ICA Source Separation for EEG Data

Motivation: ICA and The Cocktail Party Problem

Our motivation in this project derives from an interest to further explore unsupervised learning methods, extending beyond what we covered in class. Our team found an interest in Independent Component Analysis (ICA) as it functions to separate the individual signals that are mixed into larger combined input signals. In other words, taking a set of input signals, ICA attempts to deconstruct them and uncover the individual source signals that combine to make up each input signal.

In cognitive research, scientists commonly apply ICA to problems related to the "cocktail party problem". In short, the cocktail party problem models the phenomenon of the brain's ability to select audible signals to attend to from a mixed signal with noise. At a high level, this problem captures the phenomenon of how we're able to focus on an individual conversation in an audibly busy environment. ICA deconstructs this phenomenon by mathematically replicating how the brain separates the different sounds that are coming into our ears. Thus the cocktail party effect generalizes to other types of inputs. And so if you're able to mathematically model some input as a set of individual signals, ICA can uncover how that input is a combination of said independent signals/components.

ICA's use for EEG data

Electroencephalogram (EEG) testing is a testing method that records the neural activity of the brain through using a cap of electrodes placed on the scalp. Due to the size of the electrical impulses of brain activity being very small, these electrodes have to be very sensitive. In return, the electrodes are sensitive to the point that they often record signals from the slightest head movements and eye blinks. Because these outside noises are often reflected in the data, EEG testing can be likened to the cocktail party problem. Accordingly, ICA can be used as a technique to deconstruct the muddled EEG data and separate the brain's signals from the noisy artifacts from outside events. Our project attempts exactly this, specifically to separate blink recordings/artifacts from the brain waves that are captured in a sampled EEG test dataset.

From a mathematical lens, (ICA) is an algorithm that attempts to maximize the independence of a set of vectors. Applying this lens to EEG data, signals from each node can be represented as **n** vectors of time with **m** many components. Practically, the number of vectors **n** would be the number of samples that the EEG recording took (such as 14000 over 110 seconds), using **m** many electrodes. Thus an EEG data set could be represented as an **n** by **m** matrix which can then be used to perform ICA. The ICA algorithm would then attempt to maximize the independence of the brain function signals from the blinks (and presumably other artifacts) which are recorded by the EEG's electrodes, essentially categorizing the individuality of each signal.

ICA in relation to COGS 118B

A common preprocessing method used prior to running ICA on a dataset is the use of Principal Component Analysis (PCA), which we've learned in class. PCA is a method that summarizes a set of vectors/data by transforming it to a lower subspace, or reducing the dimensionality of the data, while maintaining the variant/discriminating information in the data. In fact, it can be used to organize and reshape a dataset according to its axes of maximum (or

principle if you will) variance, thus allowing one to pick and choose just how many varying components/dimensions to keep. In practice, it allows one to maintain only the most important, or “principal”, dimensions of the data. By preprocessing a dataset using PCA, we can remove redundant and/or noisy data points, keeping only the most representative parts of the data and inherently reducing computation time for future transforms of the now-smaller data set. Do note that PCA drastically differs from ICA however. At a high level, PCA aims to summarize a dataset while ICA attempts to separate the information of a dataset. While both are methods of blind source separation, it is important to remember that PCA is simply used as a preprocessing step before computing ICA.

Process of Implementation

Considering ICA is a fairly common model in data science, we were able to find and tweak pre-written software to fit our project’s implementation (Maklin, 2019). In short, the standard ICA algorithm involves whitening the data, initializing a random value for a de-mixing matrix, then iteratively updating the de-mixing matrix with respect to the whitened data and checking for convergence of said matrix. The purpose of whitening the data is to decorrelate the EEG signals, or make each individual signal uncorrelated with one another. This step was accomplished through the PCA, as PCA diagonalizes the covariance matrix of the data, such that the variance of each individual signal is 1. At a high level, the input signals may have some correlation, and by maximizing the covariance of the signals we directly decorrelate their relationships.

With respect to the de-mixing matrix, keep in mind that the input signal to the electrodes is a combination of noise from the brain’s activity and the eye blink. Thus we can think about the mixed input signal, or mixed signal, as a linear transformation on the individual signals of the blink and brain activity at any given point in time (Tharwat, 2020). Suppose that we know these individual components that underlie the input signals, then any input signal can be described in the alternative basis of the individual components. Thus by updating the de-mixing matrix with respect to the dataset, we can eventually converge to a representative de-mixing matrix that projects onto the individual components to compute the mixed signal.

As a last note we also imported SKLearn’s built-in FastICA function to test against our attempted implementation of ICA as a check for proof of concept. This proved to be particularly useful because, as we were implementing the algorithm, we often ran into inconsistencies between the two, which assured us that something was off. As it turns out, there were three extremely noisy points in the dataset that we neglected and failed to remove. As a result, our data was skewed and our results quite varying until we removed the points. This is important to note because it highlights the importance of preprocessing one’s EEG data before analysis. As powerful of an algorithm as ICA is, it still isn’t very useful on bad data.

Related, the data we used incorporated a binary vector to represent the state of the subject’s eye at the corresponding time of an EGG recording. When we initially ran our software on this dataset, we accidentally included that binary vector of blinks in our ICA and PCA calculation and the SKLearn’s version of FastICA. This added a lot of inconsistency between the graphed results between SKLearn’s and our algorithm, likely due to that binary vector influencing/varying the calculation in both implementations of the algorithm.

Results

After a number of trials and many errors in our implementation, the resulting software worked as expected. At the end of running the ICA we found that there were individual component signals input into the EEG cap that corresponded to brain functions or blinks. But it was not immediately apparent which signals to pick out the individual component graphs to correspond to the blinks due to the similarity in shape across all of them. With the help of the binary vector denoting blink times, we were able to visually align the components with respect to a graph plotting the blink vector. Through comparison we were able to visually identify two corresponding component graphs to the blink signals in the input signal. And while in retrospect, the two removed components did look a bit choppier on their graphs it would have been difficult to visually distinguish the blink components from the brain recording components without added assistance. It would be interesting to analyze specific techniques/thresholds used to choose which components to remove, whether it be uniqueness in shape, noisiness, quick changes in value, or combinations of these. With the right identification, ICA could be quite useful in removing noisy artifacts even if their source isn't entirely known.

Discussion

Two key assumptions of ICA are that the hidden individual components of the signal that we are trying to compute are both statistically independent and non-gaussian. We need to assume that the individual components are statistically independent otherwise this implies that the one individual component signal provides information about another individual component signal. In terms of EEG recordings, this would imply that a signal from the brain's activity would give us information about blinks, or vice versa. This doesn't make any sense as brain activity is not indicative of blinks and vice versa. One could argue brain activity in a part of the brain could be indicative of a blink; however our goal isn't to discover this, rather to separate a mixed signal of blink and brain activity.

One improvement to this project that our team plans on implementing over this winter break is to further develop this program possibly into a web application or executable program. As of right now our program hardcoded the basic ICA and PCA algorithms with respect to the dataset we used; however, it doesn't have good general applications. So we plan on developing this software further to be able to provide a user with proper functionality to run ICA on a dataset according to the constraints of their goals. In order to do this successfully, not only would one need generalized functions for computing PCA and ICA, but one would also need a robust way to clean up inputted EEG data to the point where it is usable. As we learned, this would involve removing significant outliers, and also ideally accounting for other skewing elements such as drift and corruption. However, this is a whole other topic with extensive techniques and applications. Though it could be interesting to observe the specific trend of effect on how more or less cleaned data can affect the usefulness of ICA.

Another direction that we could have taken ICA with more time would be to attempt to reconstruct the location of identified artifacts using spatial and temporal data in tandem. Presumably by analyzing the differences in amounts and timing of independent components expressed at each node with respect to their spatial differences. UCSD actually has extensive resources provided on this subject from their EEGLAB within the Swartz Center for Computational Neuroscience.

Furthermore, there are many different nuances to how ICA can be implemented on an algorithmic level, with different implementations presumably being more or less effective for different datasets. This would be quite a fascinating topic to explore further.

References

- We used many of these websites to gather info about the project and figure out how ICA works

Dua, D. and Graff, C. (2019). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

Roesler Oliver, it12148 '@' lehre.dhbw-stuttgart.de , Baden-Wuerttemberg Cooperative State University (DHBW), Stuttgart, Germany. https://archive.ics.uci.edu/ml/datasets/EEG+Eye+State

Talebi, Shawn. (2021). Independent Component Analysis (ICA). *Towards Data Science*. https://towardsdatascience.com/independent-component-analysis-ica-a3eba0ccec35

Maklin, Cory. (2019). Independent Component Analysis (ICA) in Python. *Towards Data Science*. https://towardsdatascience.com/independent-component-analysis-ica-in-python-a0ef0db0955e

Tharwat, A. (2021), "Independent component analysis: An introduction", *Applied Computing and Informatics*, Vol. 17 No. 2, pp. 222-249. https://doi.org/10.1016/j.aci.2018.08.006.

Articles+Links for topics of further study

https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5915520/#!po=31.4433

https://eeglab.org/tutorials/06_RejectArtifacts/RunICA.html

https://mne.tools/stable/auto_tutorials/index.html