Sleep Stage Classification

ECEC 487
Dr. Anup Das and Dr. Nagarajan Kandasamy
Jacob Baron
Group 17

Abstract

One challenge facing medical specialists is the ability to diagnose a sleeping disorder by analyzing the patient's brain signals in real-time. Current solutions involve manually reading electro-cardiogram (ECG) signals or looking for other symptoms to determine if a patient has sleep apnea or sleep paralysis. One solution is to pair machine learning algorithms with neuromorphic hardware, so medical specialists can get real-time predictions of a patient's sleep health. The first step in this solution is to classify a healthy patient's sleep stage based on brain signals. This paper describes the techniques used to classify healthy patient's sleep stages through K Nearest Neighbor (KNN) algorithm by analyzing the brain signals and classifying into one of three stages: wake, rapid eye movement (REM), or non-rapid eye movement (nREM). The developed KNN outperformed a similar classifier created for a different application, but also underperformed against a state-of-the-art implementation.

Introduction

Many individuals are faced with the unfortunate reality that they have some sort of sleeping disorder, such as sleep apnea. Not only do sleep disorders prevent an individual from getting rest, but it also puts the individual at greater risk. This can put the individual at risk of abnormal heartbeat, heart attack, stroke and even cancer or type-2 diabetes [1]. The ability to detect and alert an individual that they are diagnosed with sleep apnea and provide positional feedback in real-time to allow the user to adjust their sleeping position can prove to be helpful in a hospital setting. Currently, sleep apnea is detected and quantified manually by monitoring a patient's breathing. This technique, however, can be quite ineffective as it requires an individual to check in on a patient and potentially disturb their sleep. Some studies have shown that sleep apnea may be detected using some features of electrocardiogram (ECG) readings. If true, this would significantly minimize the invasiveness of sleep apnea detection as ECG readings are easily obtainable and relatively nonintrusive [2]. A proposed solution is to implement machine learning algorithms onto neuromorphic hardware to produce accurate, real-time feedback that enables medical specialists to know if a patient is at risk of an adverse sleep condition based on their brain signals. In order to develop this solution, effective machine learning algorithms must first be developed. One approach of supervised learning used was the K Nearest Neighbor algorithm to classify healthy patients into one of three sleep stages: wake, nREM, or REM. Once an appropriate algorithm worked for healthy patients, the training data would introduce patients with various sleep disorders; however, that work is outside the scope of this paper. The purpose of this paper is to discuss the K Nearest Neighbor implementation and analyze its accuracy when classifying the voltage values of brain signals into one of the three class: wake, nREM, or REM.

A publicly available dataset known as the Cyclic Alternating Pattern (CAP) sleep dataset was used for the development of this algorithm. The CAP sleep data provided various brain signal data for 40 patients. The focus of this paper remains on healthy patients, which of the 40 patients provided, 16 were healthy. Each patient had multiple brain signal data for an entire night's sleep and a label dataset with labels corresponding to the sleep stage in increments of 30 seconds throughout the sleep. Between the signal dataset and the label dataset, each patient had approximately 1.1 GB worth of total data. More details on these datasets and how they were used in training and testing is provided in the Experimental Results section.
It is important to understand the significance the brain signals and to have some biological background in order to better understand the context in which this data exists.

As noted above, each patient had multiple signals recorded during the process. For example, patient one had 10 different signals recorded, each with a different name, such as, C4-A1 or P4-O2. These signal names correspond to a standard placement of "scalp electrodes in the context of an electroencephalogram (EEG) exam" [3]. The EEG tracks and records brain wave patterns by sending the signals to a computer which records the measurements for each placed node [3]. This system is commonly referred to as the 10-20 system [3]. The letter corresponds to the lobe of the brain and the number refers to the electrode position. "Even numbers refer to the right hemisphere" of the brain, while "odd numbers refer to the left hemisphere" of the brain [4]. For example, P4-O2 refers to channel between the Parietal Lobe in position 4 on the right side of the brain to the Occipital Lobe in position 2 on the right side of the brain [4]. This allows for a standardized communication and was important in the feature reduction when deciding which brain signal channels to use. Figure 1 shows the 10-20 system.

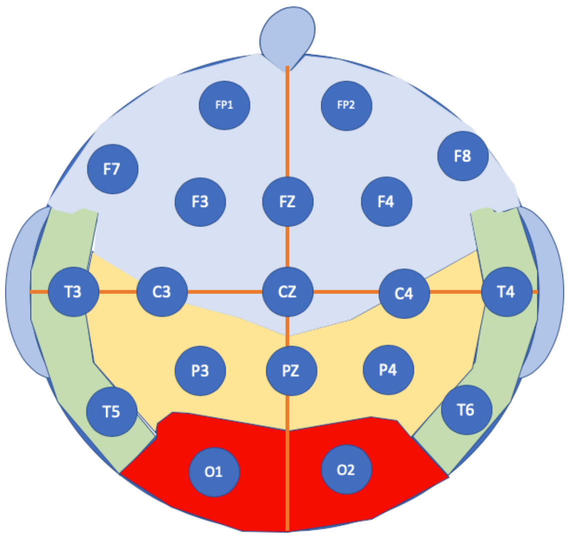

Figure 1. 10-20 System Layout. This figure demonstrates where the electrodes are placed on the head to capture the brain signals. This allows for a visualization of where the feature signals are coming from on the patient.

One of the challenges when deciding on what training data to use was determining which patients had common signal channels so that the algorithm can find patterns between the same channels across multiple patients. Figure 2 shows all of the brain signals provided for patient two.

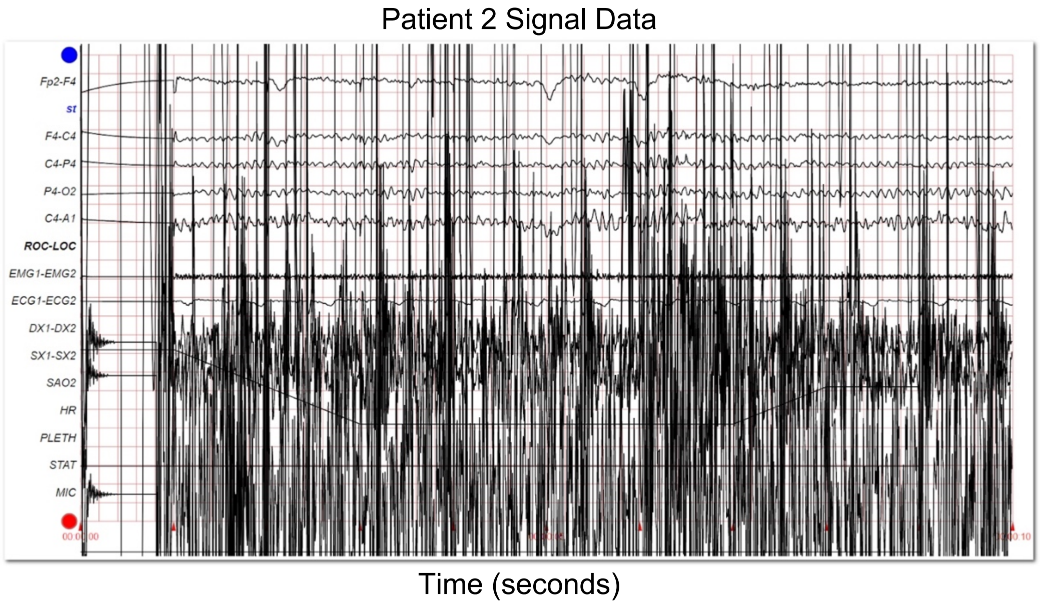

Figure 2. Brain Channels of Patient 2. This figure shows the provided brain signals as a function of time. Each patient has a comparable amount, creating great complexity in the data set.

Unfortunately, not every patient had the same signals. Figure 3 shows a bar graph of which signals were the most common amongst all of the healthy patients. From here, the total number of healthy patients used in the training data needed to be reduced since some patients did not have any brain signals that other patients had. For the purpose of simplification, signals that only had one patient were omitted from figure 3. This issue is a shortcoming of the dataset used. In practice, any brain channel could be collected from a patient. Ultimately, the signals that appeared the most frequently in the patients were used, reducing the total number patients down to 5. Each of these 5 patients was considered a "young adult", as their age was between 25 and 37 [5]. As described in Parrino's research, "The periodic arousal fluctuations reflected by CAP are a natural phenomenon of NREM sleep with specific age-related variations across the life cycle [5]." Therefore, it was important to ensure the patients' ages were in the same grouping. Further, the channels used were as follows: C4-A1, P4-O2, F4-C4, and C4-P4. The corresponding signal data for each of these four channels was the input to the machine learning algorithms.

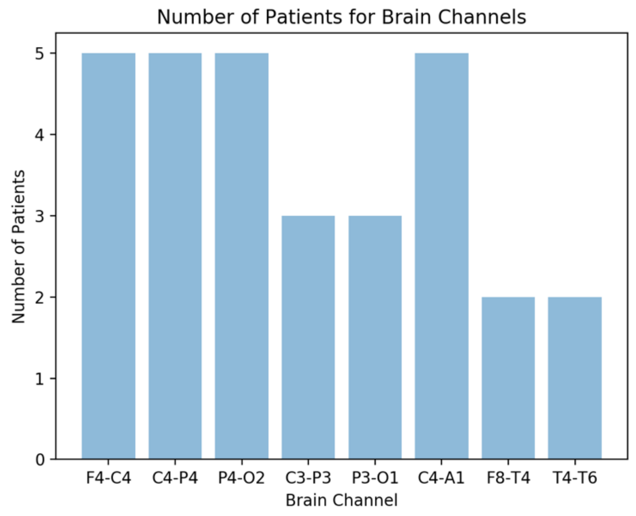

Figure 3. Distribution of Brain Channels. This figure shows the number of healthy patients that had each brain channel. Some channels only had 2 or 3 patients, so they could not be used. Channels corresponding to only 1 patient were omitted from this figure.

In total, there were approximately 17 million measurements per signal, where each patient had 4 signals. This resulted in nearly 340 million total data points across every signal in all patients. By analyzing the corresponding label to each signal data point, it was found that nREM and REM had the most data points, while sleep had the fewest. "The cyclic alternating pattern as a physiologic component of normal nREM sleep is expected" [6]. Since REM and nREM sleep stages cycle throughout a period of sleep, it is obvious that these would provide the most data in terms of number of signals corresponding to the labels.

Related Work

The dataset that was used for this project has been commonly used by other researchers to solve problems related to sleep disorders and sleep cycles; however, not many research papers have been published regarding an overall sleep classification including nREM, REM, and wake.

In particular, one research project investigated an "automatic approach to detecting the activation phases of the CAP sleep phenomenon through the similarity index of the EEG signals" [7]. This research project used the same sleep CAP database; however, in their experiments, they used all 16 healthy patients in the hopes of achieving a different result: to find the activate phase of the CAP sleep pattern cycle. They needed to identify a nREM period followed by a REM period lasting less than one minute. For this use case, using similar brain channels was not as important, as they were not aiming to classify the sleep stage based on a common data, rather they aimed to find a pattern that is not provided by the dataset. In conjunction with the time series brain signals, the project used a concept known as the "similarity index" which "can be obtained by cosine distance of feature vectors" [7]. The paper reached a maximum of 81.87% accuracy through the use of matching similarity index values to known activation periods [7].

Another research paper that attempted sleep analysis using ECG signals was "A Review of Approaches for Sleep Quality Analysis" which generated metrics based on the widely used CAP sleep database and performed machine learning techniques on the data and the metrics to draw conclusions about the quality on a patient's sleep [8]. One of the experiments conducted in this paper was applying "three machine learning approaches for measuring sleep quality by applying the discriminative graph regularized extreme learning machine, the K Nearest Neighbor, and a Support Vector Machine [8]." The findings of the paper show that with *k=1*, the KNN's highest accuracy was 37% [8]. Despite a low performance from the KNN, other approaches used in this paper had accuracies that maximized at 87% in different use cases; however, the KNN's maximal performance across several use cases was the 37% described above [8]. Although their project focuses on using the KNN to classify a patient's sleep as good or bad based on metrics and not classifying the sleep stage like the application in this paper, the use of a KNN was intriguing since it performed poorly and provided a cautionary tale when moving forward with this new implementation.

A final comparable work that relates closely to the content of this research was the "Classification of Sleep Disorders Based on EEG Signals by using Feature Extraction Techniques with KNN Classifier" [9]. This paper was the inspiration for using the KNN classifier, as they attempted to classify a patient as healthy or sick with the KNN classifier [9]. This paper detailed that prior to providing data into the KNN classifier, they performed Independent Component Analysis and Linear Discriminant Analysis on the data [9]. The paper also describes that for training the classifier, they used four healthy patients and four sick patients. For testing, they used five healthy patients and five sick patients. The results indicated a 70% accuracy using the KNN classifier in most cases but did achieve as high as 80% when changing which patients were used for training versus testing [9]. In this case, it misclassified two sick patients as healthy patients, which can lead to philosophical questions regarding whether it is better to misclassify a sick patient as healthy, or vice-versa. Nevertheless, the solid performance was encouraging, as the other implementation of KNN noted above performed poorly. It is important to realize that the KNN classifier in their research paper was used for binary classification, sick or healthy, whereas the KNN classifier detailed in this paper is used for a three-class system.

Experimental Results

Prior to developing the machine learning algorithms, the data needed to be provided in a way such that the algorithms could understand it. For each of the 5 patients, there were 512 signals every 1/512th of a second. Each recording was measured for about 9 hours, resulting in a very large number of data points. Unfortunately, the signal data and the label data were two separated datasets. The signal dataset provided the time in which the signal gathering started, but the measurements themselves were index with elapsed time. The label dataset, however, provided one label for every 30 seconds. The label dataset also provided the start time and the end time. It quickly became clear that the signal dataset began recording before the label dataset started providing labels. On average, the two datasets were out of sync by 300 seconds. For some patients, the signal dataset also extended beyond the time the label dataset finished providing labels. In each case, the signal dataset needed to be normalized to line up with the label dataset so that each measurement had a corresponding label. Although overwhelming at first, the math was quite simple. If it is known that there are 512 measurements every 1/512th of a second, then the number of measurements for per second can be calculated through dimensional analysis. Since the label dataset was fully contained within the signal dataset, then the signal dataset simply needed to have the additional data before the label dataset began and the additional data after the label dataset finished removed. The start and end time of the label dataset was known, and in conjunction with the newly calculated measurements per second, the exact number of measurements before and after the label dataset could be removed. Although this was not a necessarily difficult calculation, it was a tedious factor to include when dealing on a dataset of this magnitude. It also reduced the total number of data points for each patient.

Once all of the data was collected, the labels were then attached to each measurement by associating all measurements within the 30 seconds of the label to match that label value. Figure 4 shows the first 10,000 data points of patient two after the data processing. These data points correspond to the WAKE class.

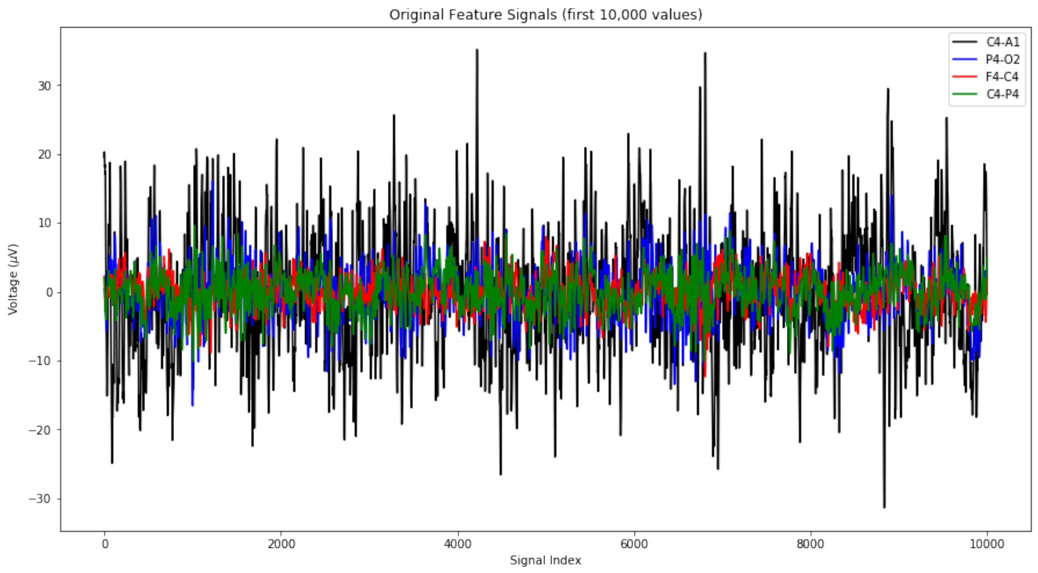

Figure 4. Sample of WAKE class before filter. This figure shows a sample of 10,000 data points from patient 2 that were labeled as WAKE. The noise and low appearance of separability made it obvious that data preprocessing was necessary prior to training the classifier.

From this figure, it was immediately noted that the signal data was very irregular. To smooth the data, a Savitsky-Golay filter was applied. A Savitsky-Golay filter was used because it is often used as a digital filter for the purpose of smoothing [10]. It aims to increase the precision of the data without distorting the signal tendency [10]. This was incredibly important for this experiment, as the signal tendencies are amongst the most important aspects in identifying which class that signal corresponds to.

Figure 5 shows the same 10,000 data points, but with the filter applied. It clearly reduces a significant amount of noise, and the patterns between the signals can be more easily observed. Since this corresponds to the wake class, this provides a nice visualization of what a typical wake stage might look like. From analyzing the figure, the wake stage tends to have channels that are not as regular, with frequent spikes throughout. The voltage ranges from -6 microvolts to 7 microvolts. Further, C4-A1 greatly exaggerates the patterns across the remaining three channels.

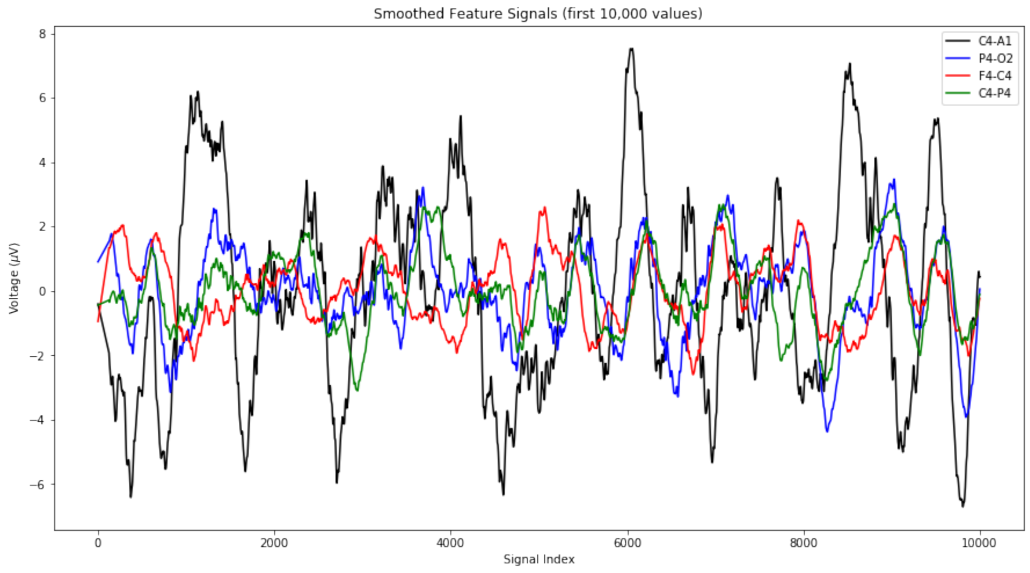

Figure 5. Sample of WAKE class after filter. This figure demonstrates the contrast of the data after a filter was applied. The waveforms become more visible and patterns across the 4 waveforms become apparent. Overall noise is reduced, leading to a cleaner looking signal.

Once a wake stage was visualized and cleaned with a filter, it was also critical to identify what a nREM and a REM stage looked like. Figures 6 and 7 show 10,000 data points for each of the nREM and REM, respectively.

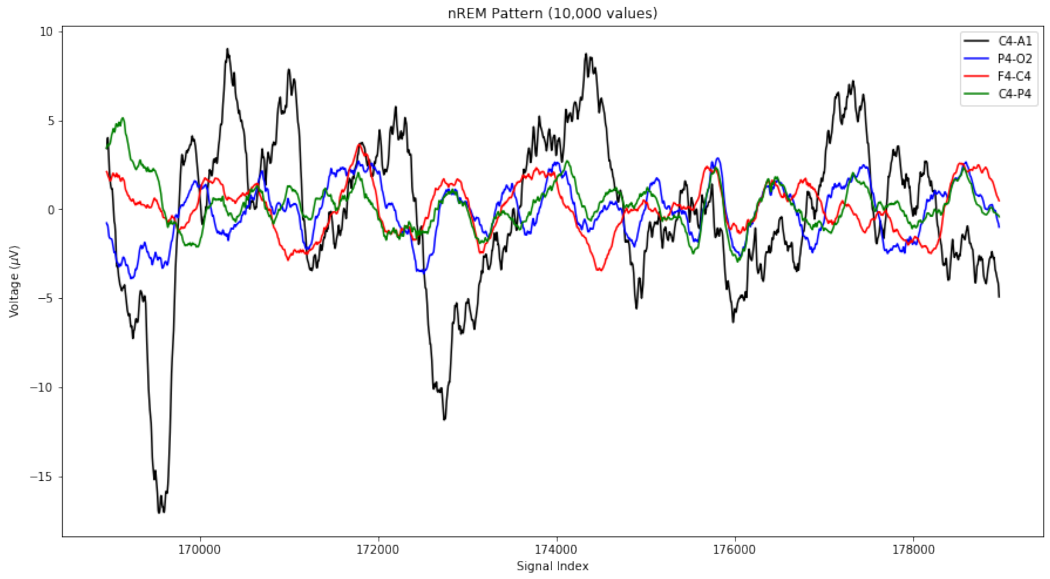

Figure 6. Sample of nREM class after filter. This figure provides a sample of 10,000 data points from the nREM class after a filter was applied. P4-O2, F4-C4, and C4-P4 remain almost the same over the course of the nREM class and remain below 5 microvolts.

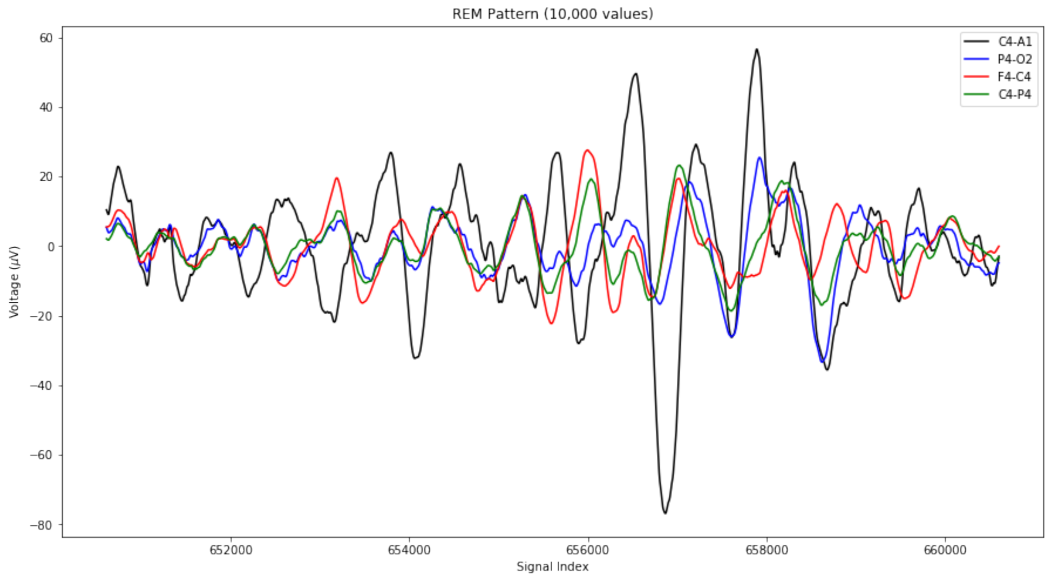

Figure 7. Sample of REM class after filter. This figure provides a sample of 10,000 data points from the REM class after a filter was applied. All channels overlap in a similar fashion for most of the REM cycle. C4-A1 does deviate at points.

As noted in figure 6, the nREM pattern tends to have the P4-O2, F4-C4, and C4-P4 signals closely match throughout the cycle. The C4-A1 channel, however, deviates significantly from the pattern of the other three. This is an interesting revelation, as during nREM, C4-A1 closely resembles the C4-A1 channel of the wake class. This would make sense, as nREM is the sleep stage in which the body moves the most [1]. A patient may switch body positions, while also some muscles may begin twitching [1]. From this information, C4-A1 correlates closely with how the body moves during a sleep period. It can also be noted that the channels outside of C4-A1 never exceed positive or negative 5 microvolts. This is different from the wake stage as those three brain channels were contained within positive or negative 2 microvolts.

Figure 7 shows a typical REM pattern. The first observation that can be made about this stage is that the brain channels are significantly smoother than that of wake or nREM. There are fewer spikes in the channels and all four channels tend to follow the same pattern, with a few exceptions from C4-A1; however, these data points were the first 10,000 data points of the REM cycle, so the patient may not have fully settled in. Research has shown that REM is considered the deepest point of a patient's sleep, often resulting in dreams and low body movement [1]. That can be seen based on the range of brain channels. Here, the range is -20 microvolts to 20 microvolts, with a few exceptions from the C4-A1 channel. Nonetheless, the higher brain activity corresponds to the dreaming phenomenon that occurs during the REM cycle. Further, since muscle and body movement are typically stopped in the REM cycle, the channels are not spiking as the movement is no longer sudden [1]. The patient remains in the same position for a longer period of time. The spikes in C4-A1 might correspond to a sudden movement by the patient, but overall is significantly reduced when compared to nREM and wake.

Now that the patterns the classifier needs to look for have been established, the architecture of the K nearest neighbor needed to be established. The KNN classifier is a form of supervised learning which accepted the 4 brain channels as input data, along with a label for each of the three classes. Since the three stages of sleep each provided distinct features to their patterns, the KNN classifier was thought to work well, as it analyzes data points near it and determines which class to assign it. The K value represents how many neighbors to use. When building the KNN, there was no proven way to determine for sure which K value would work the best. Instead, K was abstracted as an input parameter such that the user of the classifier can modify the K value and see the change in accuracy. In addition to the number of K nearest neighbors to use, the classifier accepted two other attributes: a weighting function and a nearest neighbor algorithm.

The weighting function is important to the classification of the data point. In a case where k is greater than 1, different weights are assigned to each of the nearest neighbors; that is, each neighbor does not have an equal weight. The weighting function determines how to assign a weight to each neighbor. The classifier was built such that the user can select one of two built-in weighting functions or provide their own user defined one. For this experiment, the two built-in functions, uniform and distance, were used to see the change in accuracy based on the weighting function. Uniform is the simpler of the two – it assigns an equal weight to all nearest neighbors. Distance, however, assigns a weight that is inverted to the calculated distance between the pending point and its nearest neighbors. This means that the closer the data point is to a nearest neighbor, the higher the weight. When calculating the distance, the Euclidean Distance algorithm was used.

The third component of the KNN classifier is the nearest neighbor algorithm, which describes how the nearest neighbors are determined. The classifier had four options: “Auto”, “Ball Tree”, “KD Tree”, and “Brute Force”. “Auto” makes a decision on which algorithm to use. “Auto” was used to simplify the number of times the classifier needed to be trained.

The KNN classifier was trained with 80% of the input data, which totaled approximately 100 million total data points used for testing, as each patient had approximately 14 million data points. The test data was then equivalent to one patient, at 68 million data points. Figure 8 shows the results as k increases for both weighting functions. It is interesting to note that the distance weighting function created a higher accuracy across all test cases; however, performed lower than uniform in a higher percentage of total test cases. This seems to imply that on average, the uniform weighting function is better for this application in most cases; however, performed slightly worse than the distance weighting function in the best case. Figure 8 also shows a steady decline in performance after 3 nearest neighbors were used. This may be due to overfitting of the data, as too many nearest neighbors were being used to make a classification. The best performance was k=3 with the distance weighting function, providing an accuracy of 68.24%.

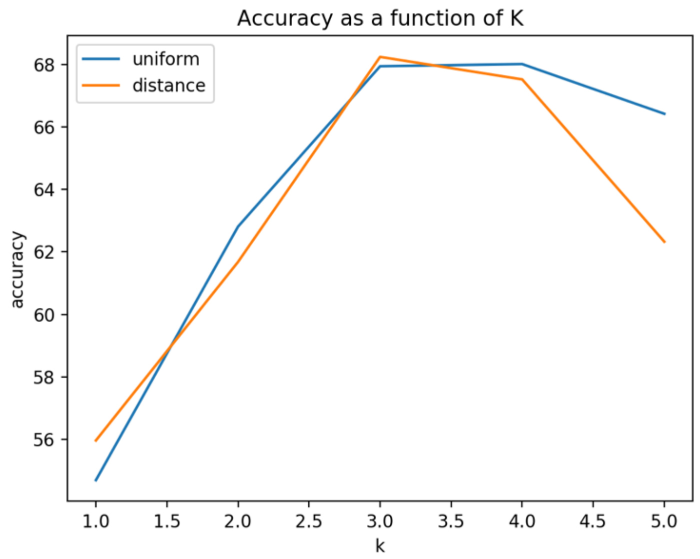

Figure 8. Accuracy of KNN as K increases. This figure provides the accuracy values for k on the integer range [1,5] while using two weight functions: distance and uniform.

Once the best performance was determined, additional figures were generated to visualize the results. Figure 9 demonstrates the total testing dataset being classified into one of the three classes: wake, nREM, and REM. The figure clearly shows that the nREM is the most frequently classified stage. nREM was classified for 57.5% of the total data set. REM was classified for 32.7% of the total data set. Wake was classified for the remaining 9.8% of the data set. In terms of the distribution of classification, it mostly matches the expected distribution of cycles. Since the data provided an entire night’s sleep, wake should be classified the least frequently. nREM is a longer cycle than REM, so it would follow that nREM is classified more frequently than REM [1].

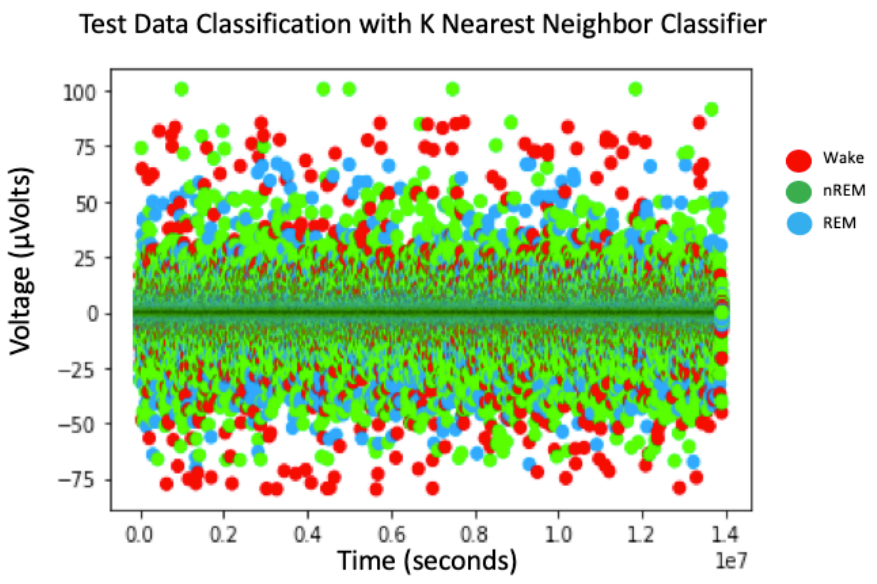

Figure 9. Data Classified after KNN. This figure shows the class each data point was classified into. As we can note, nREM was the most frequently classified class.

Figure 10 shows the confusion matrix for the highest performance of the KNN classifier. Class 0, which corresponds to wake, only classified the correctly 41.11%. This was mostly likely due to the wake having the least amount of data points associated with the label. More data for wake is necessary for improvement. Class 1, which corresponds to nREM, performed correctly 74.23%. It misclassified nREM as REM three more times as frequently as it classified nREM as wake. This makes sense since nREM and REM are closely related. Class 2, which corresponds to REM, performed correctly 65.68%, classifying REM as nREM 8 more times as frequently as it classified REM as wake. Overall, the classifier worked quite well with determining the difference between REM and nREM but struggled to classify wake correctly. This most likely due to the relatively low number of data points corresponding to the wake stage.

	Class 0	Class 1	Class 2
Class 0	549,892	553,997	170,405
Class 1	531,254	5,932,973	1,400,135
Class 2	256,317	1,507,359	3,005,908

Figure 10. Confusion Matrix of Best Performance. This figure details the number of correct and incorrect classifications of each class. Class 0 is wake. Class 1 is nREM. Class 2 is REM.

Discussion and Conclusions

One of the major takeaways from both the experimental results and the work other researchers have conducted is that greater than 90% accuracy is very difficult to achieve with this dataset. As seen with the K Nearest Neighbor approach, the highest accuracy achieved in this experiment was only 68.24%. K Nearest

Neighbor is a good algorithm for a dataset with a small number of features, but a large amount of data points [9]. A good way to determine the effectiveness of the classifier is by comparison to existing work.

In the best-case, the KNN classifier operated at a 68.24%, with a worst-case performance of 45.54%. Nonetheless, this range of accuracy exceeded the KNN classifier described in the “A Review of Approaches for Sleep Quality Analysis” paper where their KNN classifier performed at 37% [8]. Unfortunately, the paper did not detail the architecture of the KNN classifier, so it is not possible to compare the inner workings of the classifier; however, the paper does state that a *k* value of 1 achieved their highest accuracy [8]. The KNN classifier used above did not perform well under *k=1* situation, regardless of the weight function.

The other well regarded KNN classifier used on this same dataset was the classifier created in the “Classification of Sleep Disorders Based on EEG Signals by using Feature Extraction Techniques with KNN Classifier” paper. In that paper, the KNN performed with a 70-80% accuracy when classifying 10 patients as healthy or sick [9]. The paper concluded that *k=4* was the optimal k value. The paper did provide some architectural information; however, it focused more on feature extraction and data preprocessing techniques [9]. They applied Independent Component Analysis and Linear Discriminant Analysis on the data as opposed to a Savitsky-Golay filter [9]. The benefit to the filter is that it has lower overhead if this algorithm were to be implemented in hardware and could even help improve the classifier overtime by smoothing the data as it is read into the classifier. Linear Discriminant Analysis may be a benefit as the data, despite appearing non-linear, might have linear relationships outside the pure voltage values. For example, there may be a linear relationship across the Fourier Transform Coefficients that is not obvious from looking at the waveforms.

With achieving just under 70% accuracy, there is room for improvement with this algorithm. As noted above, the KNN classifier was capable of accepting a weight function used in the prediction. In the experimentation, two weight functions were used: uniform and distance. The uniform weight function gave an equal waiting to all nearest neighbors, whereas the distance weight function gave an inverse weight to the distance between the nearest neighbor and the current data point. The classifier can also accept a user defined weight function. Although that was not part of this investigation, it might be worth looking into in the future.

In addition to weight function, the classifier also had an algorithm feature such that the caller can provide the algorithm used to compute the nearest neighbor. As noted in the Experimental Results section, this option was set to “auto”, which decided what the most appropriate algorithm to use based on the input data. The options it selected from where “Ball Tree”, “KD Tree”, and “Brute Force”. Perhaps with better computational equipment, a brute force algorithm might provide higher accuracy; however, the inherent tradeoff of equipment and time is important when discussing the application of these classifiers inside a medical environment.

A third tunable feature of the KNN classifier was the distance metric function. A user can create their own or use one of the built-in functions. For this investigation, the Euclidean Distance metric was used in order to simplify the number of times the classifier needed to be trained, as each training session took between 30 and 45 minutes when spread across 10 threads. In the future applications of this classifier, developing a user created distance metric might also provide better accuracy; however, since the data is nicely represented in two dimensions, Euclidean Distance may be hard to beat.

This experimentation is not the end of the project. Aside from potential accuracy improvement of the current algorithms, a random forest algorithm might also be a viable option. Originally, random forest was a planned algorithm for this paper, but due to the time complexity of the algorithm on the current dataset and limited computational hardware, this option was not feasible as the algorithm took too long to process. Even splitting the workload across 10 threads, the random forest classifier had not finished training the data after 7 hours of operation.

In conclusion, classifying sleep stages based on various brain channel signal data is possible and works to some degree of success; however, it does not achieve a remarkably high level of success, indicating that either more patients with the same brain channels are needed. Although the K Nearest Neighbor algorithm outperformed some literature, there is still room for improvement before use in a medical setting as it underperformed against a state-of-the-art classifier. In particular, the KNN classifier performed poorly when classifying a patient as awake. More training is required for this stage in order to better tune the classifier. A classifier with accuracy less than 80% creates too high of a risk of misclassification and can lead to misdiagnosis, so the KNN still needs to improve its misclassification rate.

References

[1] P. Fonseca, X. Long, M. Radha, R. Haakma, R. M. Aarts, and J. Rolink, “Sleep stage classification with ECG and respiratory effort,” *Physiological Measurement*, vol. 36, no. 10, pp. 2027–2040, Aug. 2015.

[2] N. Cooray, F. Andreotti, C. Lo, M. Symmonds, M. T. Hu, and M. D. Vos, “Detection of REM sleep behaviour disorder by automated polysomnography analysis,” *Clinical Neurophysiology*, vol. 130, no. 4, pp. 505–514, 2019.

[3] R. W. Homan, J. Herman, and P. Purdy, “Cerebral location of international 10–20 system electrode placement,” *Electroencephalography and Clinical Neurophysiology*, vol. 66, no. 4, pp. 376–382, 1987.

[4] R. Caton, "The Electric Currents of the Brain", *Br. Med. J.*, vol. 2, pp. 278.

[5] L. Parrino, M. Boselli, M. C. Spaggiari, A. Smerieri, and M. G. Terzano, “Cyclic alternating pattern (CAP) in normal sleep: polysomnographic parameters in different age groups,” *Electroencephalography and Clinical Neurophysiology*, vol. 107, no. 6, pp. 439–450, 1998.

[6] MG Terzano, D Mancia, MR Salati, G Costani, A Decembrino, L Parrino. “The cyclic alternating pattern as a physiologic component of normal NREM sleep”. *Sleep*, vol. 8, no. 2, pp. 137-145, 1985.

[7] Hamid Niknazar, Saman Seifpour, Mohammad Mikaili, Ali Motie Nasrabadi, Anahita Khorrami Banaraki, "A novel method to detect the phases of Cyclic Alternating Pattern (CAP) using similarity index", *Electrical Engineering (ICEE) 2015 23rd Iranian Conference on*, pp. 67-71, 2015.

[8] Fábio Mendonça, Sheikh Shanawaz Mostafa, Fernando Morgado-Dias, Antonio G. Ravelo-García, Thomas Penzel, "A Review of Approaches for Sleep Quality Analysis", *IEEE*, vol. 7, pp. 24527-24546, 2019.

[9] D. V. Dhongade and T. Rao, “Classification of sleep disorders based on EEG signals by using feature extraction techniques with KNN classifier,” *2017 International Conference on Innovations in Green Energy and Healthcare Technologies (IGEHT)*, 2017.

[10] R. Schafer, “What Is a Savitzky-Golay Filter? [Lecture Notes],” *IEEE Signal Processing Magazine*, vol. 28, no. 4, pp. 111–117, 2011.