Using PCA on EEG Data to Distinguish Sleep Stages

Ida Hönigmann
Technische Universität Wien, Austria
Email: e12002348@student.tuwien.ac.at

***Abstract*—[TODO]**

I. Introduction

[TODO general introduction]

A. EEG Data and Sleep Stages

Ganong [4] describes typical patterns observed in electroencephalogram (EEG) data of a sleeping person. He describes the EEG patterns associated with rapid eye movement (REM) sleep and non-REM (NREM) sleep.

NREM sleep is further partitioned into four (although some only use three) stages, termed Stage 1 (S1) to Stage 4 (S4). Example EEG data of these different sleep stages can be seen in Figure **??** [TODO image]. The EEG data of these stages is characterized as follows:

S1: low-amplitude, high-frequency
S2: appearance of sleep spindles (bursts of higher amplitude, lower frequency waves)
S3: increased amplitude, lower frequency
S4: maximal amplitude, minimal frequency

In REM sleep the EEG data is that of high frequency and low amplitude patterns, resembling the data observed in alert humans.

II. Study of Literature

A substantial body of scientific research has been devoted to exploring Principal Component Analysis (PCA). The foundation of this method was laid by Pearson [11] and Hotelling [6].

An introduction to PCA, as well as a good overview on how to derive the formula used to compute the Principal Components (PC) is given by Shlens [14]. Recent applications and variants of PCA are explored by Jolliffe et. al. [8].

Shlens discusses the limitations of PCA, as well as examples in which PCA fails, such as the requirement of linearly dependent data. Tenenbaum proposes a non-linear method to combat this problem[15].

Generally speaking the variables must not have third or higher order dependencies[1] between them. In some cases it is possible to reduce a problem with higher order dependencies to a second order one by applying a non-linear transformation beforehand. This method is called kernel PCA[13].

[1]e.g. $\mathbb{E}[x_i x_j x_k] \neq 0$ for some i, j, k assuming mean-free variables

Another method for dealing with this problem is Independent Component Analysis (ICA) which is discussed by Naik et. al.[10].

The given problem of distinguishing sleep stages given some EEG data has been investigated by use of PCA, as well as neural networks. Some of these works are summarized below.

A review of different methods in the preprocessing, feature extraction and classification is given by Boostani et. al.[1]. They find that using a random forest classifier[2] and entropy of wavelet coefficients[3] as feature gives the best results.

Tăuţan et. al.[16] compare different methods of dimensionality reduction on EEG data, such as PCA, factor analysis and autoencoders. They conclude that the use of PCA and factor analysis improves the accuracy of the model.

Putilov[12] used PCA to find boundaries between Stage 1, Stage 2 and Stage 3. Changes in the first two PC were related to changes between the Stage 1 and Stage 2, while changes in the fourth PC exhibited a change in sign at the boundary of Stage 2 and Stage 3. This suggests that changes between Stage 1 and Stage 2 are easier to detect that ones between Stage 2 and Stage 3.

Metzner et. al.[9] try to rediscover the different human-defined sleep stages. They find that using PCA on the results makes clusters apparent. These clusters could then be used as a basis for a redefinition of sleep stages.

The PhysioNet/Computing in Cardiology Challange 2018[5] was a competition using a similar dataset. The goal was to identify arousal during sleep from EEG, EOG, EMG, ECG and SaO2 data given. The winning paper[7] of this competition describes the use of a dense recurrent convolutional neural network (DRCNN) comprised of multiple dense convolutional layers, a bidirectional long-short term memory layer and a softmax output layer.

As shown in this section, the utilization of PCA to analyze EEG data has been used with success.

III. Mathematical Basics

We define mathematical notation, which will be used in Section IV to define PCA.

A. Covariance

Assume we have two sets of n observations of variables with mean 0. Let us call the first list of observations $\mathbf{a} = (a_1, ..., a_n)$

and the second $\mathbf{b} = (b_1, ..., b_n)$.

Definition 1 (Covariance)**.** *Let us define the covariance of* $\mathbf{a} \in \mathbb{R}^n$ *and* $\mathbf{b} \in \mathbb{R}^n$ *as*

$$\sigma_{\mathbf{ab}} := \frac{1}{n}\sum_{i=1}^{n} a_i b_i = \frac{1}{n}\mathbf{a} \cdot \mathbf{b}^T.$$

From the definition it is obvious that the covariance is symmetric, $\sigma_{\mathbf{ab}} = \sigma_{\mathbf{ba}}$. In the special case $\mathbf{a} = \mathbf{b}$ the covariance $\sigma_{\mathbf{aa}}$ is called *variance* $\sigma^2_{\mathbf{a}}$.

Definition 2 (Covariance Matrix)**.** *Generalizing to* m *variables* $\mathbf{X} = (\mathbf{x_1}, ..., \mathbf{x_m})$*, each having been observed* n *times, gives us the covariance matrix.*

$$\mathbf{C_X} := \begin{pmatrix} \sigma_{\mathbf{x_1x_1}} & \cdots & \sigma_{\mathbf{x_1x_m}} \\ \vdots & \ddots & \vdots \\ \sigma_{\mathbf{x_mx_1}} & \cdots & \sigma_{\mathbf{x_mx_m}} \end{pmatrix} = \frac{1}{n}\mathbf{X}\mathbf{X}^T$$

The covariance matrix is a symmetric $m \times m$ matrix.

B. Diagonalizable Matrix

Definition 3 (Diagonalizable Matrix)**.** *A square matrix* $\mathbf{A}$ *is called diagonalizable, if there exists an invertable matrix* $\mathbf{P}$ *and a diagonal matrix* $\mathbf{D}$ *such that* $\mathbf{A} = \mathbf{PDP}^{-1}$.

Definition 4 (Symmetric matrix)**.** *A square matrix* $\mathbf{A}$ *is called symmetric, if* $\mathbf{A}^T = \mathbf{A}$.

Theorem 1. *Every symmetric matrix is diagonalizable.*

This is the main theorem we need in order to derive PCA. The proof of this theorem requires some preparation, which we will do now.

Definition 5 (Eigenvalues and Eigenvectors)**.** *Let* $\mathbf{A}$ *be a real* $m \times m$ *matrix.* $\lambda \in \mathbb{C}$ *is called a eigenvalue with eigenvector* $\mathbf{v} \in \mathbb{C}^m \setminus \{\mathbf{0}\}$ *if*

$$\mathbf{Av} = \lambda\mathbf{v}. \tag{1}$$

Lemma 1. *Every square* $m \times m$ *matrix has* m *(not necessarily unique) eigenvalues.*

Proof. We can rewrite equation 1 as

$$(\mathbf{A} - \lambda\mathbf{I})\mathbf{v} = \mathbf{0}$$

This allows us to interpret $(\mathbf{A} - \lambda\mathbf{I})$ as a function, which takes vectors $\mathbf{v} \in \mathbb{C}^m$. For λ to be a eigenvalue of $\mathbf{A}$ with eigenvector $\mathbf{v}$ it has to satisfy $\mathbf{v} \in \ker(\mathbf{A} - \lambda\mathbf{I})$ and $\mathbf{v} \neq 0$. From this we gather that all λ with $\ker(\mathbf{A} - \lambda\mathbf{I}) \neq \{\mathbf{0}\}$ are eigenvalues. We know this holds if and only if $\det(\mathbf{A} - \lambda\mathbf{I}) = 0$. The determinant is a polynomial of degree m which can be expressed in the form $(\lambda - \lambda_1)...(\lambda - \lambda_m)$ with $\lambda_1, ..., \lambda_m \in \mathbb{C}$. These $\lambda_1, ..., \lambda_m$ are the m eigenvalues we wanted to find. □

Lemma 2. *A symmetric matrix has real eigenvalues.*

Proof. Let $\bar{\cdot}$ denote the complex conjugate. Define a complex dot product

$$(\mathbf{u}, \mathbf{v}) := \sum_{i=1}^{m} u_i \bar{v_i}$$

This dot product has the following properties for all $\mathbf{A} \in \mathbb{C}^{m\times m}, \mathbf{u}, \mathbf{v} \in \mathbb{C}^m, \lambda \in \mathbb{C}$

- $(\mathbf{Au}, \mathbf{v}) = (\mathbf{u}, \mathbf{A}^T\mathbf{v})$,
- $(\lambda\mathbf{u}, \mathbf{v}) = \lambda(\mathbf{u}, \mathbf{v})$,
- $(\mathbf{u}, \lambda\mathbf{v}) = \bar{\lambda}(\mathbf{u}, \mathbf{v})$
- $(\mathbf{u}, \mathbf{u}) = 0 \iff \mathbf{u} = 0$

Let $\mathbf{A}$ be a symmetric matrix with eigenvalue $\lambda \in \mathbb{C}$.
For all $\mathbf{u} \in \mathbb{C}^m$ we have

$$\lambda(\mathbf{u}, \mathbf{u}) = (\lambda\mathbf{u}, \mathbf{u}) = (\mathbf{Au}, \mathbf{u}) = (\mathbf{u}, \mathbf{A}^T\mathbf{u}) = (\mathbf{u}, \mathbf{Au}) = (\mathbf{u}, \lambda\mathbf{u}) = \bar{\lambda}(\mathbf{u}, \mathbf{u}).$$

For $\mathbf{u} \neq \mathbf{0}$ we get $\lambda = \bar{\lambda}$ and thus $\lambda \in \mathbb{R}$. □

Are the corresponding eigenvectors real? From the proof of lemma 1 we know that the eigenvector $\mathbf{v}$ of eigenvalue λ is in $\ker(\mathbf{A} - \lambda\mathbf{I})$. Both the matrix $\mathbf{A}$ and λ are real, so $\mathbf{v}$ must be in $\mathbb{R}^m$ as well.

Lemma 3. *The eigenvectors of a symmetric matrix with distinct eigenvalues are orthogonal.*

Proof. Let λ_1, λ_2 be two distinct eigenvalues with eigenvectors $\mathbf{v}_1, \mathbf{v}_2$ of the matrix $\mathbf{A}$.

$$\lambda_1\mathbf{v}_1 \cdot \mathbf{v}_2 = (\lambda_1\mathbf{v}_1)^T\mathbf{v}_2 = (\mathbf{Av}_1)^T\mathbf{v}_2 = \mathbf{v}_1^T\mathbf{A}^T\mathbf{v}_2 = \mathbf{v}_1^T\mathbf{Av}_2 = \mathbf{v}_1^T(\lambda_2\mathbf{v}_2) = \lambda_2\mathbf{v}_1 \cdot \mathbf{v}_2$$

This shows $(\lambda_1 - \lambda_2)\mathbf{v}_1 \cdot \mathbf{v}_2 = 0$ and as λ_1 and λ_2 are distinct, $\mathbf{v}_1$ and $\mathbf{v}_2$ must be orthogonal. □

What if the eigenvalues of the matrix are not distinct? In the proof of lemma 1 we showed that every $\mathbf{v} \in \ker(\mathbf{A} - \lambda\mathbf{I}) \setminus \{\mathbf{0}\}$ is a eigenvector. If and only if $(\lambda - \lambda_i)$ appears $k \geq 2$ times in the determinant of $(\mathbf{A} - \lambda\mathbf{I})$ then $\mathbf{A}$ has a non unique eigenvalue λ_i. As $\dim(\ker(\mathbf{A} - \lambda_i\mathbf{I})) = k$ we can choose orthogonal eigenvectors.

Now we have everything we need to prove theorem 1.

Proof of Theorem 1. Let $\mathbf{A} \in \mathbb{R}^{m\times m}$ be a symmetric matrix. From lemma 1 we know that eigenvalues $\lambda_1, ..., \lambda_m$ with corresponding eigenvectors $\mathbf{v}_1, ..., \mathbf{v}_m$ exist.

Define the following matrices

$$\mathbf{D} := \begin{pmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_m \end{pmatrix} \quad \mathbf{V} := \begin{pmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \cdots & \mathbf{v}_m \end{pmatrix}$$

The definition of eigenvalues and eigenvectors gives us

$$\mathbf{AV} = \begin{pmatrix} \mathbf{Av}_1 & \cdots & \mathbf{Av}_m \end{pmatrix} = \begin{pmatrix} \lambda_1\mathbf{v}_1 & \cdots & \lambda_m\mathbf{v}_m \end{pmatrix} = \mathbf{VD}. \tag{2}$$

From lemma 3 we know that the eigenvectors, and therefore the columns of $\mathbf{V}$, are orthogonal. It follows that $rank(\mathbf{V}) = m$ which gives us the existence of $\mathbf{V}^{-1}$.

Rearranging equation 2 now gives us $\mathbf{A} = \mathbf{VDV}^{-1}$ which is what we wanted to show.

This shows that $\mathbf{A}$ is diagonalizable. □

Lemma 4. *If the columns of matrix* $\mathbf{A}$ *are orthonormal, then* $\mathbf{A}^{-1} = \mathbf{A}^T$.

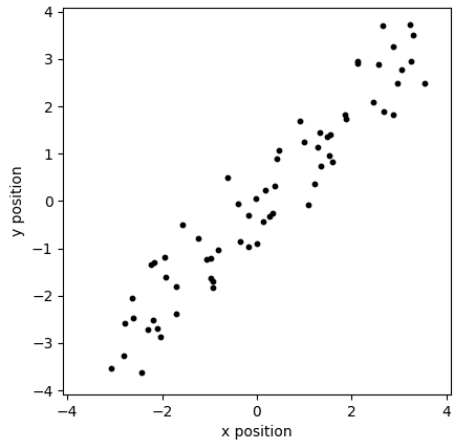


Fig. 1: Randomly generated sample data. The data lies along a line with slope 1 and has mean $\mathbf{0}$.

Proof. Let $(\mathbf{a}_i)_{i=1,\dots,m}$ be the columns of the matrix. The columns are orthogonal and normed, therefore

$$\forall i,j : \mathbf{a}_i^T \mathbf{a}_j = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases} \implies \mathbf{A}^T\mathbf{A} = \mathbf{I}$$

This shows $\mathbf{A}^{-1} = \mathbf{A}^T$. □

IV. Principal Component Analysis

Combining the concepts in section III we derive the ideas and implementation of PCA.

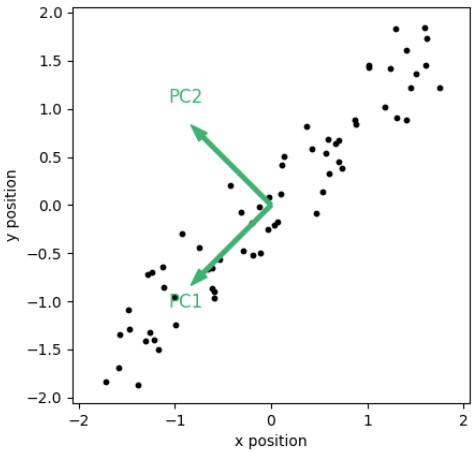
Assume we have gathered observations of different variables as part of an experiment. If we have n variables, each having been observed m times, we can create a $m \times n$ matrix of this data. The goal is to get more insight and find underlying patterns in the collected data. For $n = 2$ we could try to plot the data, with the first variable as the x-axis and the second as the y axis. An exemplary plot of some data can be seen in figure 1.

For larger values of n this gets increasingly difficult[2]. PCA tries to solve this problem by transforming the data in such a way that the most interesting features are in the first few axis of the transformed m dimensional space. This makes it easy to look at a low dimension representation of the data, without loosing much information.

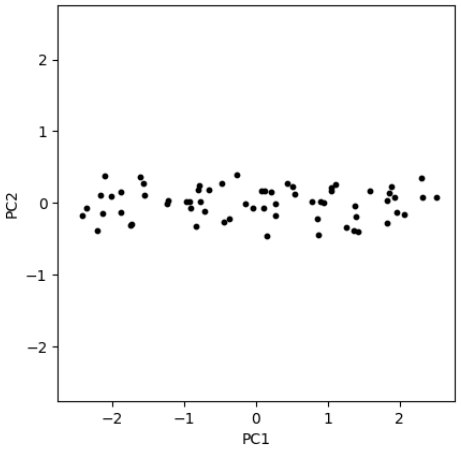
An example of PCA being applied to the data from figure 1 can be seen in figure 2. In the top figure the normed data and the direction of the new axis (called Principal Components (PC)) in relation to the two original axis are shown. The bottom figure depicts the transformed data. One can see that the variance is maximal in the PC 1 axis.

Now we derive how to compute PCA. First we formulate a goal and define some assumptions.

[2]For higher dimensionality we have to use some projection. Depending on the chosen projection the interpretation changes making it difficult to interpret the resulting image.



(a) Normed data (mean is zero and variance is one) and direction of the two PCs in relation to the x and y position.



(b) The data after being transformed by PCA. The variance along the PC1 axis is maximal, therefore the data is spread out most along this axis.

Fig. 2: Example application of PCA.

We assume that the most interesting features are those that have a large variance[3]. Our goal is to find a transformation into new coordinates such that:

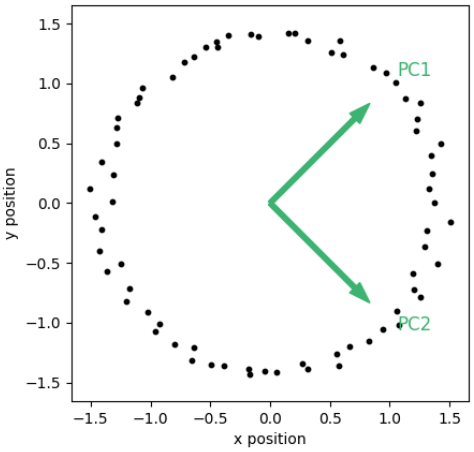
- the variance in the each axis is as large as possible.
- the axis are all orthogonal to each other.
- the axis are sorted (descending) by the variance in the axis.

From this we gather that another assumption is, that the axis are orthogonal. Lastly we are only concerned with linear dependent features in the data. Some example cases in which PCA fails are shown in figure 3.

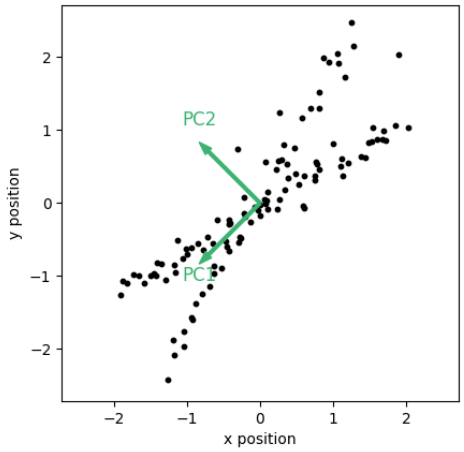
One way to achieve the goal is as follows:

1) Find the direction which maximizes the variance.

[3]This assumption can be false. For data where the noise has a larger variance than the feature we are trying to observe, PCA fails because this assumption is not met.



(a) Clearly the relationship in this figure is non-linear. PCA can not describe circular dependencies, as shown in this data.



(b) The two main axis along which the data is aligned are not orthogonal to each other. PCA always outputs orthogonal principal components, therefore it fails in this example.

Fig. 3: Examples in which some of the assumptions of PCA are not valid. The results are sub-optimal.

2) Save this direction as the next axis.
3) Determine the subspace that is orthogonal to all axis we found so far.
4) If the subspace is non-trivial start at the first step again.
5) If the subspace is trivial we have found all axis.

While this algorithm shows us what conceptually has to be done, we do not know how to compute the axis yet. We will now investigate this problem using the mathematical concepts from section III. This will lead us to an algorithm in which all axis can be computed simultaneously.

Let $\mathbf{X} \in \mathbb{R}^{m \times n}$ be the data matrix. We want to find some orthonormal matrix $\mathbf{P}$ such that $\mathbf{Y} := \mathbf{PX}$ has a diagonal covariance matrix $\mathbf{C_Y}$.

$$\mathbf{C_Y} = \frac{1}{n}\mathbf{YY}^T = \frac{1}{n}(\mathbf{PX})(\mathbf{PX})^T = \frac{1}{n}\mathbf{PXX}^T\mathbf{P}^T = \\ = \mathbf{P}(\frac{1}{n}\mathbf{XX}^T)\mathbf{P}^T = \mathbf{PC_X P}^T$$

The covariance matrix $\mathbf{C_X}$ is symmetric and therefore has a decomposition into an orthogonal matrix of eigenvectors $\mathbf{V}$ and a diagonal matrix of eigenvalues $\mathbf{D}$. We choose $\mathbf{P} = \mathbf{V}^T$. From lemma 4 it follows that $\mathbf{V}^{-1} = \mathbf{V}^T$.

$$\mathbf{PC_X P}^T = \mathbf{P}(\mathbf{VDV}^{-1})\mathbf{P}^T = \mathbf{P}(\mathbf{VDV}^T)\mathbf{P}^T = \\ \mathbf{P}(\mathbf{P}^T\mathbf{DP})\mathbf{P}^T = (\mathbf{PP}^T)\mathbf{D}(\mathbf{PP}^T) = \\ (\mathbf{PP}^{-1})\mathbf{D}(\mathbf{PP}^{-1}) = \mathbf{D}$$

In summary $\mathbf{Y}$ has a diagonal covariance matrix if we choose $\mathbf{Y} = \mathbf{V}^T\mathbf{X}$, where $\mathbf{V}$ is the matrix of eigenvectors of $\mathbf{C_X}$. The eigenvectors are the PCs and the eigenvalues are the variance in each new axis.

As pseudo code we get the program from algorithm 1 for calculating the PCA.

Algorithm 1 Principal Component Analysis

Require: matrix $X \in \mathbb{R}^{m \times n}$
Normalize each row in the matrix X
Calculate the covariance matrix C_X
Calculate the eigenvalues and eigenvectors of C_X
Sort the eigenvalues
Return sorted eigenvalues and corresponding eigenvectors

What happens if we skip the step in which we normalize each row in the matrix? A big variance is interpreted by the PCA algorithm as much information, thus the variance of the variables have an impact on how ”important” the variable is deemed. As we do not want to prioritize certain variables we avoid this behavior by normalizing the data beforehand.

V. Methodology

For the algorithm in section VII a few other methods will be used. As they are not the focus of this work we only give a short overview.

A. Classification

In classification the objective is to find assignments between data points and categories. In our context we are interested in finding an assignment which closely matches some already categorized data. One simple approach to this problem is the k-nearest-neighbors algorithm.

For data that can be represented in $\mathbb{R}^m$ and l categories the pseudo code is shown in algorithm 2.

Figure 4 shows a graphical representation of the algorithm for points in $\mathbb{R}^2$ and $k = 10$.

This algorithm is slow for big datasets as for each point the distance to x has to be calculated. One possibility to reduce calculation time is to partition space into smaller chunks. Then the loop only has to be over data points lying in the same or close chunks as the point we are interested in.

Algorithm 2 k Nearest Neighbors

Require: data points $(p_i)_{i\in\mathbb{N}} \in \mathbb{R}^m$, categories $(c_i)_{i\in\mathbb{N}} \in \{0, 1, ..., l\}$, point $x \in \mathbb{R}^m$, $k \in \mathbb{N}$
Calculate the distance between each point p_i and x
Take the k data points with the smallest distance to x
Return the category that most of the k points are assigned

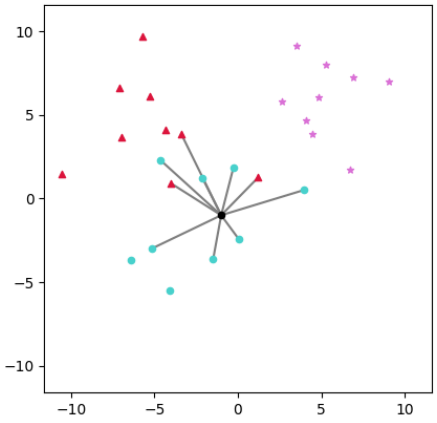


Fig. 4: Data points in three categories are given. The black point is classified as a blue, circular point by the k-nearest-neighbors algorithm for $k = 10$.

B. Fourier Transformation

[TODO]

VI. Sleep Stages and EEG Data

VII. Data and Algorithm

1) subdivide eeg signals in the temporal domain
2) apply fft transforming into frequency domain
3) pca
4) achive dimensinality reduction
5) classification of sleep stages
6) visulisation

VIII. Results

IX. Conclusion

References

[1] Reza Boostani, Foroozan Karimzadeh, and Mohammad Nami. A comparative review on sleep stage classification methods in patients and healthy individuals. *Computer Methods and Programs in Biomedicine*, 140:77 – 91, 2017. Cited by: 212.

[2] Leo Breiman. Random forests. *Machine learning*, 45:5–32, 2001.

[3] Charles K Chui, Laura Montefusco, and Luigia Puccio. *Wavelets: theory, algorithms, and applications*, volume 5. Academic press, 1994.

[4] William F. Ganong. *Review of medical physiology*. Appleton & Lange, Stamford, Conn, 18. ed edition, 1997.

[5] Mohammad M Ghassemi, Benjamin E Moody, Li wei H Lehman, Christopher Song, Qiao Li, Haoqi Sun, Roger G Mark, M Brandon Westover, and Gari D Clifford. You snooze, you win: the physionet/computing in cardiology challenge 2018. *2018 Computing in Cardiology Conference (CinC)*, pages 1–4, 2018.

[6] Harold Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of educational psychology*, 24(6):417, 1933.

[7] Matthew Howe-Patterson, Bahareh Pourbabaee, and Frederic Benard. Automated detection of sleep arousals from polysomnography data using a dense convolutional neural network. In *2018 Computing in Cardiology Conference (CinC)*, volume 45, pages 1–4. IEEE, 2018.

[8] I. T. Jolliffe and J. Cadima. Principal component analysis: a review and recent developments. *Royal Society*, 374(2065), 2016.

[9] Claus Metzner, Achim Schilling, Maximilian Traxdorf, Holger Schulze, Konstantin Tziridis, and Patrick Krauss. Extracting continuous sleep depth from eeg data without machine learning. *Neurobiology of Sleep and Circadian Rhythms*, 14, 2023. All Open Access, Gold Open Access, Green Open Access.

[10] Ganesh R Naik and Dinesh K Kumar. An overview of independent component analysis and its applications. *Informatica*, 35(1), 2011.

[11] Karl Pearson. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin philosophical magazine and journal of science*, 2(11):559–572, 1901.

[12] Arcady A. Putilov. Principal component analysis of the eeg spectrum can provide yes-or-no criteria for demarcation of boundaries between nrem sleep stages. *Sleep Science*, 8(1):16–23, 2015.

[13] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Kernel principal component analysis. In *International conference on artificial neural networks*, pages 583–588. Springer, 1997.

[14] Jonathon Shlens. A tutorial on principal component analysis. 2014.

[15] J.B. Tenenbaum, V. De Silva, and J.C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319 – 2323, 2000. Cited by: 10812.

[16] Alexandra-Maria Tăuţan, Alessandro C. Rossi, Ruben de Francisco, and Bogdan Ionescu. Dimensionality reduction for eeg-based sleep stage detection: comparison of autoencoders, principal component analysis and factor analysis. *Biomedical Engineering / Biomedizinische Technik*, 66(2):125–136, 2021.