



Capturing the Development of Internal Representations in a High-Performing Deep Network for Sleep Stage Classification

Sarun Paisarnsrissomsuk¹ · Carolina Ruiz¹ · Sergio A. Alvarez²

Received: 16 December 2020 / Accepted: 10 May 2021 / Published online: 3 June 2021
© The Author(s) 2021

Abstract

Deep neural networks can provide accurate automated classification of human sleep signals into sleep stages that enables more effective diagnosis and treatment of sleep disorders. We develop a deep convolutional neural network (CNN) that attains state-of-the-art sleep stage classification performance on input data consisting of human sleep EEG and EOG signals. Nested cross-validation is used for optimal model selection and reliable estimation of out-of-sample classification performance. The resulting network attains a classification accuracy of $84.50 \pm 0.13\%$; its performance exceeds human expert inter-scorer agreement, even on single-channel EEG input data, therefore providing more objective and consistent labeling than human experts demonstrate as a group. We focus on analyzing the learned internal data representations of our network, with the aim of understanding the development of class differentiation ability across the layers of processing units, as a function of layer depth. We approach this problem visually, using t-Stochastic Neighbor Embedding (t-SNE), and propose a pooling variant of Centered Kernel Alignment (CKA) that provides an objective quantitative measure of the development of sleep stage specialization and differentiation with layer depth. The results reveal a monotonic progression of both of these sleep stage modeling abilities as layer depth increases.

Keywords Sleep · Deep learning · Learning representations · Kernel similarity · Feature visualization

Introduction

Sleep disorders affect both the duration and quality of sleep, negatively impacting neuro-cognitive function, as well as overall health and quality of life of the affected individual [17]. At a societal level, sleep disorders lead to enormous costs due to reduced productivity and increased public health

expenditures associated with various comorbidities [10]. It is, therefore, in the best interests of individuals and society to facilitate the identification of sleep disorders with a view toward providing prompt and effective treatment.

The diagnosis of sleep disorders relies on the process of sleep stage scoring, which maps human sleep physiological signals to sequences of symbols that correspond to the different stages of sleep. Sleep stage scoring has traditionally been performed by highly trained human experts who visually detect patterns associated with individual sleep stages and make classification decisions based on guidelines such as those from the American Academy of Sleep Medicine (AASM) [4]. Automated sleep stage scoring can more efficiently and consistently provide accurate results than human scorers.

In this paper, we report a convolutional neural network architecture for performing sleep stage classification from electroencephalography (EEG) and electrooculography (EOG) human sleep signals. Sleep stage classification or scoring is the process of classifying each 30-s sleep period (known as a sleep epoch) as one of five classes: Sleep Stages 1, 2, 3, REM (Rapid Eye Movement), or Wake. We use

This article is part of the topical collection “AI and Deep Learning Trends in Healthcare” guest edited by KC Santosh, Paolo Soda and Zalelam Temesgen.

✉ Sergio A. Alvarez
alvarez@bc.edu

Sarun Paisarnsrissomsuk
spaisarnsrissomsu@wpi.edu

Carolina Ruiz
ruiz@wpi.edu

¹ Department of Computer Science, Worcester Polytechnic Institute, Worcester, MA 01609, USA

² Department of Computer Science, Boston College, Chestnut Hill, MA 02467, USA

nested cross-validation to select the best configuration from among network architectures of different widths, depths, and layer structures, and to evaluate performance of the networks on unseen data. We analyze the impact that the amount of waking data included in training has on the models' generalization.

A main focus of the paper is on understanding the internal representations of the data that develop within the network during learning. We apply t-distributed Stochastic Neighbor Embedding (t-SNE) [48] to the internal responses of the trained network to visualize how sleep stage categorization ability of the network layers improves with increasing depth. Adapting a technique first introduced in [23], we use Centered Kernel Alignment (CKA) [11] to quantify sleep specialization of the network's internal layers, and to validate the t-SNE visualization results. We propose the use of an additional pooling step prior to CKA to reduce the dimensionality of the network layer representations, providing an effective quantitative measure of sleep stage specialization that can be used to track internal feature development. This paper substantially extends the earlier version that appeared as [31]; in particular, all of the CKA-related work is new.

Related Work

A number of related works use the same dataset as in the current paper, the SleepEDF data set from the Physionet database [16]. That database comprises raw data from 20 patients, across three signal channels (two EEG, one EOG), together with polysomnograms that provide the sleep stage labels for the raw data in 30-s epochs. Most related work trains models in one of two settings, corresponding to different preprocessing decisions regarding what time period is retained in each instance of the SleepEDF data: (1) Sleep periods only; or (2) sleep periods plus 30 min of Wake data before and after sleep. We describe these two settings in the paragraphs below. Other papers use a different version or signal choice for the SleepEDF data [22, 25], greater recording length [19, 20], merged sleep stages [8], or a different dataset altogether [5, 7, 34, 42, 43], preventing direct comparison with this paper.

Setting 1: Sleep Periods Only. The first of the two settings mentioned above uses only data from sleeping periods (in-bed, after sleep onset). Tsinalis, Matthews, and Guo [46] applied ensemble learning in this setting, using an ensemble of stacked sparse autoencoders on hand-picked time-frequency analysis features extracted from the raw signals, and achieved 78% accuracy over single-channel EEG data. Tsinalis, Matthews, Guo, and Zafeiriou [47] used a convolutional neural network and reported 74% accuracy, again using a single EEG channel. Our previous work [33, 40, 41] in this setting used data from the three signal channels.

Setting 2: Sleep Periods Plus 30 min of Wake Data Before and After Sleep. The second setting in previous work includes 30 min of Wake data before and after sleep, in addition to sleep periods. Supratak, Dong, Wu, and Guo [45] implemented a deep learning model in this setting with bidirectional Long Short-Term Memory (LSTM) and obtained 78% accuracy on single-channel EEG data. Mousavi, Afghah, and Acharya [29] employed a sequence-to-sequence model that yielded 84.26% accuracy on single-channel EEG data. Phan et al. [36] trained attention-based recurrent neural networks in both settings; the accuracy was reported as 79.1% in the first setting and 82.5% in the second.

Our previous work [33, 40, 41] considers setting 1 described above. We developed a deep convolutional neural network (CNN) for automated sleep stage scoring. That architecture achieved a classification accuracy of 81% over polysomnographic data corresponding to sleep periods only. The cumulative confusion matrix showed that the performance of the model in scoring sleep stages 1 (S1) and Wake, which are under-represented classes, is not as good as for the other classes, namely sleep stages 2 (S2), 3 (S3), and REM.

The present paper considers setting 2. We carry out a more thorough model selection that considers a greater number and variety of architectural hyperparameter types and values, employing nested cross-validation, with the aim of improving out-of-sample predictive performance for mid-sleep wakefulness episodes and light sleep stage S1. We compare our results with the state of the art, and with published studies on the performance of human scorers. We investigate how interpolating between the data from the two settings by including varying amounts of pre-sleep and post-sleep wake data, affects model performance. We go on to study the development of internal representations across the layers of our network both visually and quantitatively, as outlined below in "Contributions".

Contributions

We present in the present paper an optimized deep convolutional neural network architecture for sleep stage scoring, and evaluate its out-of-sample performance. The classification accuracy of our model architecture exceeds human inter-scorer agreement [38], and is comparable with recent work over the same dataset [29].

We investigate the role of including additional wake data in improving the model's classification performance. We find that waking epochs before and after sleep help to improve prediction of waking epochs in the middle of sleep slightly, and are considerably easier to label accurately than mid-sleep waking epochs. We visualize the responses of models' internal units using t-SNE, more clearly showing their ability to differentiate between sleep stages as layer

depth increases, as compared with activation maximization approaches.

Additionally, we measure the sleep stage modeling ability of the network's internal units and layers using our adaptation of the centered kernel alignment (CKA) approach of [23]; namely, we use max pooling prior to CKA to reduce data dimensionality without significantly altering the CKA results. Our max-pooling CKA approach yields quantitative descriptions of per-stage specialization of units and layers, and of improvements in the network's ability to differentiate among sleep stages as layer depth increases, that can be computed effectively on high-dimensional data. The max-pooling CKA results validate the t-SNE visualization results quantitatively, and allow detection of instances in which the visual information alone can be misleading.

Background

Human Sleep

During sleep, the human body undergoes cyclical physiological changes, usually described in terms of a small number of sleep stages such as those in the AASM standard [4]: Wake (as brief periods of wakefulness can occur during sleep); REM (rapid eye movement); and three non-REM stages that we will refer to by the terms S1 (light sleep); S2; S3 (deep sleep). The notations N1, N2, N3 also occur in the literature. We will sometimes abbreviate the Wake stage as W and the REM stage as R, especially in contexts in which space is at a premium (some figures, in particular).

Certain overall features are typically observed in the architecture of sleep, that is, in the overall pattern of alternation among stages during the course of the night. For example, sleep stage progression typically exhibits somewhat cyclical behavior, in periods of 90 min or so. Cycles evolve during sleep: deep sleep is more prominent earlier in the night, and an alternation between REM and sleep stage S2 is observed later in the night [6]. To diagnose patients with sleep disorders, doctors rely on the appearance of unusual patterns in the sleep cycle. To capture the sleep cycle, physiological signals are first measured by body sensors during all-night sleep in polysomnography (PSG), including electroencephalograms (EEG), electrooculograms (EOG), electrocardiograms (ECG), and electromyograms (EMG).

The polysomnogram is mapped to a sequence of sleep stages in the process of sleep stage scoring, also known as sleep staging. In staging, a sleep expert splits signals into 30-s segments, called epochs, and examines those segments for the presence of patterns known to be associated with particular sleep stages. For example, based on the AASM standard [4], sleep spindles, low-amplitude 11–16 Hz bursts of 0.5–2 s duration, are common in sleep stage S2; higher

amplitude low-frequency (delta band) waves in sleep stage S3; and sawtooth waves with 2–6 Hz content in stage REM. The process of sleep scoring by human experts is tedious and error-prone. A major study estimates the mean rate of inter-scorer agreement to be 82.6% [38]. Another study suggests that agreement might be lower than 80% in some cases, and that differentiating between sleep and wakefulness is especially challenging [50].

Activation Vectors

To measure the development of internal features in deep networks in an objective, quantitative manner (as a complement to visual techniques such as [51]), we consider the responses, or activations, of the internal units of the network over the input data. These responses are known as *activation vectors*. Formally, if p is an internal network unit (filter or neuron), and if $X = [x_1, x_2, x_3, \dots, x_n]^T$ is a set of n data instances, then the activation vector of p over X is the vector a_p that consists of the activation values $a_p(x_i)$ of unit p over each of the n data instances x_i , as in Eq. (1):

$$a_p := a_p(X) = [a_p(x_1), a_p(x_2), a_p(x_3), \dots, a_p(x_n)]^T. \quad (1)$$

The activation of a given network unit results from forward propagation of the input data through the units in earlier layers, generally reflecting a composition of affine transformations, such as convolutions, and nonlinear transformations, such as pooling and ReLU functions. We also consider activation vectors at the level of entire layers of units, by collecting together the activation vectors of the individual units of that layer as the columns of a matrix. If layer P consists of neurons $[p_1, p_2, \dots, p_m]$, then the corresponding set of activation vectors of layer P is the $n \times m$ matrix $a_P = [a_{p_1}, a_{p_2}, a_{p_3}, \dots, a_{p_m}]$.

A small number of recent works have studied ways to measure similarity between pairs of activation vectors. Wang et al. [49] viewed similarity between sets of neurons as a distance between subspaces spanned by the activation vectors of those neurons. Rather than directly compute a distance between two subspaces, Raghu et al. [37] assumed that the features from different layers may be associated with different rotations of the subspaces, and introduced the Singular Vector Canonical Correlation Analysis (SVCCA) algorithm to align the two subspaces as closely as possible. Kornblith et al. [23] discussed some undesirable properties of the SVCCA algorithm, and proposed using Centered Kernel Alignment (CKA) [11] for measuring similarity between feature subspaces in a way that better detects similarity of representations resulting from different initializations of the same network. We elaborate on the CKA technique in the following subsection.

Kernel Alignment

Kernel-based learning algorithms, such as support vector machines (SVMs) and kernel Fisher discriminant analysis (KFD), map data from the input space into a feature space and apply learning algorithms in the feature space. Rather than mapping data points into the feature space directly, the mapping is given implicitly by specifying a positive-definite *kernel function* that computes the inner products of all pairs of data points in the feature space, without the need for explicitly embedding these data points in the often much higher dimensional feature space. Kernel alignment [12] measures similarity between two kernel functions. Cortes et al. [11] described theoretical and empirical shortcomings of standard kernel alignment, and proposed Centered Kernel Alignment (CKA), which mean-shifts the kernel features before calculating kernel alignment.

CKA was originally introduced in [11] with the aim of providing a similarity metric for kernel learning. Recently, [23] suggested that CKA can be applied on two sets of activation vectors to measure the similarity of learned internal features in different layers of a neural network. In the present work, we use CKA with linear kernels. This is in keeping with previous work on measuring layer similarity described in “Activation Vectors” (e.g., [37]), which seeks linear transformations that align the subspaces spanned by the activation vectors.

Linear-kernel CKA between centered X and Y can be computed as in Eq. (2), where X and Y are the two sets of data representations being compared [23]

$$CKA(X, Y) = \frac{||Y^T X||^2}{(||X^T X|| \cdot ||Y^T Y||)}. \tag{2}$$

In the present context, X and Y can be activation vectors corresponding to two layers, as described at the beginning of

“Activation Vectors”, or they could be class labels or logits, as in “Feature Development Quantification”.

Methodology

This section describes the human sleep data used in this paper (“Dataset”); general features of the deep network architectures that we considered (“Model Architecture”); nested cross-validation for training, performance evaluation, and model selection (“Model Training, Selection, and Evaluation”); and the methods employed for visualization and quantification of internal feature development in the trained network (“Understanding Internal Feature Development”). Figure 1 summarizes the steps involved in key components of the work.

Dataset

We used data of 20 patients from the SleepEDF (expanded) database [21] in the Physionet repository [16]. For each patient but one, data comprise two nights of signals in EEG Fpz-Cz, EEG Pz-Oz, and EOG (horizontal) channels, sampled at 100 Hz, with hypnograms that provide the sleep stage for each 30-s epoch; the database includes only one night of data for the remaining patient. As described in [28], the physiological signals were recorded using cassette recorders with frequency response range from 0.5 to 100 Hz, and then were digitized at a sampling frequency of 100-Hz. Each hypnogram was scored by one of six human experts. The resulting distribution of sleep stage labels in the data is as shown in Table 1. We note that S1 is the least frequently occurring stage.

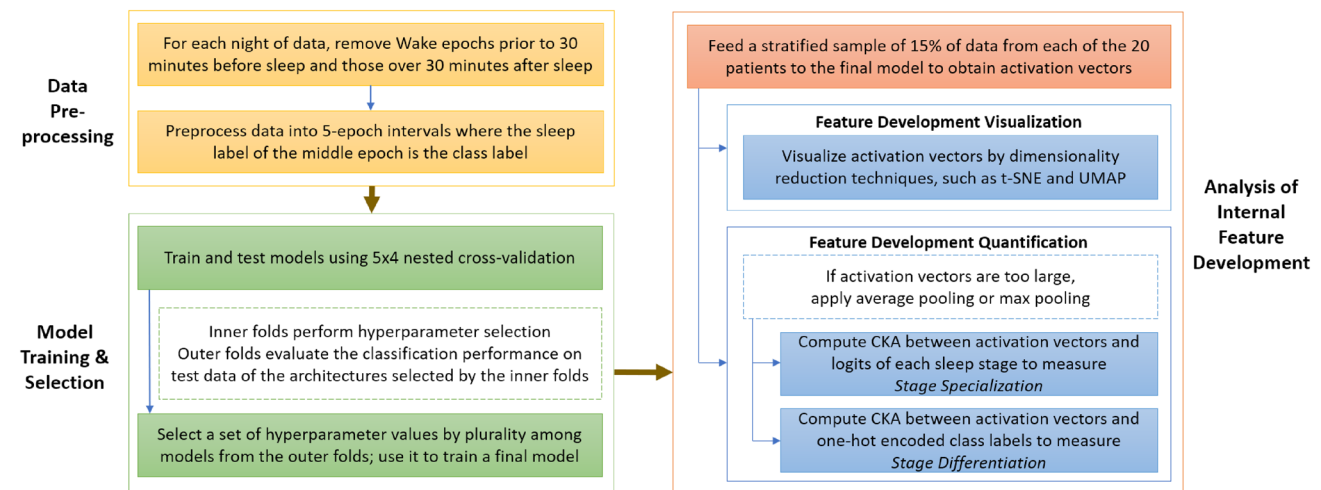


Fig. 1 Flow diagram of preprocessing; model training, evaluation, selection; and analysis of internal representations

As described in “Related Work”, for most of the work reported in this paper, we retain a limited amount of Wake data before and after sleep, removing Wake epochs prior to 30 min before the first observed non-Wake sleep epoch and those over 30 min after the end of sleep. In selected experiments discussed in “Performance on Wake”, we consider varying durations of this non-sleep data, to understand how the amount of such data affects model performance. To eliminate signal artifacts due to movement, we remove movement epochs entirely; there are only 62 such epochs in total, of which 60 occur during the sleep period.

When a human expert classifies or scores a given data epoch, information from neighboring epochs is also considered [4]. Hence, data in the present work were preprocessed into 5-epoch, or 150-s, samples; models were trained to classify the middle epoch as one of the five sleep stages, S1, S2, S3, Wake, and REM.

Model Architecture

The model architectures considered in this paper use 1-D convolutional layers stacked in stages separated by pooling layers, as shown in Fig. 2.

Each convolutional layer computes an affine transformation on the activation vector of the preceding layer; each such layer is followed by a regularization layer and an activation layer. Regularization layers are either batch normalization or dropout layers. Activation layers are ReLU linear rectifiers. At the deep end of the model, we add a dense layer with softmax activation to output a probability distribution over the five sleep stages. We use a single fully connected layer, because experiments in our prior work [33, 40, 41] suggest that a multi-layer fully connected structure does not enhance performance significantly.

We considered several alternative model architectures during model selection, described by the hyperparameters shown in Table 2. Salient points concerning hyperparameter options are described below. Details of the model selection procedure are discussed in “Model Training, Selection, and Evaluation”.

Comments on Hyperparameter Options

The number of filters (width) is assumed to double after each stage, not exceeding 128 due to computational resource

Table 1 Distribution of human expert-scored sleep stage labels in the SleepEDF data

S1	S2	S3	Wake	REM
6.63%	42.07%	13.48%	19.58%	18.24%

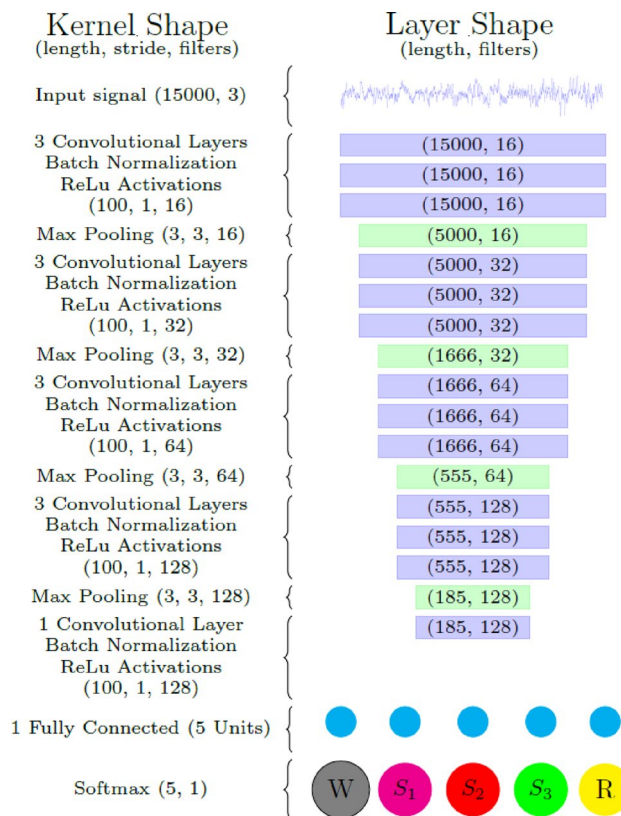


Fig. 2 Model architecture. The final model is constructed by selecting hyperparameter values by plurality among the outer folds of the nested cross-validation procedure described in “Model Training, Selection, and Evaluation”. Class values are Wake (W), sleep stages S1, S2, S3, and REM (R)

constraints. Thus, the number of filters in each stage is determined by the number of filters in the first stage.

We consider several options for the number of convolutional layers per stage: 6 layers in the first stage and 3 layers in other stages [33, 40, 41], a pattern that we denote

Table 2 Hyperparameters and values considered in model selection (inner folds of nested cross-validation)

Hyperparameters	Values considered
No. of filters in first stage	8, 16
Padding for conv layers	Yes, no
Stride of conv layers	1, 2, 4
Kernel sizes of conv layers	50, 100, 200, 400
Pooling sizes of pooling layers	2, 3, 5
Layer pattern in each stage	6C/P/3C/P, 3C/P, 1C/P
Batch norm or dropout	BN, dropout
Dropout layer at final layer	Yes, no
No. of conv layers	$[d - 3, d + 3]$ (see “Model Architecture”)

C, P denote convolutional, pooling layers, respectively

6C/P/3C/P; 3 convolutional layers per stage (3C/P); 1 convolutional layer per stage (1C/P).

For the number of convolutional layers (depth), we first calculate how the effective receptive field of the network units varies as a function of depth [2, 24].

A unit’s effective receptive field is the portion of the input window that affects that unit’s input, and thus its activation. From this information, the value, d , is extracted, such that the receptive field of the last layer covers the entire 5-sleep-epoch input; then, we consider depths in the range $[d - 3, d + 3]$.

Model Training, Selection, and Evaluation

Below, we first describe the training setup for a given model, and then the nested cross-validation procedure that performs model selection and performance evaluation. Finally, we describe the bootstrap sampling procedure that was used to obtain confidence intervals for accuracy and other performance metrics.

Training

For each model considered in the nested cross-validation procedure described below in “Model Selection and Evaluation”, we used stochastic gradient descent to minimize cross-entropy loss at the softmax output, with mini-batches of size 180, adaptive learning rate with initial value of 0.01, training time up to 300 epochs with early stopping threshold of 30 epochs. Models were built in Tensorflow [1] and Keras [9], under NVIDIA CUDA, using NVIDIA K80/P100 GPUs.

Weights were initialized according to [18], which adapts the “Xavier” approach of Glorot and Bengio [15] to the case of ReLU activation functions. Weights entering layer l were randomly sampled from a normal distribution with zero mean and standard deviation $\sqrt{\frac{2}{n_l}}$, where n_l is the effective fan-in, that is, the product of the number of input channels and the number of weights per filter in layer l .

Model Selection and Evaluation

Nested 5×4 cross-validation was used to select hyperparameter values, to train models, and to estimate their out-of-sample classification performance. An illustration of the nested cross-validation procedure appears in Fig. 3, showing the first (a) and second (b) of the five outer folds of the procedure.

In each of the 5 outer cross-validation folds, 4 patients’ data were reserved for (outer) testing; the remaining 16 patients’ data were used for training and hyperparameter selection. The inner level performed hyperparameter selection by dividing the 16 outer training patients’ data into an inner training set of 10 patients’ data, an inner validation set of 2 patients’ data, and an inner test set of 4 patients’ data.

For each fold of the inner loop, a greedy search of the hyperparameter space was performed, sequentially optimizing test performance over one hyperparameter at a time, fixing its value before adjusting the next hyperparameter; the search order considered hyperparameters with the smallest possible number of values first, to reduce the search space as little as possible at each step.

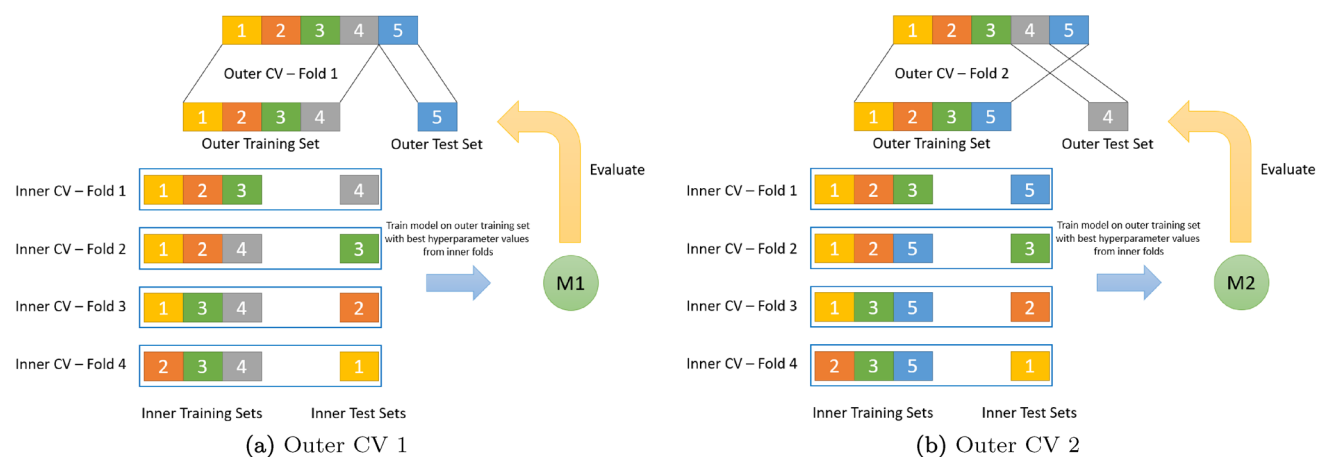


Fig. 3 Nested cross-validation is used for hyperparameter selection and for validation of performance on unseen data. We first randomly divide data into five parts. (a) In the first outer fold, we keep one of the five parts as a test set, and use the other four parts to find the best hyperparameter values by 4-fold cross-validation in the inner folds;

a new model is trained with those hyperparameter values on the four inner parts and evaluated on the test set. (b) In the second outer fold, we keep another part (out of five) as a test set and repeat the process. We do the same for the remaining three outer folds

Once the four folds of the inner loop had been completed, a model with the best-performing hyperparameter values from among the four inner folds was then trained on the 16 outer training patients' data, using 14 patients' data for training and 2 patients' data for validation; this model's performance was then evaluated over the outer test set of 4 patients' data reserved in the outer level. The entire process was repeated for each of the five outer folds.

Final Model. After nested cross-validation was complete, a single set of hyperparameter values was selected by plurality among those of the five best-performing models from the outer folds of the cross-validation procedure, and used to train a final model over the full set of all 20 patients. This final model is used throughout the experiments reported in this paper. Cross-validated estimates of the performance metrics for this final model were calculated by aggregating the confusion matrices of the five models used in the outer folds of the nested cross-validation procedure.

Boostrapped Performance Estimates

We used bootstrap sampling to compute confidence intervals of radius one standard error for overall classification accuracy. For per-stage performance, precision, recall, and F1 score were used instead, as the substantial class imbalance in the resulting per-stage binary classification tasks would yield misleadingly high accuracy values. We generated bootstrap samples by randomly selecting one of 20 patients, sampled the selected patient's data with replacement (without changing the total number of sleep epochs), and made class predictions using the model from the outer training fold that the patient was not in. The process was repeated 1000 times to produce an aggregate confusion matrix over all bootstrap samples.

Understanding Internal Feature Development

We aimed to understand the feature representation of the sleep data that arises within our deep network (see "[Model Architecture](#)") as a result of training. Sleep stage specialization is an aspect of internal feature development that is of particular interest; this term refers to the extent to which individual processing units, or layers of units, respond differently to input data epochs of different sleep stages.

We focused on approaches based on the objective quantification of the learned internal features via an analysis of the activation vectors of the units and layers of the network (see "[Activation Vectors](#)"). Visualization techniques were used to provide insight into sleep stage specialization information that is embedded in the activation vectors; thus, visualization was not used as a competing standalone approach, but rather as a means to extracting information that is rooted directly in these quantitative measurements. Additional quantitative

analysis of the activation vectors was carried out using Centered Kernel Alignment (CKA). We describe the complementary visual and quantitative elements that enter into describing internal feature development below.

Feature Development Visualization

We used t-distributed Stochastic Neighbor Embedding (t-SNE) [48] to visualize the trained network's internal activation levels via a low-dimensional embedding that preserves pairwise similarity under a Gaussian conditional model. The t-SNE perplexity hyperparameter was set to 30.

Since it is possible, in principle, that the nonlinearity inherent in t-SNE might distort the relationships among instances of different classes, and, in particular, that it might suggest greater or lesser class separation than actually occurs in the space of predictive attributes, we also carried out experiments in which we applied the linear technique of principal components analysis (PCA) to the activation vectors, which uses an orthogonal projection based on a transformation that preserves lengths and angles.

Additionally, we tested both multidimensional scaling (MDS) and UMAP [27] for visualization. Metric MDS was used. UMAP hyperparameters were set as follows: $n_neighbors = 15$, $min_dist = 0.1$. Sleep stage specialization of internal units was studied at both the level of individual units (filters), and of entire layers, using human sleep samples as network inputs. At the unit level, we obtained a unit's activation and directly applied t-SNE to embed the activation into 2-dimensions. At the layer level, we first reduced the dimensionality of the layer's response by taking the maximum value of each unit's activation along its time axis, and then applied t-SNE to the resulting vector of time-aggregated responses of all of the units in the layer. We explored alternatives to the maximum for time-aggregation, using the mean or median for pooling instead.

Feature Development Quantification

We applied Centered Kernel Alignment (CKA) [11] to measure the quality of the network's internal representations of sleep data. We did this by computing the following two measures: (1) *Stage Specialization*, a stage-specific measure that gauges the association between the learned representations of internal units with each stage, is calculated by applying CKA between an activation vector and logits of each stage; and (2) *Stage Differentiation*, an all-stage measure that gauges internal units' ability to distinguish among the various classes, is computed by applying CKA between an activation vector and one-hot encoded class labels. An illustration is presented in Fig. 4.

We have found that the dimensionality of the convolutional layer responses in shallow layers can be very large, so

that direct application of CKA would exceed the available computational resources. For example, the output of the first layer of our network, as shown in Fig. 2, has shape (15,000, 16), which is 240,000 elements when vectorized. Therefore, we performed dimensionality reduction on the activation vectors before applying CKA by grouping selected elements of the activation vectors along the time axis, taking their maximum or mean values. This is equivalent to applying a max pooling or averaging pooling layer to the activation vector with the stride equal to the kernel size. An appropriate pooling size can be selected, so that applying CKA becomes computationally feasible. In this work, for each activation vector, we dynamically picked a pooling size, such that the dimension-reduced vector is approximately a square matrix.

We tested whether using pooling before applying CKA significantly changes the results of CKA, as follows. We first selected two networks with identical architecture: the final model and the best model selected in the second outer fold of the nested cross-validation procedure (see Table 3); we will subsequently refer to the latter model as the outer2 model. The final model and the outer2 model differ slightly due to differences in initial parameter values, and due to the use of slightly different data sets: the final model was trained on the full data set, whereas the training data set for the outer2 model did not include the samples in the reserved outer2 fold. We applied CKA to calculate the layer similarity of pairs of selected layers whose activation vectors are small enough for CKA to be computed without using pooling. We compared the CKA

Table 3 Best hyperparameter set for each outer fold. Each column represents the best hyperparameter values for one outer fold of nested cross-validation procedure

Hyperparam.	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5
Initial filters	16	16	16	16	16
Padding	Yes	Yes	Yes	Yes	No
Stride	1	1	1	1	1
Kernel size	100	100	100	100	100
Pooling size	5	3	5	3	3
Pattern	6C/P/3C/P	3C/P	3C/P	3C/P	3C/P
BN/dropout	BN	BN	BN	BN	BN
Dropout final	No	No	No	0.1	No
# conv layers	13	13	10	13	13
Accuracy	83.44%	83.77%	85.32%	85.26%	85.04%

layer similarity values without using pooling, with those using max pooling prior to CKA, and with those using average pooling prior to CKA.

To select between max pooling and average pooling, we considered, as in [23, 37], the idea that for two networks with the same architecture, the same layers in the two networks should be similar. A comparison was performed using two models resulting from the model selection procedure, specifically, the best model from the outer2 fold and the final model. The winning pooling approach was then used in all subsequent CKA experiments.

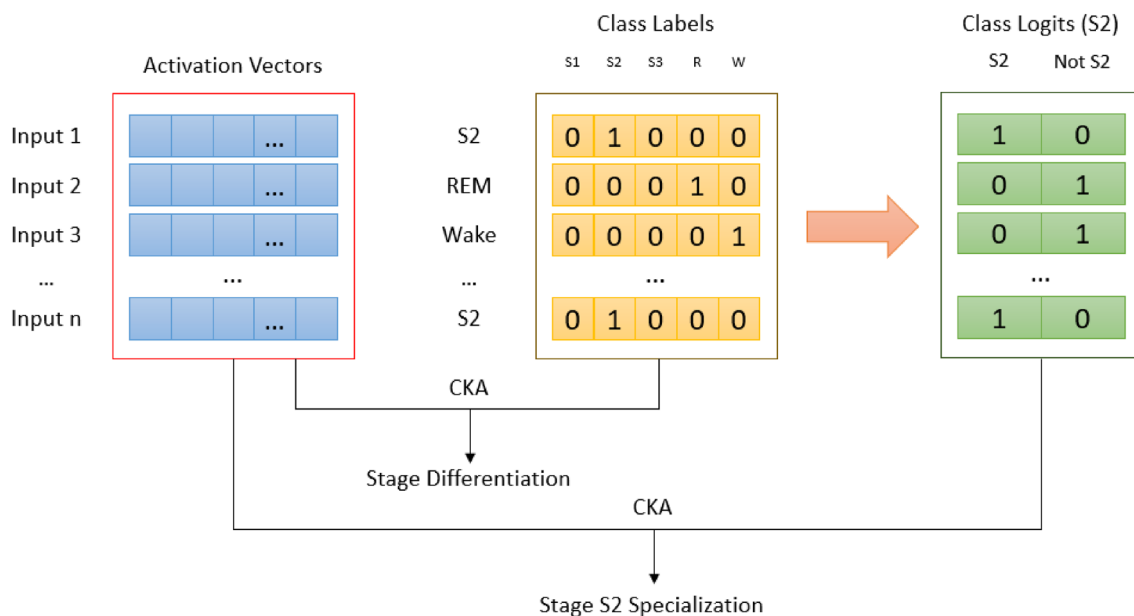


Fig. 4 Quality of the network’s modeling of sleep is captured quantitatively as *Stage Differentiation*, a measure of classification ability obtained by applying CKA between activation vectors and one-hot

encoded class labels; and as *Stage Specialization*, a stage-specific measure of the ability to identify the given stage (S2 shown here), obtained by applying CKA between activation vectors and class logits

Results

Model Selection

The best-performing set of hyperparameter values for each outer fold of the nested cross-validation model selection procedure (“[Model Training, Selection, and Evaluation](#)”) is shown in Table 3. We found that the best hyperparameters in different folds have many values in common. Padding provides better performance than no padding in all folds but one; in fold 5, not using padding yields a small margin in accuracy. Similarly, the 3C/P layer pattern wins in all folds except fold 1, in which 6C/P/3C/P is slightly better. No dropout after convolutional layers is best except for fold 4, in which dropout of 0.1 has a slim advantage. All folds select 13 convolutional layers as being best, except for fold 3, which uses 10 layers. Only pooling size differs more substantially among folds: in folds 2, 4, 5, a pooling size of 3 is best, while a size of 5 is better in folds 1, 3.

We constructed the final model by selecting hyperparameter values by plurality among the outer cross-validation folds. This yields a model with 13 convolutional layers in the 3C/P pattern, with a filter size of 100 and a stride of 1, regularization by batch normalization, and a

pooling size of 3. The final model was trained on the full dataset, using 17 patients’ data for training and the other 3 patients’ data for validation prior to t-SNE visualization of internal activations (“[Internal Feature Development](#)”). The classification performance reported in “[Model Performance](#)” is computed during model selection, as described in “[Model Training, Selection, and Evaluation](#)”.

Model Performance

Table 4(a) shows the cumulative confusion matrix of the five outer models on unseen patients. Overall accuracy is 84.57%. A bootstrap estimate of overall accuracy was also computed, yielding the confidence interval $84.50 \pm 0.13\%$, of radius one standard error around the bootstrapped mean accuracy of 84.50%. The predictive accuracy of our model comfortably exceeds published figures for human expert inter-scorer agreement of 82.6% [38]. Overall performance of our model also compares favorably with recent work over the same data set, such as [29] (84.26% accuracy on single-channel EEG data) and [36] (82.5% accuracy).

As mentioned in “[Model Training, Selection, and Evaluation](#)”, per-stage accuracy values would be misleadingly high. We therefore assessed per-stage performance using precision, recall, and F1 scores. Table 5 shows performance confidence intervals for each class spanning one standard error

Table 4 Performance of models trained on different numbers of input channels: (a) two EEG channels and one EOG channel, (b) two EEG channels, and (c) one EEG channel

	S1	S2	S3	Wake	R	Recall	F1 score
(a) Three channels (2 EEG + 1 EOG)							
S1	1287	647	17	501	352	45.90%	47.32%
S2	595	15,609	679	253	663	87.70%	87.34%
S3	3	652	5025	21	2	88.11%	87.78%
Wake	495	199	15	7473	102	90.21%	89.17%
R	255	837	10	229	6386	82.75%	83.90%
Precision	48.84%	86.99%	87.45%	88.16%	85.09%		
(b) Two EEG channels							
S1	1295	594	10	484	421	46.18%	45.49%
S2	670	15,377	537	457	758	86.39%	87.24%
S3	10	706	4897	79	11	85.87%	87.74%
Wake	525	135	14	7377	233	89.05%	86.29%
R	390	640	2	417	6268	81.22%	81.36%
Precision	44.81%	88.11%	89.69%	83.70%	81.50%		
(c) One EEG channel							
S1	1215	679	14	499	397	43.33%	44.61%
S2	482	15,516	1103	148	550	87.17%	86.57%
S3	10	656	4999	38	0	87.66%	83.10%
Wake	522	148	81	7440	93	89.81%	90.09%
R	414	1047	132	107	6017	77.97%	81.45%
Precision	45.97%	85.98%	78.99%	90.38%	85.26%		

Cumulative confusion matrices show precision, recall, and F1 scores. Each entry is the number of sleep data epochs of the stage with that row label that the model classifies as the column label

Table 5 Bootstrapped confidence intervals for performance of three-channel (2 EEG + 1 EOG) model on each sleep stage, in format mean \pm SE

Sleep stage	Precision	Recall	F1 score
S1	49.64 \pm 0.58%	45.49 \pm 0.60%	44.11 \pm 0.42%
S2	86.85 \pm 0.22%	87.51 \pm 0.24%	86.84 \pm 0.16%
S3	86.75 \pm 0.32%	88.23 \pm 0.23%	86.90 \pm 0.19%
Wake	88.48 \pm 0.28%	90.18 \pm 0.26%	88.70 \pm 0.15%
R	85.22 \pm 0.29%	83.72 \pm 0.34%	83.69 \pm 0.21%
Mean	79.39 \pm 0.16%	79.03 \pm 0.17%	78.05 \pm 0.15%

below and above the mean precision, recall, and F1 scores. Mean values differ slightly between Table 4(a) and Table 5, because bootstrapping was only used for the latter. Model performance on sleep stages S2, S3, Wake, and REM is very good. Performance is worst on the least frequent class, S1.

Dependence on the Number of Signal Channels

We trained two additional models: one that uses only the two EEG channels (no EOG), and another that uses only one EEG channel (Fpz-Cz, the best performing of the channels [33, 40, 41]). We did not redo hyperparameter selection, due to limited computational time. Overall accuracies of these models are 83.23% and 83.17%, respectively, both of which are still higher than human expert inter-scorer agreement of 82.6% [38].

Tables 4 (b) and (c) show the cumulative confusion matrices of the two models. Mean F1 values for the REM stage drop slightly in the absence of EOG information, from 83.9% for the three-channel model to 81.4% or so for both EEG-only models. This is not surprising, as eye movements are a defining feature of REM. Mean F1 score on Wake is likewise slightly lower for the two-channel model at 86.3%, as compared with the three-channel model's 89.2%. Notably, the model based on a single EEG channel outperforms the other two models on Wake, attaining a mean F1 value of 90.1% on that stage.

Performance on Wake

The performance on sleep stage Wake reported in this paper greatly improves on our previous work [33, 40, 41] in precision, recall, and F1 score. The F1 score attained by the three-channel model on sleep stage Wake has increased from 57% to 89% (see Table 4(a)). One source of this improvement is the greater number Wake epochs in the dataset. Whereas our previous work included only mid-sleep awakenings in the dataset, as in [46, 47], the present paper also includes 30 min of Wake epochs before and after sleep, as in [29, 45]. Huy Phan et al. [36] trained their models in both settings.

We wondered whether the improved performance on Wake that results from including pre- and post-sleep data is due to better generalization resulting from the greater amount of Wake data available for training, or whether perhaps the additional Wake epochs before and after sleep are easier to classify than the mid-sleep Wake epochs. As noted in "Dataset", we removed all movement epochs; hence, artifacts associated with such events are not responsible for the observed performance improvement. Aiming to understand this matter, we tested the models trained with 30-min Wake data before and after sleep on the mid-sleep data and found that F1 score on Wake is 65%, which is only 1% higher than the models trained on the mid-sleep epochs. This shows that including additional Wake data before and after sleep does not improve generalization nearly enough to account for the overall improvement described in the preceding paragraph.

We retrained our models on the mid-sleep data only. Due to computational limitations, we kept the original hyperparameters. We tested the models on a sequence of data sets, adding X minutes of Wake epochs before and after sleep. See Fig. 5. At X = 0 (mid-sleep epochs only), the F1 score on Wake is 64%. As X increases (more pre- and post-sleep Wake epochs), Wake performance improves, even though the additional data were not used in training. This suggests that the additional Wake epochs are different from Wake epochs during sleep, and are easier to predict (consistent with [36]), accounting for much of the performance improvement.

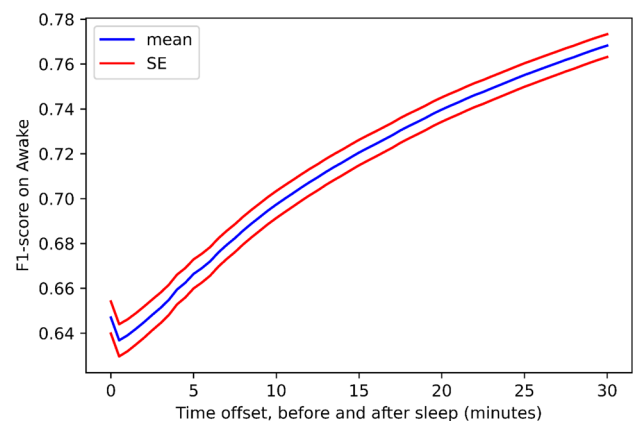


Fig. 5 F1 performance on Wake stage by three-channel models trained on mid-sleep data only. Means (blue) and mean \pm SE intervals (red) calculated from bootstrapped samples. Models make more accurate predictions on Wake data before and after sleep, leading to increased F1 as more pre- and post-sleep Wake data are included in the test set. Overall level of Wake performance is lower here than in Table 5 due to the use here of mid-sleep training data only

Internal Feature Development

Feature Development Visualization

We used visualization to study the responses to human sleep data of individual network filters and entire network layers. As described in “[Feature Development Visualization](#)”, we compared t-SNE visualization with PCA, MDS, and UMAP.

Visualization of whole-layer activations used pooling over time for dimensionality reduction, as explained in “[Feature Development Visualization](#)”. We compared the results obtained using the maximum, mean, and median as pooling operators. Using the mean led to results that are qualitatively similar to those obtained using the maximum; using the median led to noticeably poorer results. We used the maximum as the pooling operator for the remaining experiments.

PCA and t-SNE produced qualitatively similar results in terms of the degree of spatial separation that they suggest between classes. As an example, Fig. 6 shows similar differentiation among stages in the activation vectors associated with an early layer of our network. This implies that the inherent nonlinearity of t-SNE does not affect the resulting visual assessments of stage separation. MDS to some extent, and both t-SNE and UMAP especially, were found to provide better resolution than PCA for the more complex activation patterns that tend to emerge in deeper network layers. An example is shown in Fig. 7. Taking all of these results into account, we selected t-SNE for use in the work reported in this paper.

Figure 8 shows the t-SNE results for sample filters at different depths within the network. Filters in early layers,

(a)–(c), respond similarly to data instances of different stages, therefore providing poor differentiation among stages. For middle layers, (d)–(f), we see some separation among sleep stages in the t-SNE output. In deep layers, (g)–(i), some filters are seen to separate instances of a particular stage from those of other stages.

We find that the t-SNE results are consistent with our previous work [33], in which we examined sleep stage specialization of individual filters by stage-based activation maximization for those filters [14, 26, 30, 39]: the t-SNE visualization of a filter that activates maximally to a particular stage shows that the filter responds differently to the instances of that stage than to instances of other stages, as evidenced by spatial separation between the locations of the instances in the t-SNE plot. For example, in Fig. 8, the filter in (g) separates Wake instances (right) from other stages, the filter in (h) separates S2 instances (left) from other stages, and the filter in (i) separates S3 instances (top) from other stages; in each case, the stage that has been singled out is the one that most often maximizes the given filter’s activation.

Furthermore, t-SNE gives us additional information about a filter’s capabilities than does activation amplitude alone. For example, the t-SNE visualization in Fig. 9c shows that the filter in question differentiates among several sleep stages quite well, while activation maximization only identifies one of these stages. We are not arguing that activation visualization using t-SNE (or UMAP) is superior in absolute terms to activation maximization [14, 26, 30] or other techniques that aim to assign credit to particular aspects of the input based on activation [3, 39, 44], but rather that it more directly addresses the specific issue of stage differentiation.

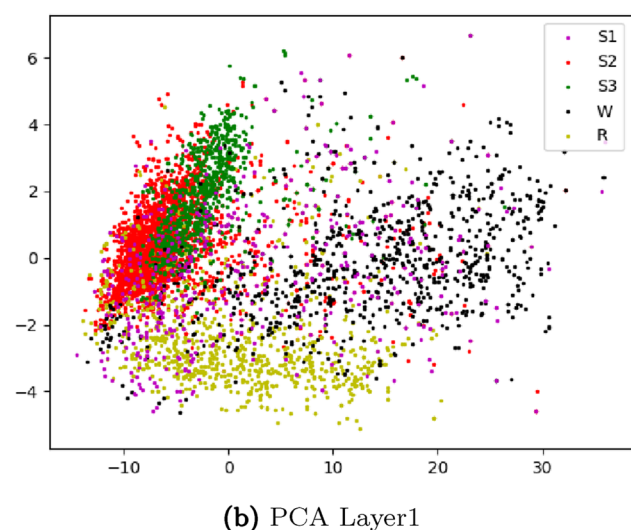
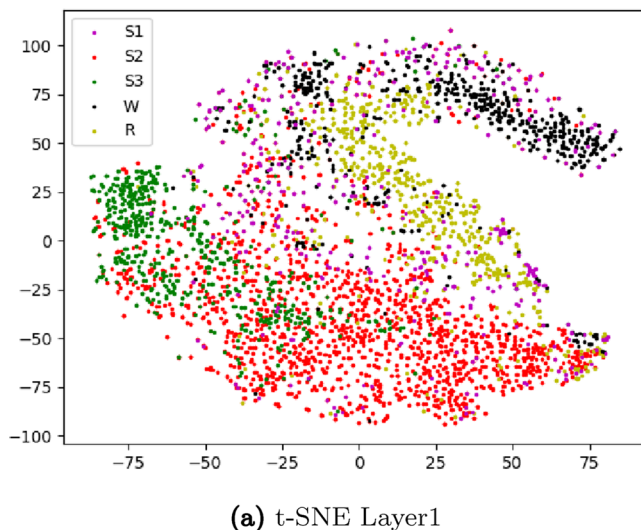


Fig. 6 Sample visualizations of whole-layer activation vectors in a shallow layer of the three-channel model using (a) t-SNE and (b) PCA. Stage labels coded by color. Axes correspond to the embedding coordinates, which are dependent on the visualization technique in

each case. Inputs to the models are random samples of 15% of data from each of the 20 patients. The two techniques suggest similar degrees of separation among stages in the given layer

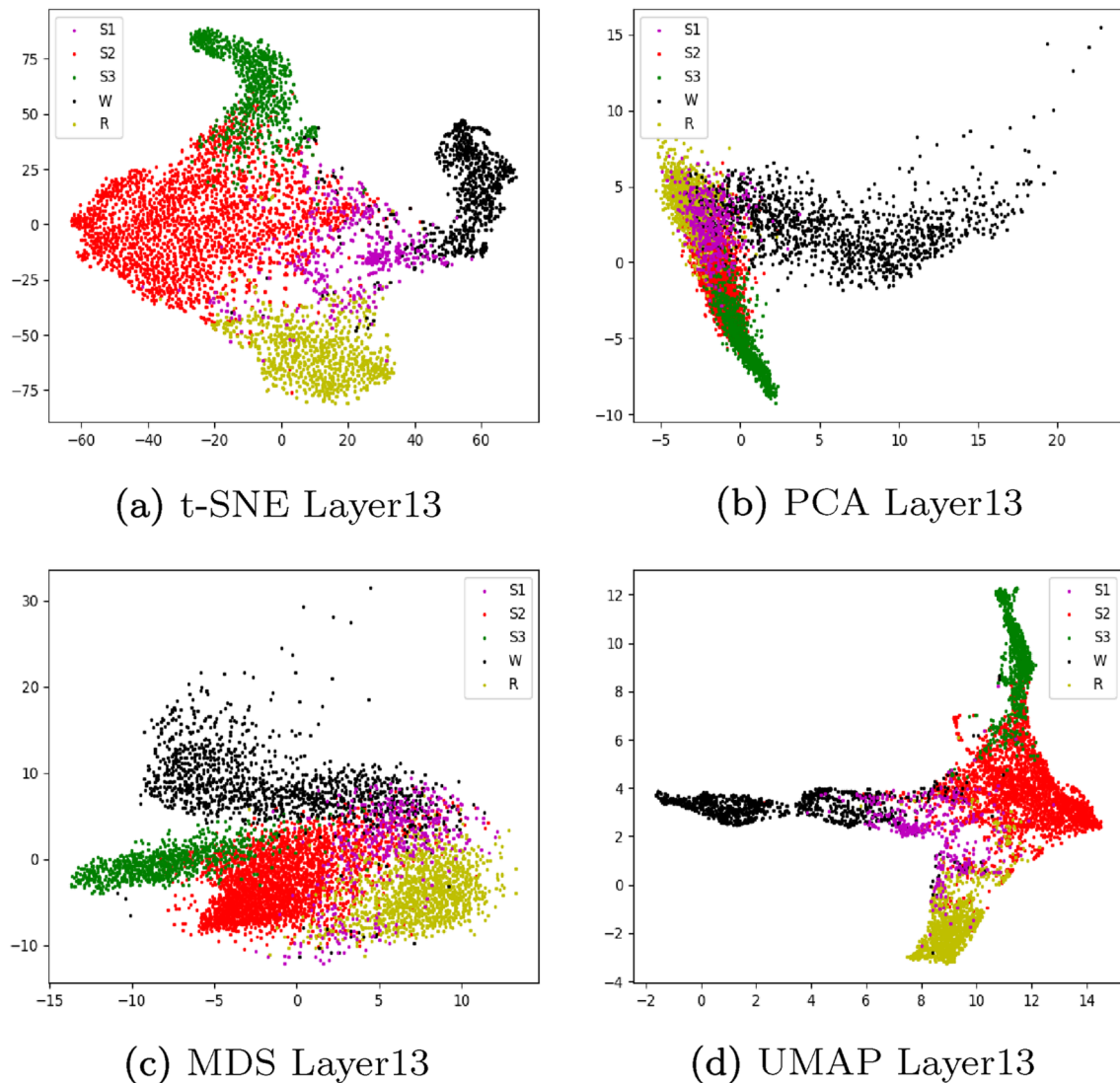


Fig. 7 Sample visualizations of whole-layer activation vectors in a deep layer of the three-channel model using (a) t-SNE, (b) PCA, (c) MDS, and (d) UMAP. Stage labels coded by color. Inputs to the mod-

els are random samples of 15% of data from each of the 20 patients. t-SNE, UMAP, and MDS show a more noticeable separation among stages than PCA does in the given layer

Activation maximization and related techniques are valuable tools in identifying input signal characteristics most closely associated with particular network units, and can provide information that complements both activation visualization and quantitative techniques such as CKA that constitute the focus of this paper.

We also applied t-SNE to the responses of all units within a given layer. Since the dimensionality of the responses of the whole layer can be very large, we first reduced dimensionality by taking the time-maximum of the response of each unit in the layer, before applying t-SNE to visualize the layer activations. Figure 10 shows the embedded activation plots for different layers. Fig. 10a Shows that the very first layer can differentiate between stages (Wake, REM)

and stages (S2, S3). Fig. 10b Shows a mid-depth layer that provides improved separation among sleep stages. Fig. 10c Shows a deep layer that differentiates all stages well.

EOG-Dependence of Wake and REM Performance

The t-SNE visualization of (Fig. 10a) suggests that very early (shallow) layers of our model are able to differentiate the pair of stages (Wake, REM) from the pair of stages (S2, S3). It is possible, in principle, that the early layers rely on eye movement information in the EOG channel to make this differentiation, as eye movements are more likely to occur in the Wake and REM stages. To investigate this possibility, we apply t-SNE to the responses of internal units of the first

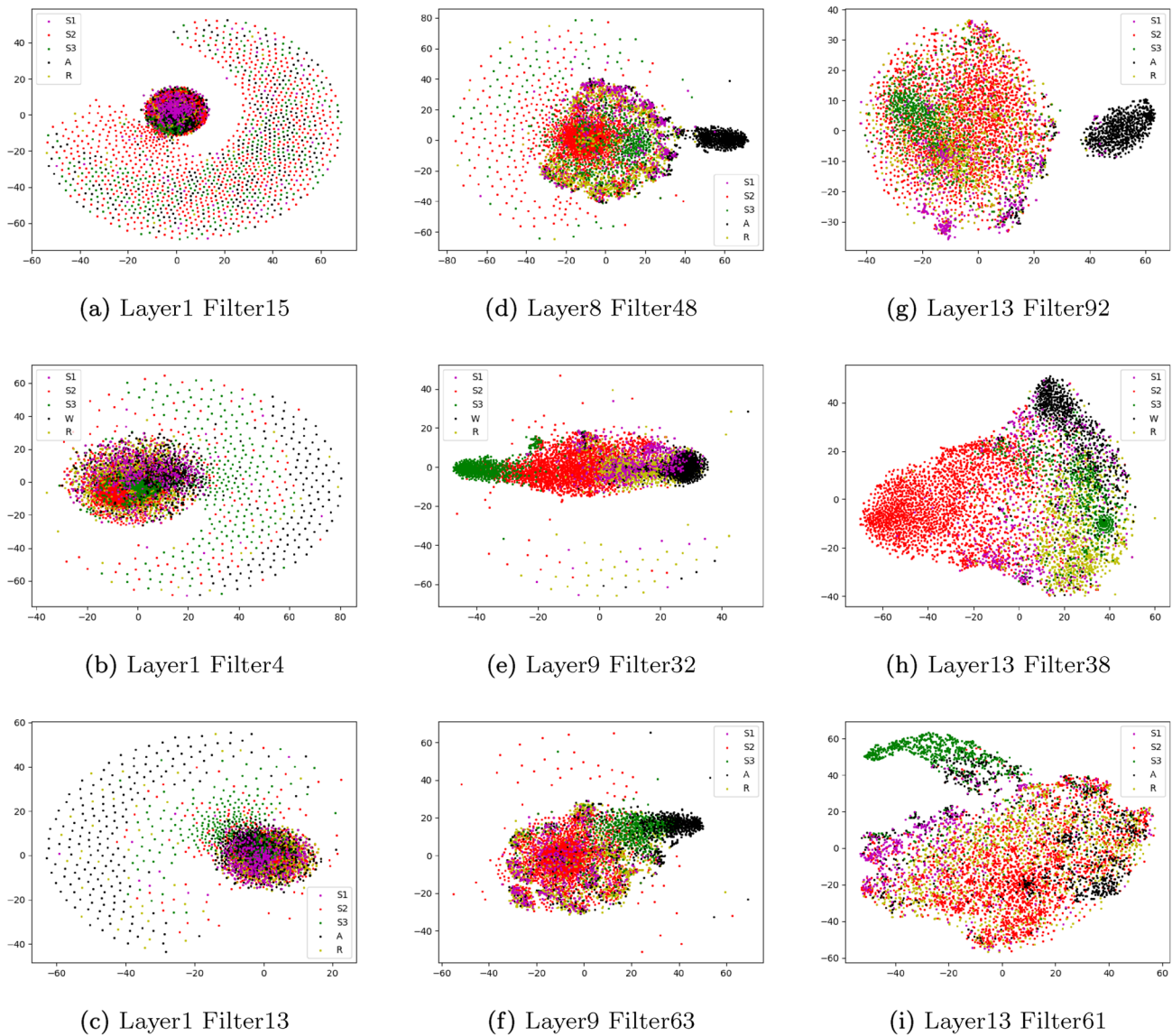


Fig. 8 Embedded t-SNE unit responses for the three-channel model. Inputs to the model are stratified samples of 15% of data from each of the 20 patients. Left column shows unit responses in shallow layers,

middle column shows unit responses in middle layers, and right column shows unit responses in deep layers. The results demonstrate an improvement in stage differentiation with increasing depth

layer for each of two networks trained, respectively, on two EEG channels and one EEG channel only (no EOG channel), as described in “[Dependence on the Number of Signal Channels](#)”. Figure 11b shows that the model trained on two EEG channels only can still differentiate stages (Wake, REM) from stages (S2, S3) in the very early layers; (c) shows that the model trained on a single EEG channel can still differentiate REM stage instances from those of other stages, though Wake is no longer well separated from other stages, in contrast with the model that uses both EEG channels. Our results suggest that the EEG-only networks begin to develop the ability to identify REM sleep very early, despite their not having access to the EOG channel.

Feature Development Quantification

We first tested the relative change that occurred in the CKA values as a result of either max-pooling or average-pooling dimensionality reduction as described in “[Feature Development Quantification](#)”. In all cases, the average pairwise absolute difference between the CKA values obtained without prior pooling of the data and those obtained after pooling was less than 0.015, which is much smaller than the range of variation of the CKA values overall, which varied from 0.76 to 0.96. The average percentage difference is less than 1.7%.

Next, we selected between max-pooling and average-pooling approaches by comparing the CKA values

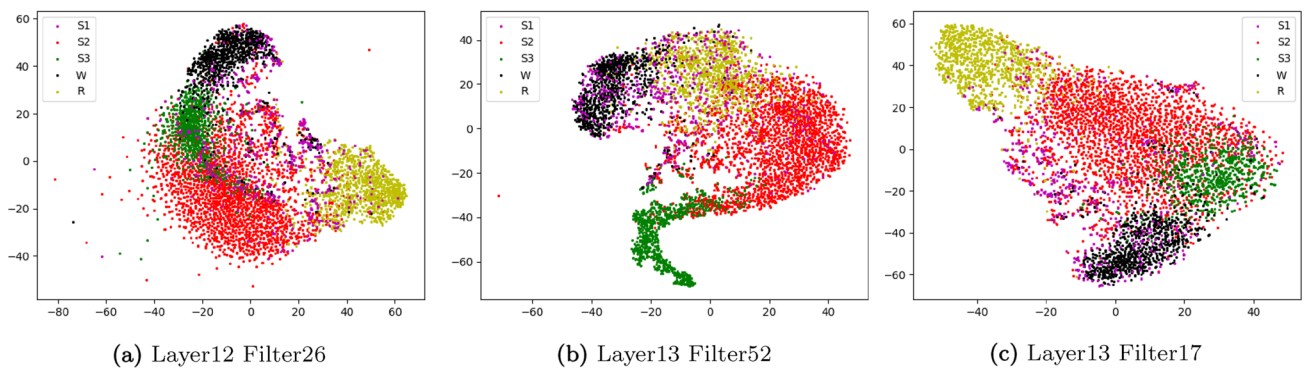


Fig. 9 t-SNE embeddings of units in deep layers of the three-channel model that activate strongly to a particular sleep stage (respectively, REM, S3, REM), showing these units' ability to also differentiate

among multiple stages. Model inputs are stratified samples of 15% of data from each of the 20 patients

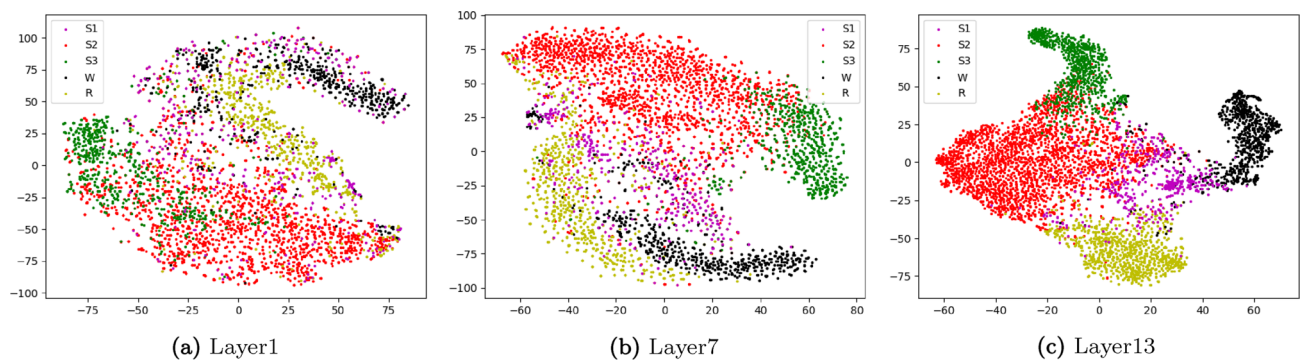


Fig. 10 Embedded layer responses in the three-channel model using t-SNE. The inputs to the final model are stratified samples of 15% of data from each of the 20 patients. The results show the improvement in the network's class differentiation ability with increasing depth

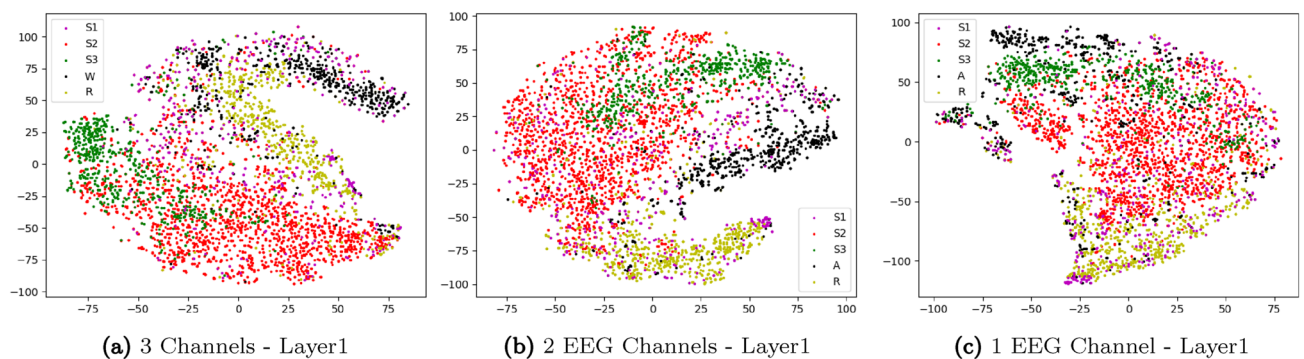


Fig. 11 t-SNE layer responses in models trained on (a) three channels (2 EEGs + 1 EOG), (b) two EEG channels, and (c) one EEG (Fpz-Cz) channel. Model inputs are stratified samples of 15% of data from

each of the 20 patients. The results show that separation of (Wake, REM) from (S2,S3) in shallow layers relies mainly not on EOG, but on both EEG channels

obtained after each of these pooling types. CKA was used to compare layers at equal depths in the final model and the best model from the outer2 fold of the nested cross-validation model selection procedure described in “[Model Training, Selection, and Evaluation](#)”.

Figure 12 shows the CKA similarity results of the two networks, using either max pooling prior to CKA (left) or average pooling prior to CKA (right). The results suggest that using max pooling is preferable to using average pooling, as the heat map on the left in Fig. 12, corresponding

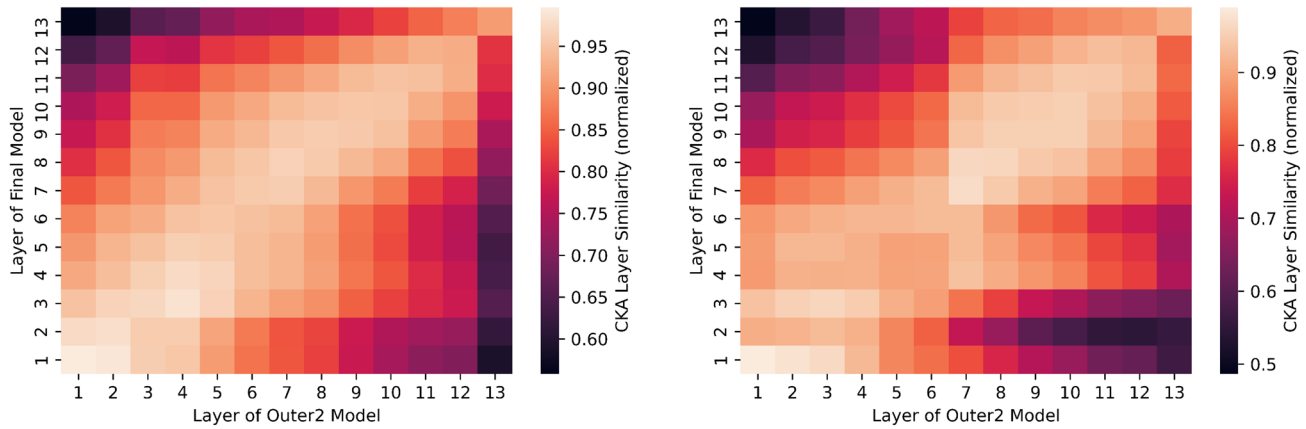


Fig. 12 Comparison of dimensionality reduction by max pooling (left) and average pooling (right) prior to layerwise CKA similarity measurement between slightly different initializations and training data samples for the same network architecture. Inputs to the mod-

els are stratified samples of 15% of data from each of the 20 patients. Max pooling shows stronger diagonal dominance, making it preferable to average pooling

to max pooling, shows higher similarity along the main diagonal than does the heat map on the right, which corresponds to average pooling. Based on this result, we selected max pooling for dimensionality reduction prior to CKA computation.

At the individual filter level, we used CKA between a filter’s response to samples of human sleep signals and logits of each sleep stage to measure the filter’s sleep stage specialization. Table 6 shows the resulting CKA quantification of filters’ stage specialization in different layers of the network, along with references to their corresponding t-SNE plots. The results show a low level of stage specialization in early network layers, and a higher level of specialization in deeper

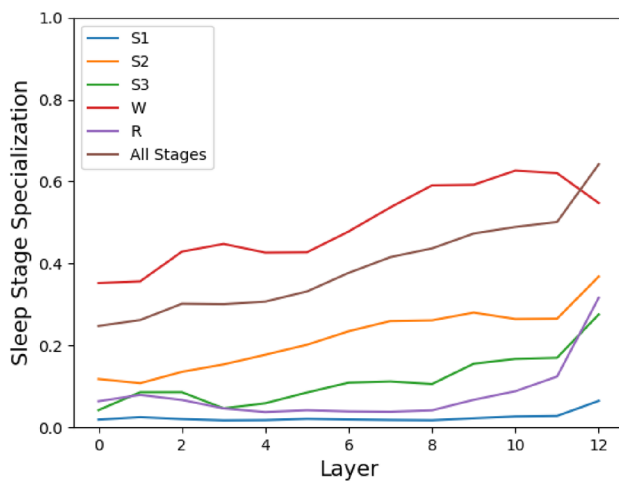
layers. The results are consistent with the visual embedded response using t-SNE. For example, Filter 38 in Layer 13 has a CKA specialization value of 0.6321 in sleep stage S2, the highest among all of the stages, and the corresponding t-SNE plot in Figure 8h similarly suggests a separation of stage S2 instances (left) from the other stages.

At the whole-layer level, we measured various layers’ ability to identify individual stages via our stage specialization measure, using max pooling over time to reduce dimensionality of the layer activation vectors before applying CKA. Figure 13a shows that the ability of the final model to identify individual stages improves with increasing depth; the “all stages” line shows that the ability to differentiate among stages,

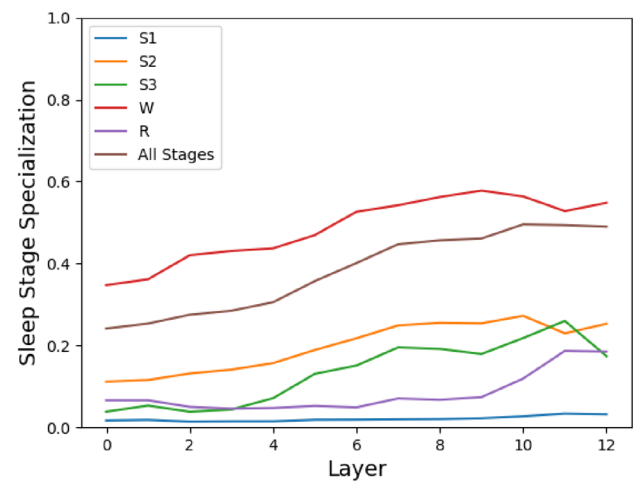
Table 6 CKA stage specialization values for filters shown in Figs. 8 and 9. Corresponding filter identity and t-SNE figure number indicated at left end of each row. Increased occurrence of larger CKA values in second and third three-row groups from top connotes an increase in stage specialization with layer depth. Largest value in each

row matches a stage that is visually well separated from others in the corresponding t-SNE plot. See text. Results show that CKA yields a quantitative measure of stage specialization that validates t-SNE visualization

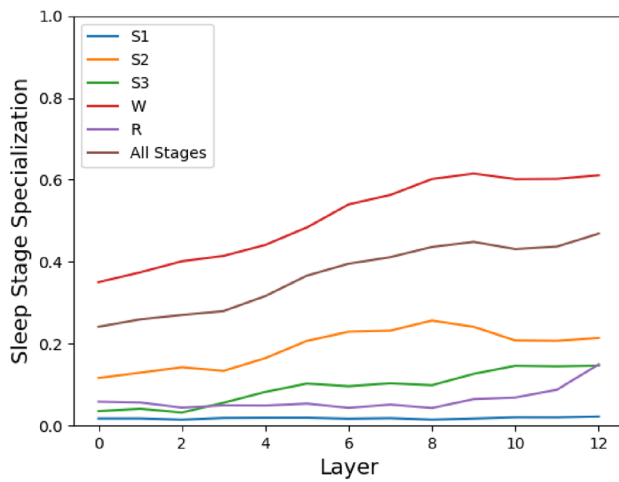
Layer	Filter	Corresponding t-SNE plot	Stage Specialization				
			S1	S2	S3	Wake	REM
1	15	8(a)	0.0405	0.0184	0.2726	0.0268	0.1077
1	4	8(b)	0.0025	0.1799	0.0015	0.3416	0.0056
1	13	8(c)	0.0064	0.1645	0.0158	0.3156	0.0156
8	48	8(d)	0.0047	0.1969	0.0039	0.5879	0.0169
9	32	8(e)	0.0290	0.0223	0.5673	0.1847	0.0901
9	63	8(f)	0.0198	0.2084	0.0514	0.2470	0.0016
13	92	8(g)	0.0053	0.1068	0.0510	0.7133	0.0499
13	38	8(h)	0.0070	0.6321	0.0939	0.1519	0.1037
13	61	8(i)	0.0018	0.1270	0.5529	0.0009	0.0296
12	26	9(a)	0.0013	0.0245	0.1152	0.1293	0.6588
13	52	9(b)	0.0140	0.0472	0.7479	0.0513	0.0695
13	17	9(c)	0.0024	0.0303	0.0989	0.1122	0.6786



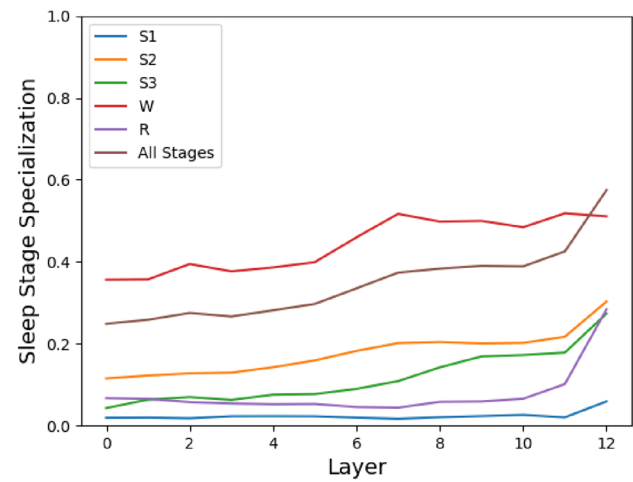
(a) Final model



(b) Outer2 - inner1 model



(c) Outer2 - inner2 model



(d) Outer5 - inner2 model

Fig. 13 CKA quantifies the improvement in stage specialization (individual stage labels in figures) and stage differentiation (“All Stages”) with layer depth. Models trained on three-channel data. The notation “Outer i - inner j ” refers to the model resulting from the j th inner fold of the i th outer fold of the nested cross-validation procedure.

Precise results differ slightly among networks with the same architecture (different parameter initializations and training data samples), but show a very similar overall growth trend for every one of the sleep stages

and hence make a correct class prediction, also increases with depth. We verified the results using other models with the same architecture as the final model. The results are shown in Fig. 13b–d. Despite slight variations in the detailed growth patterns among the individual figures, all of the models show the same trend in the increase in class specialization and class differentiation with layer depth.

Discussion

The convolutional neural network architecture developed in this paper attains an overall classification accuracy of $84.50\% \pm 0.13\%$, which is comparable to that of recent published results over the same data set, such as [29]

(84.26% accuracy on single-channel EEG data) and [36] (82.5% accuracy). Unsurprisingly, we see in Table 4(a) that predictive performance is weakest on stage S1, which is under-represented in the data (Table 1) and has low inter-scorer reliability among human experts [35]. We used oversampling of the S1 data instances in an attempt to improve performance on this stage. The results show only a minimal improvement in the F1 score of sleep stage S1; furthermore, a deterioration in predictive performance on other stages occurs, with a consequent decrease in the classification accuracy of the model. Similar results using oversampling were reported by [42], which used a different dataset.

As observed in Table 4, the one-channel model (c) slightly outperforms the other two models (a), (b) on Wake data, as measured by F1 score, for example. We hypothesize that this may be a matter of sample complexity: models that use a greater number of input channels have correspondingly larger numbers of parameters, and therefore (e.g., [13]) can be expected to require more data points to attain a comparable level of generalization performance relative to their asymptotic (large sample) performance limit, as compared with the single-channel model. If this hypothesis is correct, one expects that the three-channel model will perform best on a larger data set. We have not yet attempted experiments on a larger dataset to verify this prediction.

Taken collectively, our work and that of other groups (e.g., [29] and [36]) shows that the state of the art in automated sleep stage scoring now matches or exceeds human expert performance; for comparison, mean human expert inter-scorer agreement has been reported to be 82.6% [38]. We therefore focused our attention in this paper to studying the internal representations learned by high-performing deep learning models, which are known to be highly difficult to interpret.

Our approach combines visualization of activation vectors using t-SNE with quantification of stage differentiation using CKA after pooling. We found this combination to be effective. For example, it allows us to establish a near-monotonic improvement in sleep stage specialization with layer depth within our network, with partial stage differentiation even in shallow layers. We found, further, that CKA can be useful for validating the visual information provided by t-SNE. For example, at first, the embedded response of the filter in Fig. 8f gave us the impression that the filter specializes in sleep stage Wake, but the CKA value shows a low level of specialization. We revisited the t-SNE plot, and found that we had visually misjudged the filter's degree of specialization, due to the order in which instances of the various stages had been plotted: instances of sleep stage Wake had been plotted last, overlaying them on other instances in a way that suggested a higher degree

of separation of Wake from the other stages than was actually present in the two-dimensional projection.

Conclusions

We have developed a deep convolutional neural network for sleep stage classification that attains an overall classification accuracy of $84.50 \pm 0.13\%$ over three-channel (two EEG and one EOG) polysomnographic data, which is better than human expert inter-scorer agreement [38]. Single-channel performance is only slightly lower, and is competitive with contemporary results over the same dataset [29, 36].

Our prior work [33, 40, 41] used mid-sleep data only. In the present work, we used 30 additional minutes of pre-sleep and post-sleep Wake data to better align our evaluation approach with related work. We found that Wake epochs before and after sleep have only a minor impact on mid-sleep generalization, but that they are easier to predict than mid-sleep Wake epochs.

Our main focus was on the interpretation of the internal representations that develop within our deep network during training. We explored these representations through a combination of visual and quantitative means. Low-dimensional t-SNE plots of the activation vectors of different units and layers of the network provided insight into the degree of stage differentiation that arises across the network. The visual information was complemented by applying CKA to different measurements on the activation vectors, yielding quantitative measures of differentiation among stages and of specialization in particular stages.

We proposed the use of pooling over time to make the CKA computations feasible by reducing the dimensionality of the activation vector data. Our approach enabled us to establish a gradual increase in stage specialization and stage differentiation ability with increasing layer depth within the network, and to reveal that partial stage differentiation occurs even in shallow layers.

Many methods developed for interpreting and understanding deep learning focus on activation maximization [14, 26, 30] or on input credit assignment based on maximum impact on activation [3, 39, 44]. Our results in this paper suggest that the use of embedding techniques such as t-SNE could be helpful in providing further insight into the internal representations in deep networks. Our results further suggest that the visual inspection of such low-dimensional data embeddings can be validated and enhanced by using Centered Kernel Alignment (CKA) to provide objective quantitative descriptions of the development of internal feature representations during learning.

Future Work

Possible avenues for future work include investigating the use of CKA with nonlinear kernels, the application of our proposed techniques to alternative network types such as recurrent neural networks, and the simultaneous and complementary use of CKA-based quantification and activation maximization approaches. Our work in progress [32] involves the latter direction.

Acknowledgements The authors thank the anonymous reviewers for their helpful comments.

Availability of data and materials Publicly available.

Code availability Available upon email request.

Declarations

Conflict of interest Sarun Paisarnsrisomsuk declares that he has no conflict of interest. Carolina Ruiz declares that she has no conflict of interest. Sergio A. Alvarez declares that he has no conflict of interest.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Abadi M, Agarwal A, Barham P, Brevdo E, Chen Z, Citro C, Corrado GS, Davis A, Dean J, Devin M, et al. Tensorflow: large-scale machine learning on heterogeneous distributed systems. arXiv preprint [arXiv:1603.04467](https://arxiv.org/abs/1603.04467) (2016).
- Araujo A, Norris W, Sim J. Computing receptive fields of convolutional neural networks. *Distill*. <https://doi.org/10.23915/distill.00021>. <https://distill.pub/2019/computing-receptive-fields> (2019).
- Bach S, Binder A, Montavon G, Klauschen F, Müller KR, Samek W. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLoS One*. 2015;10(7):E0130140.
- Berry RB, Brooks R, Gamaldo CE, Harding SM, Marcus C, Vaughn BV, et al. The AASM manual for the scoring of sleep and associated events. Rules, Terminology and Technical Specifications, Darien, Illinois, American Academy of Sleep Medicine, vol. 176; 2012.
- Biswal S, Sun H, Goparaju B, Westover MB, Sun J, Bianchi MT. Expert-level sleep scoring with deep neural networks. *J Am Med Inform Assoc*. 2018;25(12):1643–50.
- Carskadon MA, Dement WC. Normal human sleep: an overview. In: Kryger MH, Roth T, Dement WC, editors. *Principles and practice of sleep medicine*. 6th ed. Amsterdam: Elsevier; 2016. p. 15–24.
- Chambon S, Galtier MN, Arnal PJ, Wainrib G, Gramfort A. A deep learning architecture for temporal sleep stage classification using multivariate and multimodal time series. *IEEE Trans Neural Syst Rehabil Eng*. 2018;26(4):758–69.
- Chen K, Zhang C, Ma J, Wang G, Zhang J. Sleep staging from single-channel EEG with multi-scale feature and contextual information. *Sleep and breathing*, p. 1–9 (2019).
- Chollet F, et al. *Keras* (2015)
- Colten HR, Altevogt BM, editors. *Sleep disorders and sleep deprivation: an unmet public health problem*, vol. 4. National Academies Press, Washington (2006). <https://www.ncbi.nlm.nih.gov/books/NBK19958/>.
- Cortes C, Mohri M, Rostamizadeh A. Algorithms for learning kernels based on centered alignment. *J Mach Learn Res*. 2012;13:795–828.
- Cristianini N, Kandola J, Elisseeff A, Shawe-Taylor J. On kernel target alignment. In: *Innovations in machine learning*. Springer, p. 205–56; 2006.
- Du SS, Wang Y, Zhai X, Balakrishnan S, Salakhutdinov R, Singh A. How many samples are needed to estimate a convolutional neural network? In: *Proceedings of the 32nd international conference on neural information processing systems, NIPS'18*, p. 371–81. Curran Associates Inc., Red Hook; 2018.
- Erhan D, Bengio Y, Courville A, Vincent P. Visualizing higher-layer features of a deep network. In: *ICML 2009 workshop on learning feature hierarchies*.
- Glorot X, Bengio Y. Understanding the difficulty of training deep feedforward neural networks. In: Teh YW, Titterton DM, editors. *Proceedings of the thirteenth international conference on artificial intelligence and statistics, AISTATS 2010, Chia Laguna Resort, Sardinia, Italy, May 13–15, 2010, JMLR Proceedings*, vol. 9, p. 249–256. *JMLR.org*; 2010. <http://proceedings.mlr.press/v9/glorot10a.html>.
- Goldberger A, Amaral L, Glass L, Hausdorff J, Ivanov P, Mark R, Mietus J, Moody G, Peng C, Stanley H. *Physiobank, physiotookit, and physionet*. *Circulation*. 1997;101(23).
- Grandner MA. Sleep, health, and society. *Sleep Med Clin*. 2017;12(1):1–22. <https://doi.org/10.1016/j.jsmc.2016.10.012>.
- He K, Zhang X, Ren S, Sun J. Delving deep into rectifiers: surpassing human-level performance on imagenet classification. In: *Proc. IEEE Intl. Conf. on Computer Vision*, p. 1026–1034; 2015.
- Humayun AI, Sushmit AS, Hasan T, Bhuiyan MIH. End-to-end sleep staging with raw single channel EEG using deep residual convnets. In: *2019 IEEE EMBS Intl. Conf. Biomedical and Health Informatics (BHI)*, p. 1–5. IEEE; 2019.
- Jiang D, Lu Y, Yu M, Yuanyuan W. Robust sleep stage classification with single-channel EEG signals using multimodal decomposition and hmm-based refinement. *Expert Syst Appl*. 2019;121:188–203.
- Kemp B, Zwinderman AH, Tuk B, Kamphuisen HAC, Obery J. 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>.
- Korkalainen H, Aakko J, Nikkonen S, Kainulainen S, Leino A, Duce B, Afara IO, Myllymaa S, Toyra J, Leppänen T. Accurate deep learning-based sleep staging in a clinical population with suspected obstructive sleep apnea. *IEEE J Biomed Health Inform*; 2019.

23. Kornblith S, Norouzi M, Lee H, Hinton G. Similarity of neural network representations revisited. arXiv preprint [arXiv:1905.00414](https://arxiv.org/abs/1905.00414) (2019).
24. Le H, Borji A. What are the receptive, effective receptive, and projective fields of neurons in convolutional neural networks? CoRR. [arXiv:1705.07049](https://arxiv.org/abs/1705.07049) (2017).
25. Liao Y, Zhang M, Wang Z, Xie X. Design and FPGA implementation of an high efficient XGBoost based sleep staging algorithm using single channel EEG. In: Intl. Conf. on Cognitive Systems and Signal Processing, p. 294–303. Springer; 2018.
26. Mahendran A, Vedaldi A. Understanding deep image representations by inverting them. In: Proceedings of IEEE conference computer vision and pattern recognition, p. 5188–5196; 2015.
27. McInnes L, Healy J, Saul N, Großberger L. UMAP: uniform manifold approximation and projection. *J Open Source Soft.* 2018;3(29):861. <https://doi.org/10.21105/joss.00861>.
28. Mourtazaev M, Kemp B, Zwinderman A, Kamphuisen H. Age and gender affect different characteristics of slow waves in the sleep eeg. *Sleep.* 1995;18(7):557–64.
29. Mousavi S, Afghah F, Acharya UR. SleepEEGNet: automated sleep stage scoring with sequence to sequence deep learning approach. *PLoS One.* 2019;14(5):e0216456.
30. Nguyen A, Clune J, Bengio Y, Dosovitskiy A, Yosinski J. Plug and play generative networks: conditional iterative generation of images in latent space. In: Proceedings of IEEE conference on computer vision and pattern recognition, p. 4467–4477; 2017.
31. Paisarnsrisomsuk S, Ruiz C, Alvarez S. Improved deep learning classification of human sleep stages. In: 2020 IEEE 33rd international symposium on computer-based medical systems (CBMS), p. 338–343. <https://doi.org/10.1109/CBMS49503.2020.00070> (2020).
32. Paisarnsrisomsuk S, Ruiz C, Alvarez S. Interpretable deep learning for predictive modeling of human sleep (2021) (in progress).
33. Paisarnsrisomsuk S, Sokolovsky M, Guerrero F, Ruiz C, Alvarez SA. Deep sleep: convolutional neural networks for predictive modeling of human sleep time-signals. *ACM KDD2018 Deep Learning Day*; 2018.
34. Patanaik A, Ong JL, Gooley JJ, Ancoli-Israel S, Chee MW. An end-to-end framework for real-time automatic sleep stage classification. *Sleep.* 2018;41(5):zsy041.
35. Penzel T, Zhang X, Fietze I. Inter-scorer reliability between sleep centers can teach us what to improve in the scoring rules. *J Clin Sleep Med.* 2013;9(1):89–91.
36. Phan H, Andreotti F, Cooray N, Chén OY, De Vos M. Automatic sleep stage classification using single-channel EEG: learning sequential features with attention-based recurrent neural networks. In: 2018 40th international conference of IEEE the Engineering in Medicine and Biology Society (EMBC), p. 1452–1455. IEEE; 2018.
37. Raghu M, Gilmer J, Yosinski J, Sohl-Dickstein J. Svcca: Singular vector canonical correlation analysis for deep learning dynamics and interpretability. In: Advances in neural information processing systems, p. 6076–6085; 2017.
38. Rosenberg RS, Van Hout S. The American Academy of Sleep Medicine inter-scorer reliability program: sleep stage scoring. *J Clin Sleep Med.* 2013;9(01):81–7.
39. Simonyan K, Vedaldi A, Zisserman A. Deep inside convolutional networks: visualising image classification models and saliency maps. arXiv preprint [arXiv:1312.6034](https://arxiv.org/abs/1312.6034) (2013).
40. Sokolovsky M, Guerrero F, Paisarnsrisomsuk S, Ruiz C, Alvarez S. Deep learning for automated feature discovery and classification of sleep stages. *IEEE/ACM Trans Comput Biol Bioinform.* 2019. <https://doi.org/10.1109/TCBB.2019.2912955>.
41. Sokolovsky M, Guerrero F, Paisarnsrisomsuk S, Ruiz C, Alvarez SA. Human expert-level automated sleep stage prediction and feature discovery by deep convolutional neural networks. In: Proceedings of 17th international workshop on data mining in bioinformatics (BIOKDD), in conjunction with KDD2018; 2018.
42. Sors A, Bonnet S, Mirek S, Vercueil L, Payen JF. A convolutional neural network for sleep stage scoring from raw single-channel EEG. *Biomed Signal Process Control.* 2018;42:107–14.
43. Stephansen JB, Olesen AN, Olsen M, Ambati A, Leary EB, Moore HE, Carrillo O, Lin L, Han F, Yan H, et al. Neural network analysis of sleep stages enables efficient diagnosis of narcolepsy. *Nat Commun.* 2018;9(1):1–15.
44. Sung A. Ranking importance of input parameters of neural networks. *Expert Syst Appl.* 1998;15(3–4):405–11.
45. 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.
46. Tsinalis O, Matthews PM, Guo Y. Automatic sleep stage scoring using time-frequency analysis and stacked sparse autoencoders. *Ann Biomed Eng.* 2016;44(5):1587–97.
47. Tsinalis O, Matthews PM, Guo Y, Zafeiriou S. Automatic sleep stage scoring with single-channel EEG using convolutional neural networks. arXiv preprint [arXiv:1610.01683](https://arxiv.org/abs/1610.01683) (2016).
48. Van der Maaten L, Hinton G. Visualizing data using t-SNE. *J Mach Learn Res.* 2008;9.
49. Wang L, Hu L, Gu J, Hu Z, Wu Y, He K, Hopcroft J. Towards understanding learning representations: to what extent do different neural networks learn the same representation. In: Advances in neural information processing systems, p. 9584–9593; 2018.
50. Younes M, Raneri J, Hanly P. Staging sleep in polysomnograms: analysis of inter-scorer variability, p. 885–894; 2016. <https://doi.org/10.5664/jcsm.5894>.
51. Zeiler MD, Fergus R. Visualizing and understanding convolutional networks. In: Fleet D, Pajdla T, Schiele B, Tuytelaars T, editors. *Computer vision—ECCV 2014*. Cham: Springer International Publishing; 2014. p. 818–33.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.