experiments also confirm that the proposed Gaussian transformations can significantly gain normality to EEG band-power features and in turn facilitate the semi-supervised learning process. This cluster-then-label learning approach, combined with novel Gaussian transformations, can significantly improve the accuracy and efficiency of sleep stage classification, enabling more effective diagnosis of sleep disorders.

Keywords: Gaussian transformation, Semi-supervised learning, Sleep stage classification

Introduction

The rapid development of deep learning has created models that are reliable and practical in many aspects of engineering and technology, surpassing human experts in certain fields [1]. However, typical state-of-the-art deep models usually require enormous amount of labeled data for training in supervised learning missions like classification and regression. For some specific fields where data labeling is utterly time-consuming and expensive, the models must exploit unlabeled data more efficiently to manifest satisfying

performance [2]. One representative scenario is sleep disorder research, where patients' whole night sleep histories need to be manually labeled by well trained experts [3] to get access to patients' sleep rhythms, which are often represented as sleep stage transitions. According to the recent American Academy of Sleep Medicine (AASM) scoring manual [3], sleep stages contain non-rapid eye movement (NREM) and rapid eye movement (REM) sleep, and NREM sleep can be further subdivided into N1, N2, and N3 stages. In medical practices, when using brain waves for sleep stage classification, physicians prefer to use EEG signals than other physiological measurements such as electromyography (EMG) and electrooculography (EOG) because EEG signals contain rich information about sleep [4].

Since manual labeling of sleep EEG data is challenging and time-consuming, a fully labeled data set of sleep EEG signals normally consists of data from fewer than 100 subjects. This challenges the performance of classifiers on out-of-distribution subjects, given the significant inter-subject variability in EEG signals. Although fully labeled EEG data are not abundant, unlabeled EEG histories are much more accessible. Utilizing the geometric patterns reflected by the unlabeled data, a classifier can adjust for the out-of-distribution subjects thus maintains its performance. For such reason, machine learning approaches which can exploit unlabeled data are very appealing for EEG based sleep stage classification. This direction has been explored by many researchers [5–12].

In this paper, we propose a semi-supervised, cluster-then-label strategy, synergistically integrating a clustering procedure into the generic supervised learning pipeline. We show that the proposed strategy evidently enhances the classifier's performance on out-of-distribution subjects. Moreover, to improve the accuracy of pseudo-labels produced by the clustering process, we investigate Gaussian transformations for EEG band power features. Two types of transformations are explored: assembled-fixed transformation and neural network based transformation. With raised normality of features (meaning that the distributions of the features after the normal transformations are closer to Gaussian distributions), the Gaussian-based model can potentially achieve higher accuracy in both clustering and classification steps.

The rest of the paper is organized as follows: "Related works" briefly reviews the related works, and "Methods"explains our proposed methods. "Results" presents the results and discusses the observations. Conclusions are presented in "Conclusion".

Related works

Sleep stage classification

Sleep stage classification is a crucial step for the study of sleep from the perspectives of both neuroscience and health care. Researchers have developed numerous machine learning approaches for sleep stage classification. These approaches span from supervised to unsupervised, semi-supervised, and transfer learning.

Since classification is traditionally a supervised learning problem, supervised learning is the most consulted field for sleep stage classification. Researches in this field usually strive for better structures of classifiers and for more expressive and robust features. The two most commonly used classifiers are support vector machines (SVM) [4, 13, 14] and

artificial neural networks [15–18]. Other classifiers include linear discriminate analysis (LDA) [19], bagged tree [20], etc. For feature extraction, wavelet is the most popular method [14, 15, 19]. Another way for feature extraction is to arrange the input of a classifier as a graph [4, 13, 18]. Some deep models directly use raw signals to realize end-to-end approaches [17, 18].These approaches can attain satisfactory performance on an identically and independently distributed (i.i.d.) test set, but they often rely on large sets of labeled data for training and might not generalize well when dealing with out-of-distribution subjects.

Compared with supervised learning approaches, unsupervised learning methods are less commonly used for sleep stage classification. This is mainly because unsupervised leaning approaches do not explore the information from labels thus they usually require extra steps before generating predictions. One representative work in this field is contributed by Rodríguez-Sotelo et al., who extract entropy features from multi-channel EEG signals, analyze feature relevance with the Q-α algorithm, and partition the data with the J-means clustering algorithm [21].

Semi-supervised learning, utilizing both labeled and unlabeled data, is a promising technique for sleep stage classification. Munk et al. additionally introduce unlabeled data counterpart into their maximum likelihood estimation (MLE) cost function and propose their own form of conditional probability of unlabeled data [5]. Wuzheng et al. improve sparse concentration index to evaluate pseudo-labels' confidence [6], and Bai and Lu use small fully labeled data to pre-train the classifier and feed the generated pseudo-labels back to the model for training [7]. Li et al. focus on children sleep analyses and propose a bi-stream adversarial learning network to generate pseudo-labels with higher confidence and catch the desired feature distribution using a powerful symmetric positive definite manifold structure in the student branch [22]. They also propose a multi-task contrastive learning strategy for semi-supervised pediatric sleep stage recognition, which enhances the neural network's representation ability with signal-adapted transformations [23].

Transfer learning has also been explored for sleep stage classification. For example, Zhao et al. add domain classifiers to basic convolutional neural networks (CNN) to learn domain information from different levels [8]. Jadhav et al. pre-train a CNN on ImageNet data set, extract time-frequency features from raw EEG data with continuous wavelet transform, and retrain the network on these features [9]. Other transfer learning researches include [10–12].

Domain generalization

Traditional machine learning models are trained based on the i.i.d. assumption of training and test data. This assumption fails in certain biomedical fields including sleep disorder research, where the size and diversity of a data set are usually limited. Such data deficiency confines the models to generalize well on out-of-distribution subjects. Solving this distribution shifting problem by simply gathering more data in such fields is prohibitively impossible because collecting data is very expensive and expertise intensive. Various approaches are proposed in domain generalization (DG) to enhance the

generalization ability of models when the target domains' distributions are different yet related to the distributions of the training domains. These approaches can be categorized into three categories: data manipulation, representation learning, and learning strategies. Detailed surveys can be found in [24] and [25].

Data manipulation

Data manipulation methods enrich the training set by manipulating existing data points. Following this track, there are two popular strategies: data augmentation and data generation.

Data augmentation distorts the initial data set with various operations including adding noise, flipping, rotation, etc. It is a general strategy for improving model's robustness and is not limited to DG. Being required to handle the distorted data, the model has to capture general features of different domains. One special data augmentation method is called adversarial augmentation [26–28]. Specific noises are designed forcing the current model to misclassify. By explicitly overcoming its current weakness, the model can generalize better.

Data generation based DG strengthens the model's generalization capability by generating diverse data points. Unlike data augmentation, which manipulates the original data, data generation first trains a generative model using the current data set then produces new data with the generative model. Popular generative techniques include variational auto-encoder (VAE) [29], generative adversarial networks (GAN) [30], and Mixup [31].

Representation learning

Representation learning conceptually decomposes a prediction function into two parts—the feature extractor and the executor (e.g., a classifier). A major subcategory of representation learning is domain-invariant representation learning, which is built on the theory that domain invariant features are general and transferable to different domains. One of the most popular representation learning methods is kernel based method: it projects original data points onto a higher dimensional feature space to construct better patterns and avoids computational burden with kernel tricks [32–34]. Also, many methods have been proposed with the idea of domain adversarial learning [35–37] and explicit feature alignment [38–40]. The former uses adversarial learning to reduce domain discrepancy in a manifold space and the latter uses explicit distribution alignments or feature normalization to align the feature distributions across domains.

Learning strategies

Numerous learning strategies can be used for DG directly or with minimum modifications. They can be categorized into ensemble learning based DG, meta learning based DG, and others.

Ensemble learning is built upon an assumption that any input is a weighted superposition of existing training domains. Thus, the final prediction of that input can be obtained by assembling multiple models from different domains. Mancini et al. use a domain predictor to generate weights for results from domain-specific predictors and then yield the final predication as a weighted sum [41]. Segu et al. compute the domain-specific weights according to the distance between the batch norm statistics of the target sample and those of each training domain [42]. The classifiers for all the domains share the parameters except the batch normalization parameters.

Meta learning is also referred to as "learning to learn," which inducts a general model from multiple sources. Li et al. stimulate distribution variations by randomly dividing source domains into meta-training and meta-test domains at each training iteration [43]. Balaji et al. parameterize the regularization term with a separate neural network. This regularizer is trained with meta learning so that it can enable generalization through domains [44]. Other studies in this category include [45–47].

There are also other learning strategies that can be adopted to DG, and the proposed method in this paper belongs to this category. Carlucci et al. propose a self-supervised method that learns general representations by solving jigsaw puzzles [48]. Li et al. train the feature extractor and the classifier using episodic training [49]. Self-challenging mechanism is used in [50] to iteratively abandon domain-specific features.

Gaussian transformations for EEG signals

Models that assume their inputs subject to Gaussian distributions are usually simple, and their behaviors are easy to interpret. However, most EEG features are not subject to Gaussian distributions by nature. Researchers have tried to modify the distributions of EEG features for better normality and then easier classification. In [51], Gasser et al. compare the performance of various fixed transformations like $\sqrt{x}$, $\log(x)$, and $\log(x/(1-x))$, where the x's are either absolute values of EEG band powers or relative band power ratios. These transformations can symmetrize skew distributions. Boyd and Lacher propose a two-step transformation procedure for clinical data. The first step removes the skewness, and the second step handles kurtosis [52]. All these works transform data in a complete open-loop manner. In other words, their transformations are designed only with prior statistical knowledge without any feedback from the transformed results. In this paper, we design two types of data-driven Gaussian transformations (one of which works in a close-loop manner), which are helpful to the proposed cluster-then-label strategy.

Methods

Problem formulation

Denote the a raw EEG signal data set as $S_0 = \{(s_1, y_1), ..., (s_N, y_N)\}$, where s_i's are raw EEG signals and y_i's are the according sleep stages. The set $S = \{(x_1, y_1), ..., (x_N, y_N)\}$ is derived from S_0 after replacing the raw EEG signals with extracted features. The clustering algorithm is denoted as G, which takes $\{x_1, ..., x_N\}$ as input and generates clusters (groups) $\{g_1, ..., g_K\}$. The classifier is denoted as C, which receives a feature vector x and

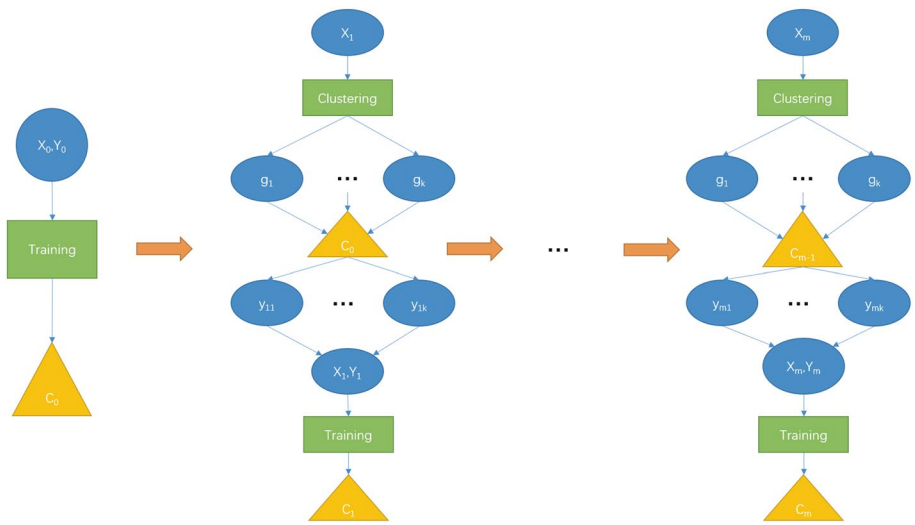

Fig. 1 Cluster-then-label algorithm pipeline

predicts the according sleep stage. The classifier's performance is evaluated on an independent data source S_{target}, where the features may be subject to a distribution different from that of S.

Cluster-then-label algorithm

Figure 1 shows the overall pipeline of our proposed cluster-then-label strategy. Blue nodes stand for data, where X and Y are the sets of EEG signal features and labels, respectively. Note that only the initial data set is labeled: X_0 and Y_0 are from S, while X_1 ,..., X_m are from S_{target}. Yellow triangles denote classifiers, and green rectangles represent clustering and training processes. We start with training a classifier C_0 on the fully labeled yet relatively small data set. In the following iterations, instead of directly generating pseudo labels using the pre-trained classifier, we use a clustering process to utilize the geometrical information in the feature space and thus correct the labels of the points which otherwise would have been mis-classified. In these iterations, unlabeled data, as a more accessible type of data source, are fed into a clustering model—Gaussian mixture model (GMM) or k-means as in our experiment. The clustering model returns K clusters, where K indicates the number of sleep stages we want to classify. Using the pre-trained classifier, each cluster is given a uniform label corresponding to the dominating class in that cluster. Then, these new data with pseudo labels are used to retrain the classifier. Such process can be repeated as long as new unlabeled data are available. We omit the feature extraction process in this figure, because sometimes it is a separate process and in other cases a part of the classifier. To sum up, the classifier C is initially trained on the fully labeled data set S and then retrained using the pseudo-labeled samples from S_{target}. This general structure is suitable for various combinations of classifiers and clustering algorithms.

We test the effectiveness of the proposed semi-supervised learning algorithm on two architectures: an advanced architecture using a deep learning model for classification

plus *k*-means for clustering, and a relatively naive architecture using LDA for classification plus GMM for clustering. The main reasons why we even try the Gaussian-based LDA plus GMM architecture are: i) Gaussian assumption can simplify and facilitate the theoretical analysis, which may gain insights about the robustness and accuracy of the proposed cluster-then-label strategy, and ii) the distributions of EEG band power features can be transformed to Gaussian-like distributions.

TinySleepNet classifier plus *k*-means clustering

We use a state-of-the-art deep model, TinySleepNet [17], as the classifier for the deep model plus *k*-means scheme. The model consists of a CNN part and a recurrent neural network (RNN) part. The CNN part contains four convolutional layers with a max-pooling layer and a dropout layer inserted after the first and the last convolutional layers. Using only one deeper branch of convolutional layers with smaller filters, the network can obtain the same effective receptive fields as adding another branch of convolutional layers with larger filters [17, 53]. The outputs of the feature extractor (CNN) can be viewed as non-normalized probabilities of sleep stages. These outputs may not be

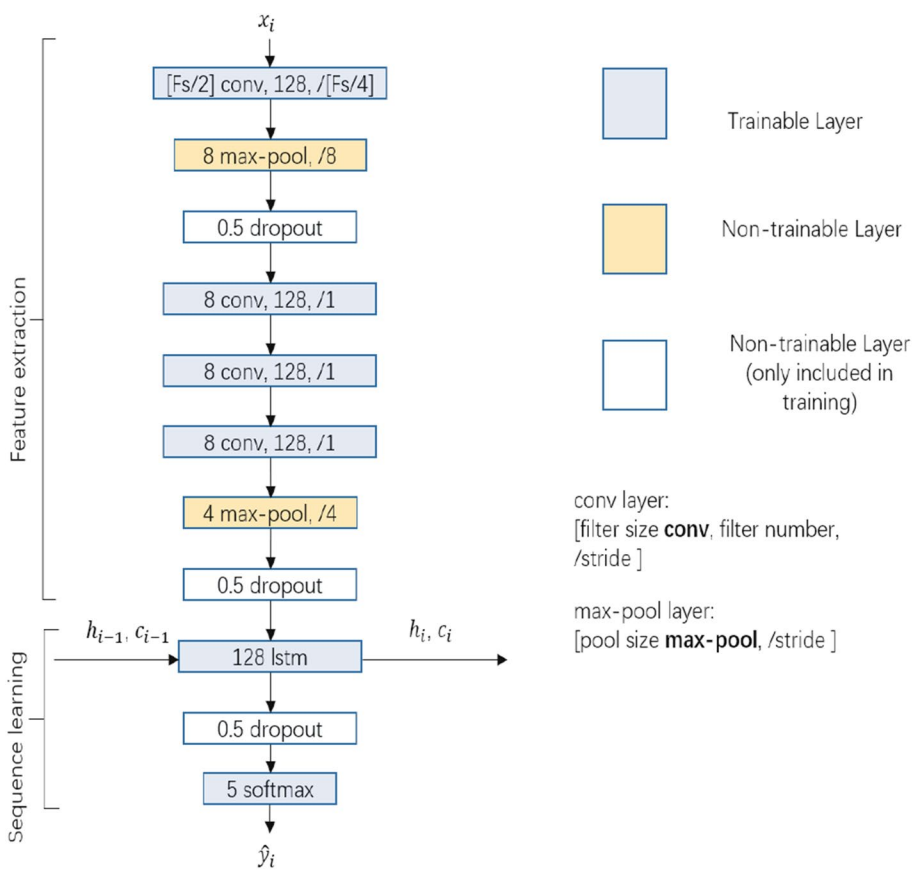

Fig. 2 The structure of TinySleepNet

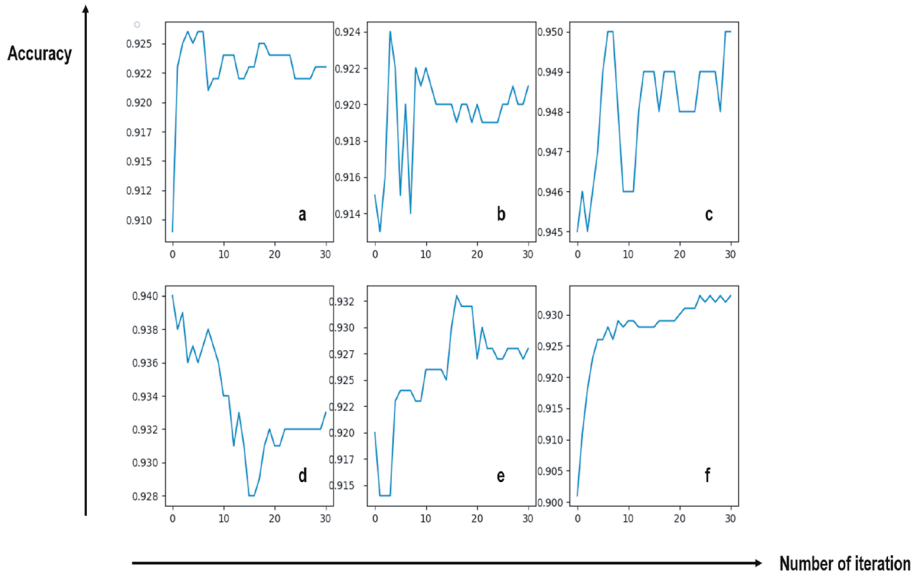

Fig. 3 Six examples (**a**-**f**) of cluster-then-label training histories. Curve **d** is a typical failure

subject to Gaussian distributions by nature and not suitable to be modified into Gaussians. This is the reason for choosing *k*-means over GMM for the clustering algorithm here.

The RNN part of the TinySleepNet captures the sleep transition rules that modifies the likelihood of current sleep stage based on the previous sleep stages. This part consists of one layer of unidirectional long short term memory (LSTM) cells (Fig. 2).

LDA classifier plus GMM clustering, part I: multitaper spectrogram

The cluster-then-label algorithm using deep learning plus *k*-means architecture can significantly improve the classifiers' performance on data from out-of-distribution subjects, but we observe some random failures of the algorithm during the experiments. This indicates that a successful execution of the algorithm depends on its random initialization, because all the trials of the algorithm execution are identical except for their initialization processes. Fig. 3 displays several training histories of independent toy experiments. Figure 3d illustrates a typical failure. Data for these experiments are sampled from two 2-D Gaussian distributions. To gain theoretical insights and simplify the potential robustness analysis of the cluster-then-label scheme, we try the Gaussian based LDA plus GMM architecture. Experimenting with this relatively naive architecture can also promote our understanding of more advanced cluster-then-label architectures.

Because LDA and GMM cannot directly handle high dimensional temporal inputs like raw EEG signals, we need a separate feature extraction step to use these models in our experiments. One of the most intuitive and natural feature sets for EEG signals is EEG band power feature. With this feature set, an EEG slice can be represented by a 4-D vector, where each dimension stands for the total power distributed into a certain frequency

band range (delta: 1–4 Hz, theta: 4–8 Hz, alpha: 8–12 Hz, and beta: 12–30 Hz). We use a technique called multitaper spectral analysis [54]. Spectral analysis is a classic tool for signal processing. It extracts the frequency information of a signal. However, typical spectral analysis approaches, for example, fast Fourier transform (FFT), suffer from the side lobe leakages and the high variance of EEG signals, which result in very noisy and unclear spectra. Instead, multitaper spectral analysis uses multiple specially designed tapes (or windows) to reduce the leakages and the variance by taking the average spectra. The tapes are called discrete prolate spheroidal sequences (DPSS). They are able to remove the false power from the side lobes and are orthogonal to each other. The feature extraction steps are formally described in Algorithm 1.

Algorithm 1 EEG Band Power Feature Extraction.

1: Denote s a raw EEG signal, t the duration of the signal in seconds, f the required frequency resolution, β, α, θ, and δ the bands of interest as defined in Section 3.4.
2: The time half-band-width product $w \leftarrow \frac{tf}{2}$.
3: The number of tapers is $m \leftarrow \lfloor 2w \rfloor - 1$.
4: Generate m DPSS tapers $\{t_1, ..., t_m\}$ according to w and m.
5: Separately multiply the EEG signal to each taper, getting $S = \{s_1, ..., s_m\}$.
6: Apply FFT to each element in S, and calculate the average of the results.
7: Sum up S on β, α, θ, and δ, obtaining the four dimensional feature x of s.

LDA classifier plus GMM clustering, part II: Gaussian transformation

As most EEG features are not subject to Gaussian distributions by nature, we introduce two methods (assembled-fixed based and neural network based transformations) to reshape the distributions of these features to be more Gaussian-like and thus improve the robustness and accuracy of the Gaussian-based LDA plus GMM architecture. Our research about Gaussian transformation focuses on scalar transformations, i.e., we transform one feature dimension at a time, because a multivariate Gaussian is a combination of multiple 1-D Gaussians.

Assembled-fixed transformation

In [51], Gasser et al. report their best results on resting EEG with the Gaussian transformation $\log(\frac{x}{1-x})$, where x stands for the relative band power ratio. The curve of this transformation is shown in Fig. 4. This transformation works in a way that dilutes points in the tails of the distribution because the curve becomes steeper in those areas. We follow this insight in our transformation. The difference is that our data contain EEG signals from multiple stages. This means the band power features are subject to a mixture distribution in our setting. Hence, we keep the basic shape of the transformation curve and apply a variational version of it to each stage. We take the combined curve as our final transformation. Algorithm 2 formally describes the transformation. Note that all sets should be treated as sequential data type in this algorithm.

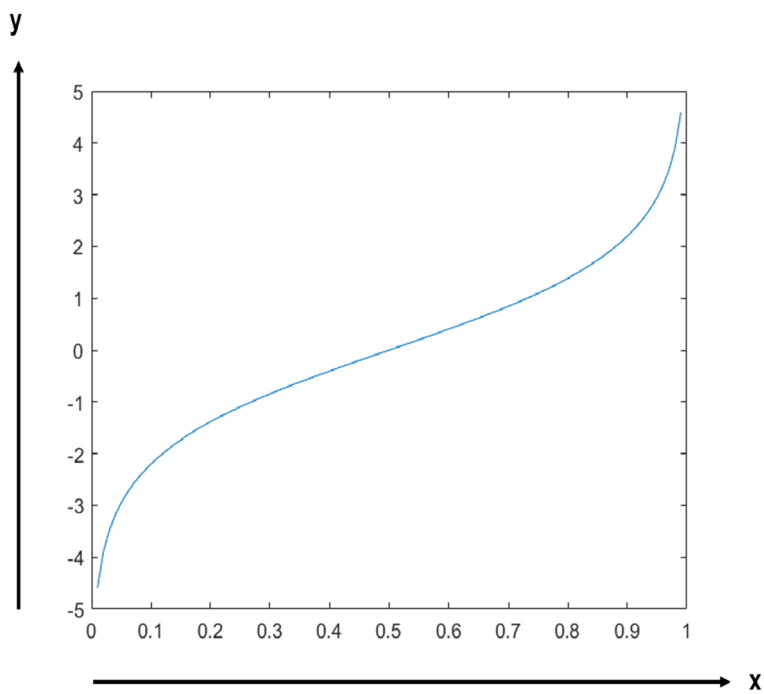

Fig. 4 The curves of the basic transformation $y = \log(\frac{x}{1-x})$

Algorithm 2 Manipulation of Basic Transformation.

Input: $S = \{(x_0, y_0), .., (x_N, y_N)\}$.
Define $R_{\log}$ as the range of $\log(\frac{x}{1-x})$ and r as the effective ratio.
for each sleep stage **do**
  $S_j = \{(x_i, y_i) \mid y_i =$ `current sleep stage`$\}$.
  Define L to be the length of S_j.
  $D_x \leftarrow L \times \frac{r}{2}$, $U_x \leftarrow L - D_x$
  $S_{\text{sort}} \leftarrow \text{sort}(S_j)$
  Define $D \leftarrow$ the D_x'th element of S_{sort}, $U \leftarrow$ the U_x'th element of S_{sort}
  $R \leftarrow U - D$, $\mu \leftarrow$ the mean value of x_i's in S_j.
  Define $S' = \{x_i \mid D < x_i < U\}$.
  Elementwise apply $S' \leftarrow \frac{S'-D}{R}$
  Elementwise apply $S' \leftarrow (\log(\frac{S'}{1-S'})/R_{\log}) \times R + \mu$
end for
$S \leftarrow$ `elementwise mean of all` S_j

In Algorithm 2, we first select a portion of $r \times 100\%$ of all points in each sleep stage, where $r \in (0, 1]$ is the effective ratio. Then these data points are scaled into (0, 1), and the original transformation is applied to them. Finally, they are re-scaled into $(\mu - 0.5R, \mu + 0.5R)$, where μ is the mean value of the original data points in the effective region and R stands for the range of that region.

The assembled-fixed transformation brings more flexibility than the transformations studied in [51] and better serves our cluster-then-label algorithm under the Gaussian assumption, but it still has some drawbacks. First, the curve's basic shape is fixed, which

limits the overall flexibility. Second, when dealing with data from the target domain, a new point needs to go through the transformation for every stage because we do not know which stage it belongs to. As a result, the steep tails of one stage may intrude into the other stages' flatten areas, leading to grooves around the centers the other stages. Finally, the designing process of the assembled-fixed transformation is still in an open-loop manner.

Neural network based gaussian transformation

To address the limitations of the assembled-fixed transformation, we parameterize each stage's transformation with a neural network. The transformation networks are generic multilayer perceptrons with a single hidden layer. To train the neural networks, our propose to use a loss function which is a modified statistic of Jarque-Bera (JB) normality test:

$$loss = \frac{n}{6}\left(S^2 + \frac{(K-3)^2}{4}\right) + \lambda\left(\bar{X}_{in} - \bar{X}_{out}\right)^2, \tag{1}$$

where $S = \hat{\mu}_3/\hat{\mu}_2^{3/2}$ is sample skewness and $K = \hat{\mu}_4/\hat{\mu}_2^2$ sample kurtosis. The notation $\hat{\mu}_i = \frac{1}{n}\sum_{j=1}^{n}(x_j - \bar{x})$ is the estimation of ith order central moment, where $\bar{x}$ is the mean value of x's. The term, leading with a hyperparameter λ, constrains the mean value shift caused by the transformation.

Results

Cluster-then-label using TinySleepNet and *k*-means

This section corroborates how the cluster-then-label strategy improved deep classifier's performance on raw EEG signals. We conducted our experiments on the open-source data set "Sleep EDF Expanded" [55]. All the EEG signals in this data set were sampled at 100 Hz and were sliced into 30 s pieces. The codes were implemented with Python 3.6 and TensorFlow 1.13.1.

First, we pre-trained the TinySleepNet classifier on the first twenty subjects using "twenty-fold" method. Specifically, we independently trained twenty classifiers, all of which started from random initialization. For each classifier, we selected a different subject out of the overall 20 subjects as the test set. Seventeen remaining subjects were used for training and another two subjects for validation. The model with the best performance on its test set was selected for further training.

In the second step, we mixed EEG data from three additional subjects to form a new data source. At this step, the annotations that came along with the data set were invisible in training process and were only used for evaluating the performance. With the mixed data source, the CNN part of pre-trained classifier was used for feature extraction, which compressed the raw, time serial data of 3000 dimensions into 5 dimensions. Then the clustering process was conducted based on the 5 dimensional features. We used the classic *k*-means clustering algorithm with random initial cluster centroids. Each cluster was assigned with the pseudo-label of its major class using the pre-trained classifier. The resulting clusters with pseudo-labels were then mixed and evenly divided into 6 folds. With a similar strategy as in the pre-training stage, we retrained the classifier 6 times independently. Each time a different fold was selected

as test data and another one for validation. The accuracy and f1 scores were improved by $2.06 \pm 1.16\%$ and $2.03 \pm 1.80\%$, respectively, with one round of cluster-then-label retraining. Detailed data can be found in Tables 1 and 2.

Cluster-then-label using LDA and GMM

In this section, we briefly demonstrate how the cluster-then-label strategy improved the performance of the simple LDA classifier in a binary (wake and N1 stages) classification problem. Here, the GMM was the clustering algorithm for the cluster-then-label strategy. The LDA classifier took 4-D band power features as their inputs. We used the EEG data from the first twenty subjects from "Sleep EDF Expanded" and extracted band power features using multitaper spectral analysis. Note that we discarded some data points to keep the number of points of each stage the same in the training set to avoid the bias from prior probabilities. The "twenty-fold" training strategy was also used. Each time, we selected one different subject for test, and eighty percent of the remaining data were used as the training set and twenty percent as the validation set. The classifier with the best performance on its test set was chosen for retraining.

As in "Cluster-then-label using TinySleepNet and k-means", we mixed the data from three new subjects as a new data source, keeping the annotations only used for performance evaluation. In the clustering process, the resulting clusters with pseudo-labels were mixed and evenly divided into 6 folds. With a similar strategy as in the pre-training stage (using LDA), we retrained the classifier 6 times independently with the newly added pseudo-labeled data. Each time a different fold was selected as test data and another one for validation. The accuracy and f1 scores were improved by $7.17 \pm 6.15\%$ and $9.17 \pm 4.98\%$, respectively, with one round of cluster-then-label retraining. Detailed data can be found in Tables 3 and 4.

Table 1 Improvements in accuracy with TinySleepNet plus *k*-means

Fold no.	1	2	3	4	5	6
Before retraining	74.6	74.2	82.4	84.9	89.6	83.4
After retraining	77.2	78.1	84.7	87.3	90.4	83.8

Table 2 Improvements in F1 score with TinySleepNet plus *k*-means

Fold no.	1	2	3	4	5	6
Before retraining	55.1	49.7	68.0	68.4	73.6	69.5
After retraining	58.4	54.8	69.5	70.7	73.6	69.5

Table 3 Improvements in accuracy with LDA plus GMM

Fold no.	1	2	3	4	5	6
Before retraining	0.57	0.72	0.63	0.83	0.62	0.65
After retraining	0.72	0.73	0.76	0.89	0.60	0.75

Table 4 Improvements in F1 score with LDA plus GMM

Fold no.	1	2	3	4	5	6
Before retraining	0.51	0.62	0.57	0.54	0.46	0.70
After retraining	0.65	0.64	0.73	0.63	0.50	0.80

Table 5 Performance (AD test statistics) of fixed transformation

Band	Beta	Alpha	Theta	Delta
Before transformation				
Wake	137.09	252.17	93.45	178.01
N1	4.15	6.39	2.14	4.86
N2	1.81	30.19	0.48	10.01
N3	2.49	2.70	1.16	0.22
REM	5.92	4.09	2.07	1.47
After transformation				
Wake	156.99	223.13	52.09	49.78
N1	2.26	3.96	2.87	2.40
N2	0.73	18.64	1.26	4.09
N3	1.19	2.44	1.54	0.38
REM	3.67	7.44	1.27	0.55

Assembled-fixed Gaussian transformation

According to [51], applying the transformation $\log \frac{x}{(1-x)}$ to relative band power ratio features can efficiently convert current feature distributions to be closer to Gaussian distributions. We reproduced their experiment on our data and the result is shown in Table 5. Note that data from multiple human subjects were included in this experiment. Though this fixed transformation can already reduce the AD test statistics, i.e., increasing normality, it cannot handle the scenario where data from different stages are mixed together. As designed in Algorithm 2, we conducted experiments of the assembled-fixed Gaussian transformation. We used the same data set as in "Cluster-then-label using LDA and GMM". One of the representative results is shown in Fig. 5 and Table 6. In Fig. 5, the transformation was applied on the delta band of the EEG data from the third subject in the "Sleep EDF Expanded" data set. We chose delta band because it gave best separability. We can observe that the transformation clearly fixed the skewness of the wake stage and made the peak of every stage more evident. The grooves (most clear in the middle of the wake stage) appear just as expected.

Neural network based Gaussian transformations

In this section, we show the results of neural network based Gaussian transformation. We used three layers, fully connected network structure. The hidden layer contained 400 nodes with rectified linear unit (ReLU) activation function. Adam algorithm [56] was used to optimize the model for 400 epochs. Data set were formed by mixing the relative band power features of the first ten subjects from "Sleep EDF Expanded" data

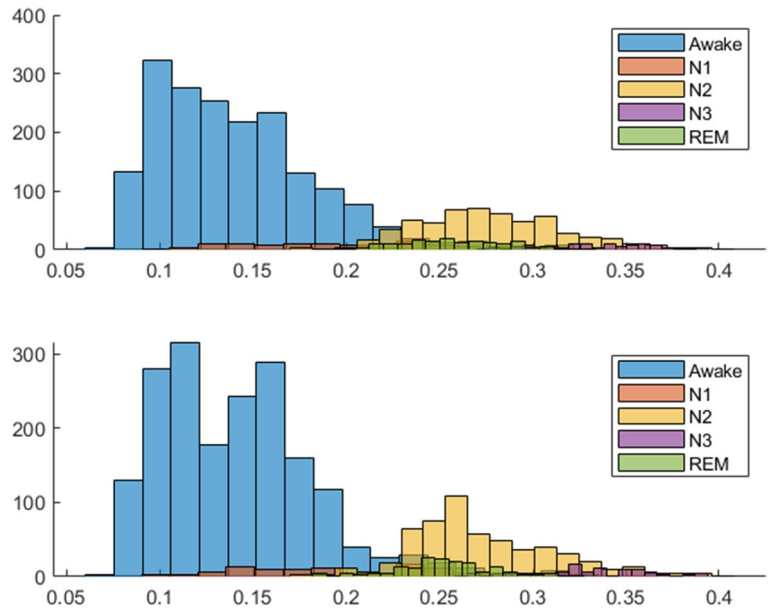

Fig. 5 Data distribution before (top) and after (bottom) assembled-fixed transformation

Table 6 Improvements in AD statistics (smaller value indicating better normality) with assembled-fixed transformation

Sleep stages	Wake	N1	N2	N3	REM
Before transformation	20.13	1.16	6.91	0.42	0.75
After transformation	26.70	0.82	1.44	0.28	0.21

set. We concentrated on wake and N1 stages and relative band power features of the delta band in our experiments.

We only considered the band power feature of the delta band from two sleep stages (wake and N1 stages) for the following reasons: (i) the major proportion of sleep disorders involve sleep onset difficulties, and this emphasizes the importance to detect wake and N1 stages; (ii) all the 5 sleep stages can be manually pair-wise distinguished based on single band information; iii) to learn a uniform transformation for 5-stages data, the network needs huge amount of hidden nodes and data for fine-tuning. With limited data availability in our scenario, large amount of hidden nodes will cause overfitting.

The resulting Gaussian transformations should be nearly monotonically increasing functions and have a reasonable output range so that they can preserve meaningful biological information. To induce such properties, we first initialized the network with the assembled-fixed transformation supervising with mean square error loss function. The training inputs for initialization was uniformly sampled in the range from 0.0001 to 0.9999 with the step size of 0.0001. Figure 6 displays the initialization result.

At the second step, we trained the network using JB loss with the hyperparameters: $\lambda = 0.1$, number of epochs $= 500$, batch size $= 128$, and learning rate $= 0.0001$. Ninety

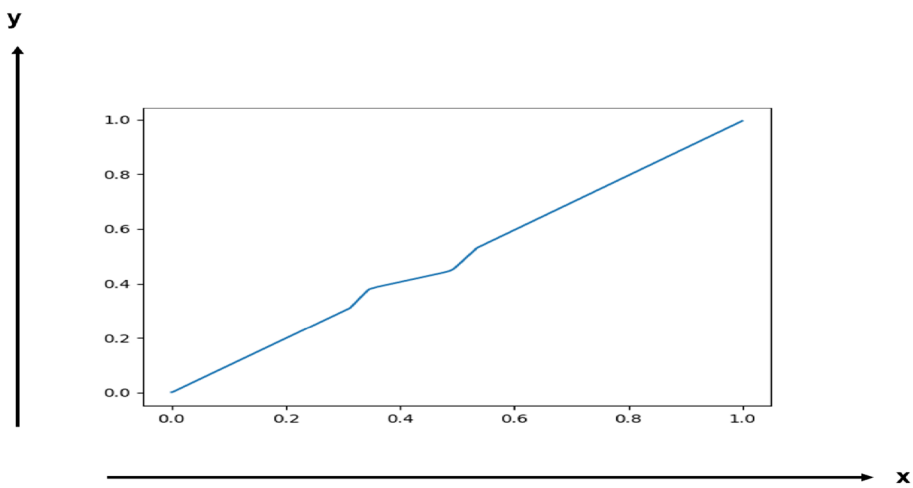

Fig. 6 The initialization result using assembled-fixed transformation as the target

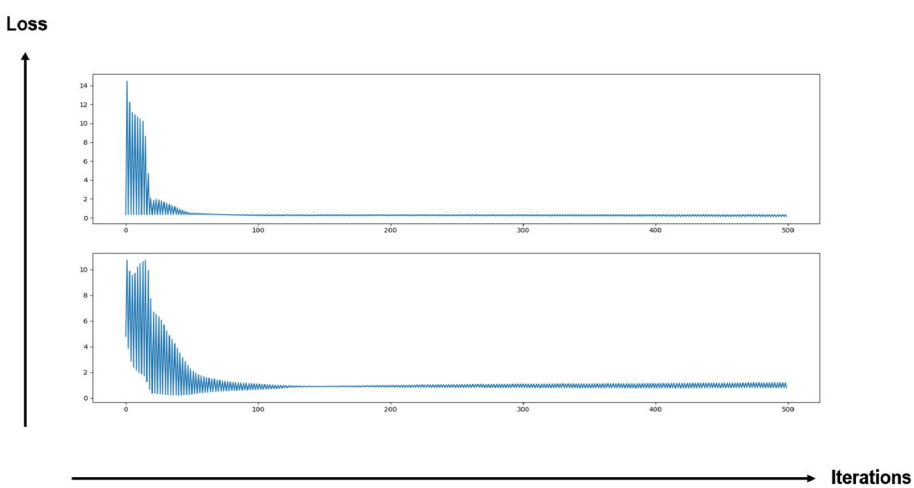

Fig. 7 The training history of the transformation network

percent of the data were used for training and the rest for validation. Data batches from the two sleep stages were alternately fed into the neural network. Figure 7 presents the training history of the transformation network. The first curve is the loss on the training set and the second curve is the loss on the validation set. The x axes denote the number of epoch and the y axes the loss values. The cyclical oscillations were due to the alternate data feeding. Figure 8 presents the modifications on the distributions of EEG relative band power features. The values of s in the subtitles are the JB normality test statistics, and the values of *mean* are the corresponding average values of the distributions. The first and second rows in the figure are the distributions before and after the transformation, respectively. The first and second columns correspond to the wake stage and N1 stage, respectively. We can observe that the normality statistics evidently decrease, which indicates the distributions are more Gaussian-like under the standard of JB normality test. The resulting transformation curve is shown in Fig. 9.

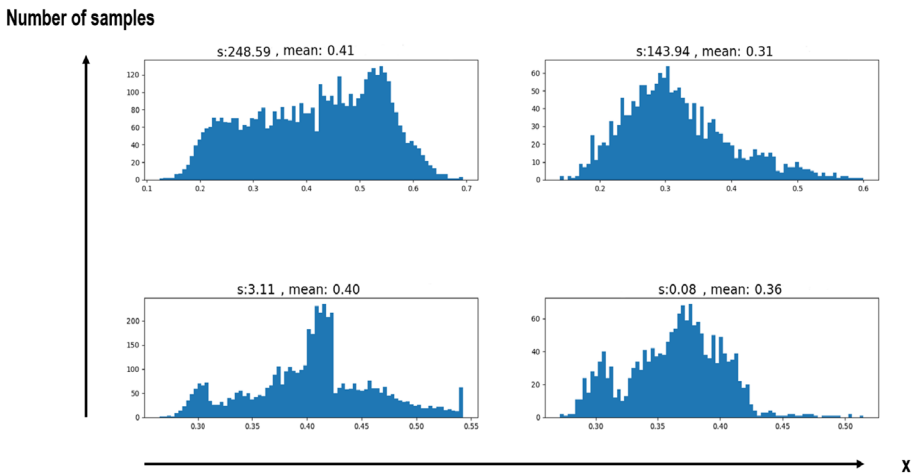

Fig. 8 The distributions of EEG relative band power features before and after the transformation. Values of *s* are the JB normality test statistics. The *mean* values are the corresponding average values of the distributions

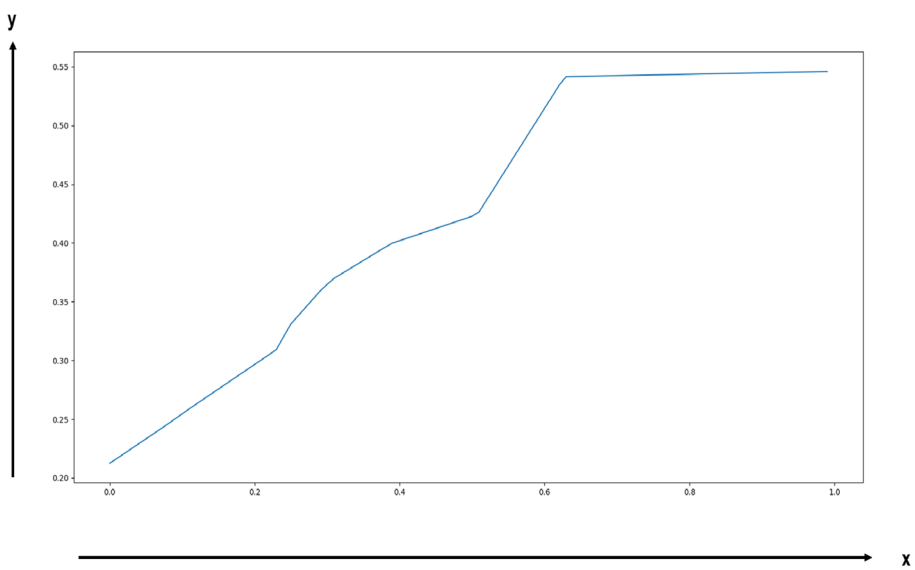

Fig. 9 The resulting transformation for relative band power features

Conclusion

In this paper, we propose a cluster-then-label algorithm and prove its effectiveness on an advanced deep learning based classifier and a relatively naive LDA classifier. The proposed method can evidently improve the classification performance on out-of-distribution subjects. Moreover, we introduce two types of Gaussian transformation to make the proposed method more robust and accurate in the LDA classifier plus GMM clustering architecture. Both transformations can improve the normality of the distributions of EEG relative band power features. The assembled-fixed transformation has the merits of accurate boundaries but works in an open-loop manner. The neural network based transformation optimizes the distributions in a close-loop

manner but is hard to tune the number of nodes in the hidden layer, balancing its flexibility and the ability of generalization.

Acknowledgements
Not applicable.

Author contributions
YG designed and implemented the proposed algorithms. HM actively involved in the algorithm designing and debugging works. JY and ZM provided valuable high-level guidance during algorithm designing, implementation, and paper writing. All authors read and approved the final manuscript.

Funding
This is a non-sponsor research program.

Availability of data and materials
The data sets generated and analysed during the current study are available in the PhysioNet repository, https://www.physionet.org/content/sleep-edfx/view-license/1.0.0/.

Declarations

Ethics approval and consent to participate
The used data set "Sleep EDF Expanded" is open-sourced under the "Open Data Commons Attribution License (ODC-By) v1.0" license.

Consent for publication
Not applicable.

Competing interests
The authors declare that they have no competing interests.

Received: 25 January 2023 Accepted: 8 May 2023
Published online: 27 May 2023

References

1. Balaban S. Deep learning and face recognition: the state of the art. In: Biometric and Surveillance Technology for Human and Activity Identification XII. 2015; 9457
2. Wu H, Prasad S. Semi-supervised deep learning using pseudo labels for hyperspectral image classification. IEEE Trans Image Process. 2017;27(3):1259–70.
3. Berry RB, Brooks R, Gamaldo CE, Harding SM, Marcus C, Vaughn BV. The AASM manual for the scoring of sleep and associated events: rules, terminology and technical specifications. Am Acad Sleep Med. 2012; 176.
4. Diykh M, Li Y, Wen P. EEG sleep stages classification based on time domain features and structural graph similarity. IEEE Trans Neural Syst Rehabil Eng. 2016;24(11):1159–68.
5. Munk AM, Olesen KV, Gangstad SW, Hansen LK. Semi-supervised sleep-stage scoring based on single channel EEG. In: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing. 2018; pp. 2551–2555.
6. Wuzheng X, Zuo S, Yao L, Zhao X. Semi-supervised sparse representation classification for sleep EEG recognition with imbalanced sample sets. J Mech Med Biol. 2021;21(5):2140006–1214000613.
7. Bai H, Lu G. Semi-supervised end-to-end automatic sleep stage classification based on pseudo-label. In: 2021 IEEE International Conference on Power Electronics, Computer Applications. 2021; pp. 83–87. https://doi.org/10.1109/ICPECA51329.2021.9362521.
8. Zhao R, Xia Y, Zhang Y. Unsupervised sleep staging system based on domain adaptation. Biomed Signal Process Control. 2021;69:1–9. https://doi.org/10.1016/j.bspc.2021.102937.
9. Jadhav P, Rajguru G, Datta D, Mukhopadhyay S. Automatic sleep stage classification using time-frequency images of CWT and transfer learning using convolution neural network. Biocybernet Biomed Eng. 2020;40(1):494–504.
10. Abdollahpour M, Rezaii TY, Farzamnia A, Saad I. Transfer learning convolutional neural network for sleep stage classification using two-stage data fusion framework. IEEE Access. 2020;8:180618–32.
11. Andreotti F, Phan H, Cooray N, Lo C, Hu MT, De Vos M. Multichannel sleep stage classification and transfer learning using convolutional neural networks. In: 2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society. 2018; pp. 171–174.
12. Radha M, Fonseca P, Moreau A, Ross M, Cerny A, Anderer P, Long X, Aarts RM. A deep transfer learning approach for wearable sleep stage classification with photoplethysmography. NPJ Digital Med. 2021;4(1):1–11.
13. Zhu G, Li Y, Wen P. Analysis and classification of sleep stages based on difference visibility graphs from a single-channel EEG signal. IEEE J Biomed Health Inform. 2014;18(6):1813–21.
14. Alickovic E, Subasi A. Ensemble SVM method for automatic sleep stage classification. IEEE Trans Instrum Meas. 2018;67(6):1258–65.
15. Jain VP, Mytri V, Shete V, Shiragapur B. Sleep stages classification using wavelettransform and neural network. In: Proceedings of 2012 IEEE-EMBS International Conference on Biomedical and Health Informatics. 2012; pp. 71–74.
16. Sokolovsky M, Guerrero F, Paisarnsrisomsuk S, Ruiz C, Alvarez SA. Deep learning for automated feature discovery and classification of sleep stages. IEEE/ACM Trans Comput Biol Bioinf. 2019;17(6):1835–45.

17. Supratak A, Guo Y. TinySleepNet: An efficient deep learning model for sleep stage scoring based on raw single-channel EEG. In: 2020 42nd Annual International Conference of the IEEE Engineering in Medicine & Biology Society. 2020; pp. 641–644.
18. Jia Z, Lin Y, Wang J, Ning X, He Y, Zhou R, Zhou Y, Lehman L-WH. Multi-view spatial-temporal graph convolutional networks with domain generalization for sleep stage classification. IEEE Trans Neural Syst Rehabil Eng. 2021;29:1977–86.
19. Fraiwan L, Lweesy K, Khasawneh N, Fraiwan M, Wenz H, Dickhaus H. Classification of sleep stages using multi-wavelet time frequency entropy and LDA. Methods Inf Med. 2010;49(3):230–7.
20. Zhang L, Xiong J, Zhao H, Hong H, Zhu X, Li C. Sleep stages classification by CW Doppler radar using bagged trees algorithm. In: 2017 IEEE Radar Conference. 2017; pp. 788–791.
21. Rodríguez-Sotelo JL, Osorio-Forero A, Jiménez-Rodríguez A, Cuesta-Frau D, Cirugeda-Roldán E, Peluffo D. Automatic sleep stages classification using EEG entropy features and unsupervised pattern analysis techniques. Entropy. 2014;16(12):6573–89.
22. Li Y, Peng C, Zhang Y, Zhang Y, Lo B. Adversarial learning for semi-supervised pediatric sleep staging with single-EEG channel. Methods. 2022;204:84–91.
23. Li Y, Luo S, Zhang H, Zhang Y, Zhang Y, Lo B. MtCLSS: multi-task contrastive learning for semi-supervised pediatric sleep staging. IEEE J Biomed Health Informat.
24. Wang J, Lan C, Liu C, Ouyang Y, Qin T, Lu W, Chen Y, Zeng W, Yu P. Generalizing to unseen domains: a survey on domain generalization. IEEE Trans Knowl Data Eng. 2022
25. Zhou K, Liu Z, Qiao Y, Xiang T, Loy CC. Domain generalization: a survey. IEEE Trans Pattern Anal Mach Intell. 2022
26. Shankar S, Piratla V, Chakrabarti S, Chaudhuri S, Jyothi P, Sarawagi S. Generalizing across domains via cross-gradient training. Int Conf Learn Represent. 2018
27. Volpi R, Namkoong H, Sener O, Duchi JC, Murino V, Savarese S. Generalizing to unseen domains via adversarial data augmentation. Adv Neural Informat Process Syst. 2018; 31.
28. Zhou K, Yang Y, Hospedales T, Xiang T. Deep domain-adversarial image generation for domain generalisation. In: Proceedings of the AAAI Conference on Artificial Intelligence. 2020; vol. 34, pp. 13025–13032.
29. Kingma DP, Welling M. Auto-encoding variational bayes. 2013; arXiv preprint arXiv:1312.6114.
30. Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y. Generative adversarial networks. Commun ACM. 2020;63(11):139–44.
31. Zhang H, Cisse M, Dauphin YN, Lopez-Paz D. Mixup: beyond empirical risk minimization. International Conference on Learning Representations. 2018.
32. Blanchard G, Lee G, Scott C. Generalizing from several related classification tasks to a new unlabeled sample. Adv Neural Informat Process Syst. 2011; 24.
33. Grubinger T, Birlutiu A, Schöner H, Natschläger T, Heskes T. Domain generalization based on transfer component analysis. In: International Work-Conference on Artificial Neural Networks. 2015; pp. 325–334 . Springer.
34. Hu S, Zhang K, Chen Z, Chan L. Domain generalization via multidomain discriminant analysis. In: Uncertainty in Artificial Intelligence. 2020; pp. 292–302. PMLR.
35. Li H, Pan SJ, Wang S, Kot AC. Domain generalization with adversarial feature learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018; pp. 5400–5409.
36. Ganin Y, Lempitsky V. Unsupervised domain adaptation by backpropagation. In: International Conference on Machine Learning. 2015; pp. 1180–1189 . PMLR.
37. Li Y, Tian X, Gong M, Liu Y, Liu T, Zhang K, Tao D. Deep domain generalization via conditional invariant adversarial networks. In: Proceedings of the European Conference on Computer Vision. 2018; pp. 624–639.
38. Zhou F, Jiang Z, Shui C, Wang B, Chaib-draa B. Domain generalization with optimal transport and metric learning. 2020; arXiv preprint arXiv:2007.10573.
39. Jin X, Lan C, Zeng W, Chen Z, Zhang L. Style normalization and restitution for generalizable person re-identification. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020; pp. 3143–3152.
40. Pan SJ, Tsang IW, Kwok JT, Yang Q. Domain adaptation via transfer component analysis. IEEE Trans Neural Networks. 2010;22(2):199–210.
41. Mancini M, Bulo SR, Caputo B, Ricci E. Best sources forward: domain generalization through source-specific nets. In: 2018 25th IEEE International Conference on Image Processing. 2018; pp. 1353–1357. IEEE.
42. Segu M, Tonioni A, Tombari F. Batch normalization embeddings for deep domain generalization. Pattern Recogn. 2023;135: 109115.
43. Li D, Yang Y, Song Y-Z, Hospedales T. Learning to generalize: meta-learning for domain generalization. In: Proceedings of the AAAI Conference on Artificial Intelligence. 2018; 32.
44. Balaji Y, Sankaranarayanan S, Chellappa R. Metareg: Towards domain generalization using meta-regularization. Adv Neural Informat Process Syst. 2018; 31.
45. Li Y, Yang Y, Zhou W, Hospedales T. Feature-critic networks for heterogeneous domain generalization. In: International Conference on Machine Learning. 2019; pp. 3915–3924.
46. Du Y, Xu J, Xiong H, Qiu Q, Zhen X, Snoek CG, Shao L. Learning to learn with variational information bottleneck for domain generalization. In: European Conference on Computer Vision. 2020; pp. 200–216.
47. Chen K, Zhuang D, Chang JM. Discriminative adversarial domain generalization with meta-learning based cross-domain validation. Neurocomputing. 2022;467:418–26.
48. Carlucci FM, D'Innocente A, Bucci S, Caputo B, Tommasi T. Domain generalization by solving jigsaw puzzles. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2019; pp. 2229–2238.
49. Li D, Zhang J, Yang Y, Liu C, Song Y-Z, Hospedales TM. Episodic training for domain generalization. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. 2019; pp. 1446–1455.
50. Huang Z, Wang H, Xing EP, Huang D. Self-challenging improves cross-domain generalization. In: European Conference on Computer Vision. 2020; pp. 124–140.
51. Gasser T, Bächer P, Möcks J. Transformations towards the normal distribution of broad band spectral parameters of the EEG. Electroencephalogr Clin Neurophysiol. 1982;53(1):119–24.

52. Boyd JC, Lacher DA. A multi-stage gaussian transformation algorithm for clinical laboratory data. Clin Chem. 1982;28(8):1735–41.
53. Supratak A, Dong H, Wu C, Guo Y. Deepsleepnet: a model for automatic sleep stage scoring based on raw single-channel EEG. IEEE Trans Neural Syst Rehabil Eng. 2017;25(11):1998–2008.
54. Prerau MJ, Brown RE, Bianchi MT, Ellenbogen JM, Purdon PL. Sleep neurophysiological dynamics through the lens of multitaper spectral analysis. Physiology. 2017;32(1):60–92.
55. Kemp B, Zwinderman AH, Tuk B, Kamphuisen HAC, Oberye JJL. Analysis of a sleep-dependent neuronal feedback loop: the slow-wave microcontinuity of the EEG. IEEE Trans Biomed Eng. 2000;47(9):1185–94. https://doi.org/10.1109/10.867928.
56. Kingma DP, Ba J. Adam: A method for stochastic optimization. 2014; arXiv preprint arXiv:1412.6980.

Publisher's Note