PhD Preliminary Exam
Department of Electrical Engineering
Temple University

Christian Ward

3 December 2015

List of Figures

List of Tables

Contents

1 Introduction

The focus of my research is to facilitate the development of software capable of automatically cataloging Electroencephalogram (EEGs). The ultimate goal is the creation of an open search-by-signal database of EEG records. A database is primarily a referential tool intended to function when relationships are applied to the data, or in the case of EEGs classes of assorted feature sequences, that enable quick reference between stored content. To achieve this end it is fundamental to understand the development of *relevant primitives* or *features* derived from clinical domain knowledge and existing signal analysis techniques. The methods used to establish robust universal features will inform the classification method used to query the database records. With well chosen features naturally creating a hierarchy for storage and search, novel patterns of EEGs should emerge that improve the speed with which diagnosis can occur while providing new metrics to evaluate previously undefined signals against.

A large amount of work related to feature manipulation resides in literature pertaining to speech processing which serves to inform this paper. Equivalent developmental steps once used to narrow the knowledge deficit in speech guide the way for progress in the realm of EEGs. The full process of collecting EEG features, evaluating features, and deploying and training appropriate classifiers is taken directly from the speech community. While not perfect, it boosts the rate at which the knowledge gap between supervised and unsupervised techniques can be narrowed. The results of this research will fill in fundamental knowledge gaps about how the brain works as a system.

This document considers major work in the realms of feature specific domain knowledge, feature manipulation, and classifier development to present critical understanding of the EEG signal processing pipeline. Research highlighting the importance of melding domain knowledge with advanced analytical techniques, brute forced feature components designed to mitigate the curse of dimensionality, and assessments of the latest in classifier development will be discussed at length. As insightful as the presented works are, they are not exhaustive, but instead serve to highlight fundamental developments in the areas of interest related to my PhD research.

2 Mining Event-Related Brain Dynamics

A main component of understanding electroencephalograms is the ability to correctly map a response to a stimulus. In numerous studies, subjects are directed to perform actions or triggered into specific responses to allow for closed loop testing easily duplicated across subjects. This ranges from responses to real and imagined motion, event-related triggers such as sounds or visual cues, physical external stimulus of tactile stimulation, or altered states of mind through the use of drugs. Each approach provides insight into the functionality of the brain, but the complexity of correct interpretation varies greatly amongst each approach. The work of Scott Makeig et al in “Mining Event-Related Brain

Dynamics" examines a novel way to address categorizing responses from event-related potentials (ERPs).

Event-related potentials (ERPs) are responses of the brain triggered by external visual or auditory stimuli. A common and somewhat novel use of this response is seen in P300 spellers. A P300 speller operates by tracking a latent 250 to 500ms response over the parietal lobe through the *oddball paradigm*. Subjects are asked to search out the letters for a specific word and when that letter is flashed before them, their brain responds to 'finding' the letter by producing a spike in their parietal lobe. [8]

This sequence of events allows for researchers to link changes in the subject's brain waves together. The stimulus provides a common index in time for each subject's response because the delay time is precise across trials. The work in Makeig's paper goes beyond an analysis of the single channel response of P300 in an effort to discern the changes in the subject's entire brain wave state given the ERP. The aim is to develop a better methodology for mapping and modeling the state of a brain in response to an ERP. A lack of congruence between the two dominant analysis strategies prevents a full dynamic model of the brain's response to the ERP. The two approaches used (1) average the time responses to specific trials to build a class response or (2) average the full frequency response of the subject over the duration of the ERP. By combining the ERPs and the frequency based approach, event-related spectral perturbation (ERSP), their work produces a state-space response map.

2.1 Feature Fundamentals

2.1.1 Event Related Potentials — ERPs

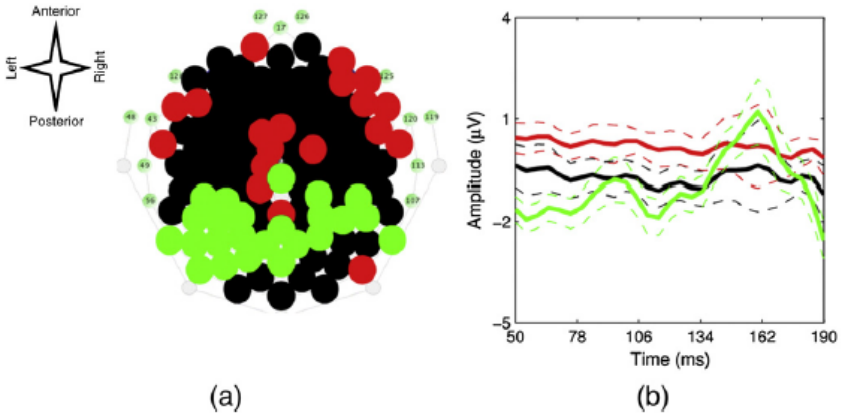

Figure 1: Example of ERP recording from an image recognition trial. [8]

Notice the P300 response in green spike near 162ms. The coloring indicates brain region clustering via the Quality Threshold clustering algorithm.

ERPs are a response to sensory stimulus, shown in Figure 1, where their expression can be seen as both *bottom-up* and *top-down*. Early studies believed in the *bottom-up* theory where the peak was the manifestation of activity deep in the cortex that had no further impact on the brain. Changes outside the evoked peaks were considered nothing more than random background noise and discarded in the analysis. The original *bottom-up* premise relied on the brain being in a static state waiting to be perturbed by the stimulus. In truth, the base state of the brain is very active and the interplay of this activity is what triggers the ERPs, but can obscure ERPs because of signal mixing at the site of the electrodes.

In contrast, a model that combines *bottom-up* with *top-down* for neural activation can capture the true impact of a stimulus response. This is because the temporal proximity of cortical activity can contribute to multiple aspects of the overall response, usually beyond the main ERP seen. A *top-down* view takes into account that the visible response layer is impacting and impacted by other temporal events. Previous ERP studies average the response over numerous trials, which effectively eliminates dynamic information pre- and post-stimulation which hides such *top-down* features.

The problem in focusing on just the peak responses, in terms of their delay and location after sensory stimulus, ignores dynamic information before, during and after the ERP. Continued application of this approach reduced ERPs to nothing more than timing markers, when larger factors were in play to produce them and occurring because of them. While useful in the application of novel spellers, there true power was being subjugated by uninformed physiological understanding.

2.1.2 Event Relates Spectral Perturbations — ERSP

The ERSP captures the dynamic environment of the brain as a temporal power spectrum map in response to sensory stimulus. Where the ERP captures a single event, the ERSP represents the frequency spectrum at each electrode highlighting the mixing of cortical sources. Changes in brain activity are captured to show the propagation of waveforms throughout the brain over the duration of the ERP. This feature evolved from *event-related desynchronization*, ERD, which was used to describe a shift in cortical arousal via spatial and temporal synced rhythms being replaced by faster spatially differentiated rhythms. These events are noticeable when subjects shift from eyes closed to eyes opened, as the alpha-band sees a decrease over the occipital lobe. From an ERP view, this would appear as more background noise not related to stimulus response.

Providing the frequency content at a given electrode does not intuitively detail a response to the sensory stimulus, but places the response in the context of the brain state. Difficulty arises in understanding these maps to determine where a source is located given that the shifting of frequencies is unclear without a reference point for the subject. It shows how the frequency activity is shifting over time, providing insight for the *top-down* view and capturing the dynamic environment ignored by the ERPs.

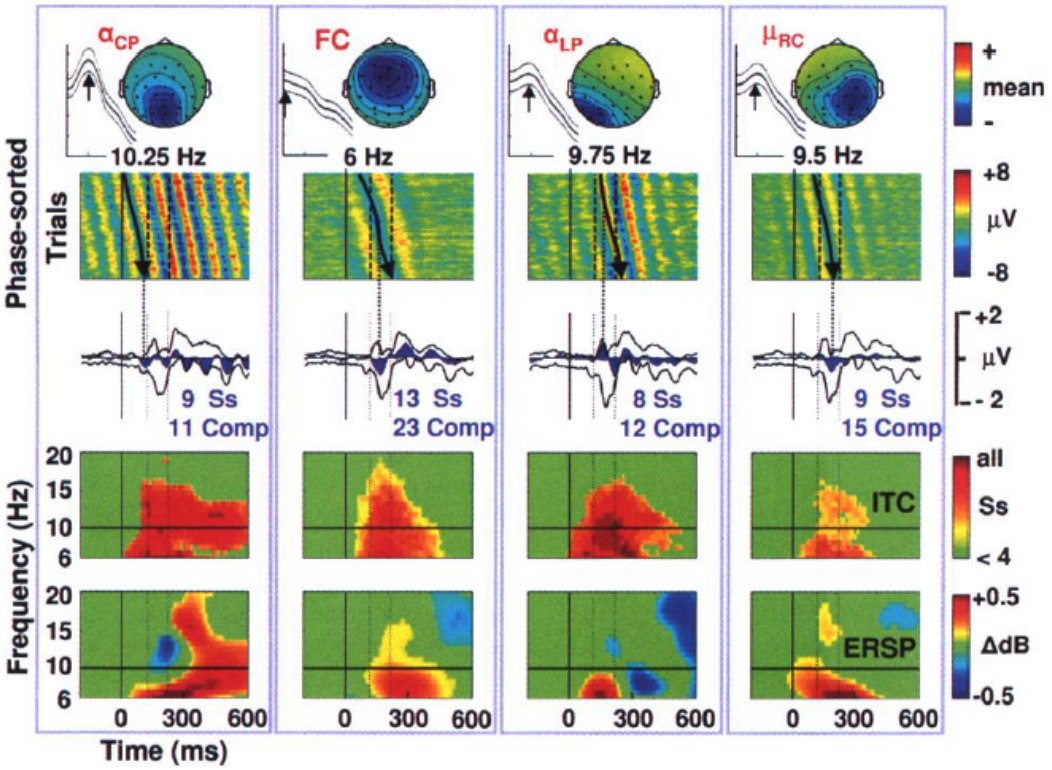

Figure 2: Combination of ERP, ERSP, and ITC results. [5]

This data comes from Makeig's paper that developed ERSP. The ERP is plotted in the middle.

Makeig et al introduced this feature in an earlier work to compensate for the narrow-banded ERD. The goal was to measure the relative changes across the brain being induced by the stimulus and other latent features. The resultant data is three dimensional providing time versus frequency with intensity mapped in the third dimension as the log of the EEG amplitude seen in Figure 2.

2.1.3 Inter-Trial Coherence — ITC

Originally called the *phase-locking factor* , ITC is a two dimensional image, phase vs. latency, of how well the EEG signals phase lock to the time-locking events. Changes in the phase will not be visible through the ERSP because it only captures magnitude. However, it is known that changes in phase can impact the ERP because the it is sensitive to the polarity of the EEG. Even a subtle adjustment in phase, if occurring across a large enough frequency band can trigger additional spikes or cancel out previously anticipated spikes. As the spatial mixing occurring at each electrode prevents a true understanding

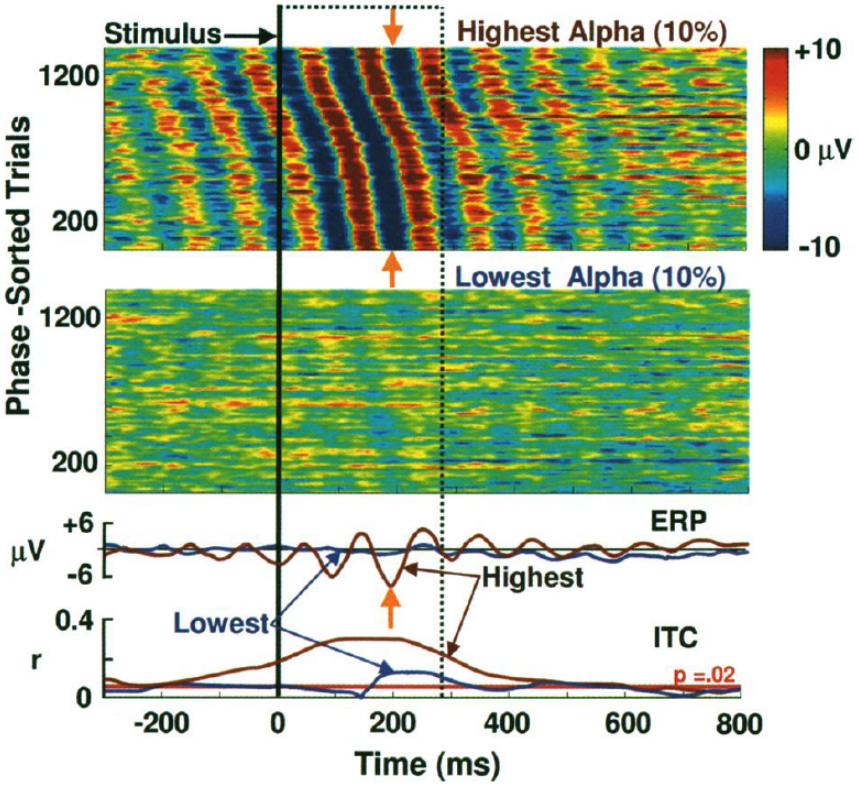

Figure 3: Response to visual stimulus across ERP, ITC, and phase.[5]

Two averaged response are plotted, one is high-alpha (brown) and the other is low-alpha (blue). The phase trials are taken from the subject's responses used to make the average ERP and ITC plots.

of how the waveforms are contributing to the recorded signal.

To resolve how the waveform are mixing, constructively, destructively or not all, calculating the phase at each electrode provides insight into the direct of the changes in the frequency amplitude shown in the ERSP. In turn this leads to understanding how the ERPs are being produced in terms of identifying which frequencies are responsible of the change and where they originated from in the brain. With all three measures being recorded prior to and after the targeted ERP, the full picture of the dynamic brain state can be understood in the context of the stimulus.

Figure 3 presents a response to the subject's left visual field. In both cases a similar pattern emerges where sigmoidal striations appear in the phase-sorted trials windows. The stronger alpha wave, at 10.25Hz, response produces a stronger ERP and ITC as the majority of the phase change is negative. The lower alpha response produces a smaller change corresponding with the weaker phase change associated with its subjects. Each

of the $>$ 1200 trials was averaged to make the modeled ERP and ITC waves, but the details from the phase plots illustrate how the change is being generated.

2.2 Feature Selection

The information in the time, ERPs, and the information in frequency, ERSP, maps needs to be decomposed to find the distinct components that compose the signals. By using ICA to act as a spatial filter they are able to reduce the 69 scalp electrodes down to 20 maximally independent components. An additional two channels are added to address artifacts, from eye artifacts via a bipolar diagonal electrooculogram (EOG), and a ground on the right mastoid for general muscle contractions. The 20 maximally independent components is a higher threshold across all the data, because they need to resolve each component back to a linear independent set of the electrodes. If a combination of the electrodes returned as a component, with their associated weights, created a linear dependency within the 20 it would be removed.

From the resultant 20, or less, components found for each response a projection of the single dipole or bilaterally symmetric dipole pair can be plotted with respect to electrode probes. The chosen electrodes' data is then turned into an ERP, ERSP, and ITC plot to provide the full projection of the stimulus response. These 20 components are actually 20 representations of unique responses from sensory stimulus.

2.2.1 Independent Component Analysis — ICA

ICA allows for the multichannel data to be decomposed via its distinctiveness by finding all the non-Gaussian signals in the data. The behavior of channels of interest is assumed to be statistically independent and as such it should appear as distinct from the common 'idle' state of the brain. It relies on *eigenvalue decomposition* to *whiten* the data of interest as it iteratively searches for the most distinct components from the average of the signals present. Results from the algorithm promise no guarantee of their ordering nor importance so there is still some liability on the user to interpret the resultant components.

These resultant components come in the form of a matrix $\mathbf{W}$ that models the relationship between the input vector $\mathbf{x}$, and an offset vector w. These are used to transform $\mathbf{x}$ into a statistically independent vector $\mathbf{u}$. This vector $\mathbf{u}$ represents a statistically independent set of the N features presented in $\boldsymbol{x} = [x_1 ... x_N]$.

$$\begin{aligned} \boldsymbol{u} &= \boldsymbol{W}\boldsymbol{x} + w. \\ &\text{where} \\ \boldsymbol{u} &= [u_1 ... u_N]^T. \\ \boldsymbol{x} &= [x_1 ... x_N]^T. \end{aligned} \tag{2.1}$$

ICA attempts to ensure that $\mathbf{u}$ has minimal *mutual information* between each element. This is exemplified in 2.2, where the summation of all the elements of $\mathbf{u}$ must equal the probability density function of the entire vector. If elements in $\mathbf{u}$ contained mutual information the product would exceed the vector's probability from the redundant components not contributing to the growth of the distribution, see 2.3. The function $f_u(q)$ represents the probability density function of the variable q and the function $F_u(q)$ is the cumulative density function of variable q.

$$f_{\mathbf{u}}(\mathbf{u}) = \prod_{i=1}^{N} f_{u_i}(u_i). \tag{2.2}$$

$$\begin{aligned} \text{ICA: } I(u_i, u_j) &= 0, \forall ij. \\ \text{PCA: } \langle u_i u_j \rangle &= 0, \forall ij. \end{aligned} \tag{2.3}$$

Operating under an assumption that the components sought exhibit the same probability density function, the generic form of [2.1] can be updated to find the maximum entropy $H(\boldsymbol{y})$, where $\mathbf{y} = F_u(\boldsymbol{u})$ is a non-linearly transformed vector. An equation for the stochastic gradient ascent of both $\mathbf{W}$ and w is shown in [2.4] based upon the entropy equation.

$$\Delta \mathbf{W} \propto [\mathbf{W}^T]^{-1} + \hat{\mathbf{y}}\mathbf{x}^T, \Delta w \propto \hat{\mathbf{y}}. \tag{2.4}$$

The result turns the elements of $\hat{\mathbf{y}}$ into proportional ratios between the probability density function of the data and its entropy in [2.5]. Relaxation of [2.4,2.3] allows the ICA solution to reach a stable point to find the resultant $\mathbf{W}$ and w in solving [2.1].

$$\hat{y_i} = \frac{\partial}{\partial y_i}\frac{\partial y_i}{\partial u_i} \quad but\ [\mathbf{y} = F_u(\boldsymbol{u})]\ so \quad = \frac{\partial f_u(u_i)}{\partial F_u(u_i)}. \tag{2.5}$$

The complications of breaking down biologic data when the underlying triggers are masked, requires one to find those triggers that are most independent and/or conversely most dependent on each other. This is especially true when resolving the ERPs of a subject across all the recorded channels. Finding the most temporal distinct patterns enables effective mappings of how the various ERPs are generated and propagate through the brain scaling as needed to the number of channels present. As ICA is a blind separator it is also able to separate contributions from non-desired sources, eye blinks and muscle artifacts, as they would be physiologically decoupled from other activity in the brain. This is why ERPs are important because they enable data related to the targeted class, in our case the ERP response, to be focused in a supervised manner. Without knowing what is already present in the data, it would be difficult to discern what components ICA returned to classify.

Caution must be advised because ICA does not provide any assurance that the found principal components are the best principal components. The decomposition finds only a suitable minimum for what best represents the presented data given the specified optimization parameter. The resultant components are not ordered by strength, in terms of contribution to the original signal, nor is any indication given as to how much overlap there is between components. Ideally, the results are all decoupled from each other, but this relies on the size and quality of the observations to cover the full space of the class. The quality of data on hand and the variation of ICA used will greatly impact the resultant mapping. This is not to say that the method won't continue to work, but the results may not always produce the ideal mapping of electrodes to stimulus response.

ICA's indifference to outside information makes it ideal in this application where external data, scalp maps or head geometry, is not available. This is in contrast to to Principal Component Analysis (PCA) that requires additional knowledge to be able to separate the data. This requirement comes from needing to scale each set of variables in an observation to properly map out the observation in a *nth* order space, where n is the number of variables in the observation. This *a priori* knowledge would be hugely beneficial in the context of this work, but the degrees of freedom are too large to make it feasible to attain. PCA returns components in order of their variance which makes the first component the strongest contributor and each successive component less impactful.

In this application, the algorithm is working as a spatial filter to discern which channels are the most distinct at capturing the ERPs. There initial recording device consists of 71 channels from which they are able to resolve 20 maximally independent components using *runica* - an automated form of ICA built out of the extended infomax ICA algorithm developed by the author.[7] These components give rise to both single pole source projections and bilateral symmetric dipole sources. The resultant sources are mapped back to the intersection locations in the brain, but suffer from variance in the placement of the probes on the subjects' heads. Various sensor stimuli indicate location maps which further support the effectives of the approach given the final mappings align with commonly seen types of EEG activity for such a stimulus response.

2.3 Results — Event-Related Brain Dynamic State Space

Compiling the results of processing all the ERPs from the study they are able to build a state-space model, [5], related to EEG frequency, EEG Power, and ITC. The states modeled in Figure 5 are event-related desynchronization (ERD), event-related synchronization (ERS), ERP, partial phase-resetting (PPR), and a grouping label "?" which represents a feature of undefined event-related data. This space exists where signals would undergo phase-locking while decreasing in power, however it lacks an official label. It also enables ERPs to be mapped to brain locations enhancing the results of studies working with repetitive sensory stimulus beyond a single probe location. The

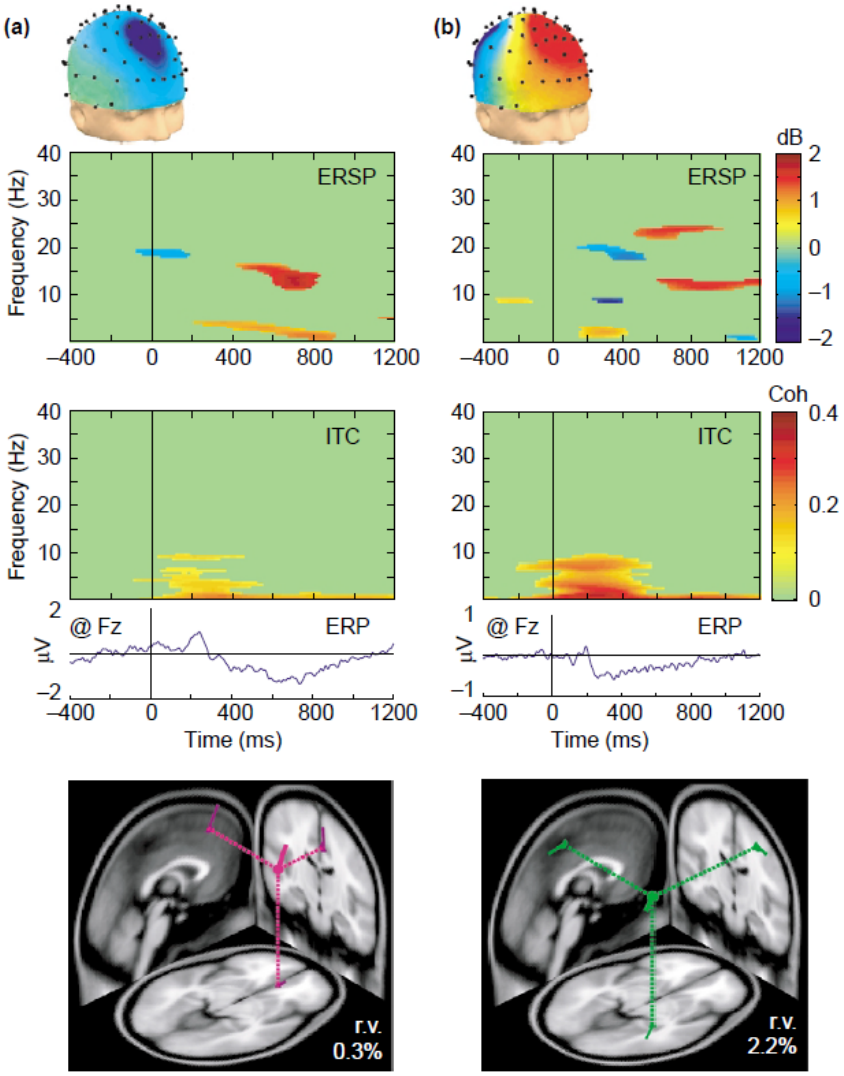

Figure 4: Combine ERP, ERSP, and ITC mapping. [4]

tightening of the labeling standards can lead to new ways to separate or combine already existing data results. The largest contribution can be seen from taking what was assumed to be a 1D ERP image and turning it into a 2D image that can be used to map a location in 3D dimensions when compared across channels.

2.4 Discussion — A Biological Feature Basis

Even with the relative ease that ERPs can be tracked in EEGs, difficulty remains in correctly determining their source as the ICA results can only map back to the electrodes

location. Maintaining uniform electrode placement across subjects and mitigating errors from skull structures handicaps the true accuracy of the source detection. As the brain heat maps of Figure 4 show, the location is generalized on the scalp and then pinpointed with assumed electrode placement. This provides a more complete contextual answer to the question of 'how does the brain response to triggered stimulus' and what label is that response given. The dynamic state space provides a 4-dimensional vector resolved directly from linearly independent components providing classification that is robust in regards to subject-specific variability and subject variability.

Given the success of the author in finding upwards of 20 unique components, the approach can be deployed on presently existing recordings of similar tests to unearth their own results. As this work was presented as an opinion piece, it seems the authors did not fully work through how the physically mapped location correlates with the type of event-related dynamic experienced. They only present the method on a few test cases from their data set, but even in that set of data they note that sorting by different parameters can produce different ERP image representations. This speaks to the complexity of EEGs in that even when the data is consistent, there is still concern that not enough variables have been taking into account. Chiefly among them would be the number of electrodes used in the data set, as 69 seems to be above average for many ERP studies.

Even after noting there are limitations, the author posits that the combined use of time and frequency to model evoked and induced event-related EEG signals enable improved

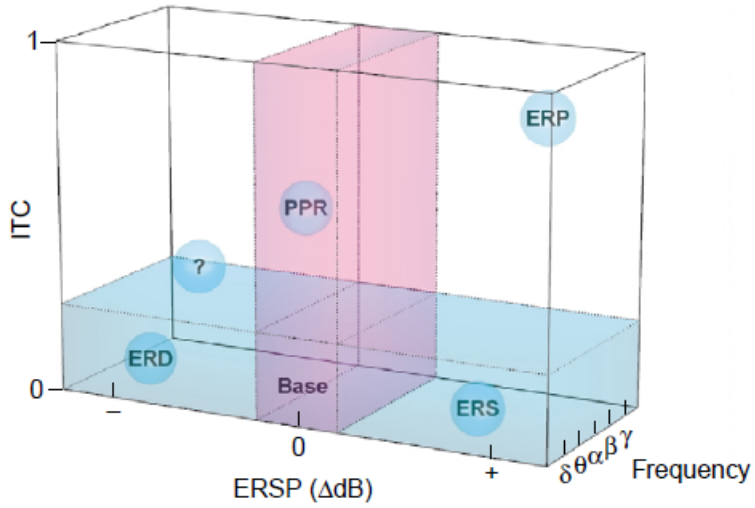

Figure 5: State space mapping results. [4]

The dimensions are the five major frequency bands [alpha, beta, delta, gamma, theta], the level ratio of ITC, and the strength of ERSP.

understanding of the brain's pathophysiology in a non-invasive manner. The precision of the surmised locations is not assured, but the methods provided locations that matched with previously known areas of interest. If specific regions are intended to react to specific stimulus then correlations between the ITC, ERSP, and frequency results may lead to a more substantial model of event related dynamics in time. However, not enough data is presented to evaluate the merit of this claim, but it does appear plausible given the direction of the work.

3 Feature Selection Based on Mutual Information: Criteria of Max-Dependency, Max-Relevance, and Min-Redundancy

Feature selection plays a critical role in accurate modeling the observed biological states which serves to minimize classification error within and across data sets. In the previous work the results matched across multiple subjects' EEG waveform activity, suggesting that their novel approach was able to replicate results from different users using the same methodology. This is one way to confirm that present feature set is applicable to the task at hand, however this method of verification is not always available.

The ability to classify data correctly beyond the scope of what is presently understood requires insight into the nature of the data itself. Domain knowledge may help, as it did in the earlier paper, yet they still relied on the effectiveness of the first 20 components returned by their ICA algorithm. The algorithm assures that the returned features will be good, but does not guarantee they are the best 20 features. In order to truly find the best features in a given set, the features that *minimize the classification error* of a given class c are the goal.

The problem is that to optimize against that parameter, minimum classification error, the entire process of feature selection followed by classification must take place. Given that the observation space, R^M with M features, of data of interest can be quite large, this iterative method takes too much time to be productive. In addition, each step introduces more degrees of freedom that must be accounted for when determining how to optimize the features which increases the complexity of the feature search.

To find a subspace containing m features that optimally models a given class, c, minimum classification error can be replaced with the maximal statistical dependency of the target class. This is one approach taken in this paper, but there are many approaches to searching an observations space such as the aforementioned maximal dependency, maximal relevance, or minimum redundancy. What one method does well, the others will struggle with so understanding the nature of the data and the classification goal can elevate one over the rest.

Maximal dependency exposes features that are most dependent on the data, maximal relevance highlights only features which share large amounts of mutual information with the target class, and minimum redundancy finds features that have the smallest mutual

information with other features. This makes dependency match well with models in the initial data set, but it leads to bias given that features will be dependent upon each other. If one misses, many will likely miss as well. Relevance on high mutual information with the target class has the same pitfall as the relationship between features is never analyzed. Redundancy doesn't take the target class into account so it is easily undone by noisy data that misrepresents the true behavior of the targeted class.

In Peng et al.'s work, they introduce the minimal-redundancy-maximal-relevance (mRMR) criterion that is applicable to continuous and discrete data sets show in Figure 6. The goal is to find promising features by balancing measures of relevance and redundancy in an effort to make robust, but comprehensive feature sets. This new selection method is compared against Max Relevance and Max Dependency feature selection methods to show it exceeds present classification levels. They are all passed through three classifiers (naive Bayes, support vector machine (SVM), and linear discriminate analysis) working over four sets of data (handwritten digits, arrhythmia, NCI cancer cell lines, and lymphoma tissues) to validate the results across continuous and discrete data through leave one out cross validation and ten-fold cross validation.

3.1 Feature Selection

3.1.1 Mutual Information

Mutual information is a metric to detail the reduction in uncertainty about an unknown random condition given knowledge about a related, but also random condition. This is expressed in (3.1) with continuous random probabilities of two random variables x and y. When attempting to match features of S_m, a feature set S with m features, to a given class c a high value would be best served as it indicates a large gain in knowledge on the state of c, useful for classification purposes. Conversely, if two random variables have low mutual information they are relatively independent of each other. This linear independence is used in the context of the first work to find features that are independent of their surroundings and thus more likely to be linked to a stimulus. Depending on the two random variables being compared, a given class c to a feature from a set S_m or two random features from S_m, the algorithm can attempt to remove probabilistically common elements or detect probabilistically unique components.

$$I(x;y) = \iint p(x,y) \log \frac{p(x,y)}{p(x)p(y)}\, dx\, dy. \tag{3.1}$$

By itself mutual information scores do not indicate how successful a given classification scheme will be once implemented. Rather, mutual information scores indicate a decrease in the uncertainty of one random variable given knowledge about another random variable. To understand the limits imposed by the features Fano's Inequality (3.2) can be

used to show the lower bound on the error of a given feature set. Fano's Inequality provides the ideal limit of error, p_e, given the probabilities of a system of random variables. Originally conceived to address the average information lost in a noisy communication channel due to error in categorization by the receiver. If the set of vectors to be communicate are more distinct, more independent of each other, then more noise will be needed to trigger information loss. Conversely, if the vectors are very similar, less independent, minimal noise will be able to obscure the information. Of course, achieving this bound target error rate is up to the classifier which is discussed in the third paper.

$$p_e \geq \frac{H(C,F)-1}{\log N} = \frac{H(C)-I(C;F)-1}{\log N} \tag{3.2}$$

Data set	HDR MultiFeat		Arrhythmia		NCI		Lymphoma	
Acronym	HDR		ARR		NCI		LYM	
Source	UCI [28], Duin et al [6][14][13]		UCI [28]		Ross et al [26] Scherf et al [27]		Alizadeh et al [1]	
Raw data type	Continuous							
Experimental data type	Discrete				Continuous			
Processing method	Binarize at μ		Discretize at μ±σ to be 3-state		z-score (mean value 0, standard deviation 1)			
# Variable	649		278		9703		4026	
# Sample	2000		420		60		96	
# Class	10		2		9		9	
Class	Name	# Sample	Name	# Sample	Name	# Sample	Name	# Sample
C1	0	200	Normal	237	NSCLC	9	DLBCL	46
C2	1	200	Abnormal	183	Renal	9	CLL	11
C3	2	200			Breast	8	ABB	10
C4	3	200			Melanoma	8	FL	9
C5	4	200			Colon	7	RAT	6
C6	5	200			Leukemia	6	TCL	6
C7	6	200			Ovarian	6	RBB	4
C8	7	200			CNS	5	GCB	2
C9	8	200			Prostate	2	LNT	2
C10	9	200						
Testing method	10-fold CV				LOOCV			

Figure 6: The four data sources with their testing method. [3]

Notice the discrete and continuous time data sets are tested using different methods, 10-fold CV and LOOCV.

The resultant p_e bounds the lower end of the probability of error in relation to: the N number of classes, F the feature set under test, C as the class label and H the resultant entropy. From this approach an understanding of the potential utility of the features can be surmised during feature selection. Given enough data, accurate estimates of the probability density functions of the features can be generated. Ideally the initial data captures all possible variations of the features to be classified, but in practice this is seldom the case. There are problems with large data sets in that (1) they become computational intensive and (2) they can become strongly biased towards the training set.

3.1.2 Maximal Dependency

Max-Dependency (3.4) selects a feature set S with m features that possess the highest dependency on the class c, the target of the search. Dependence (3.3) is best expressed as two random variables, X and Y, producing a conditional probability or joint probability that exceeds the individual probability of a given variable. The protocol to search for these features is to evaluate the mutual information, seen in (3.1) where x and y are two random variables, of all the features against the target class and select the feature with the highest probability density. Typically this involves computing the probabilistic density over the space of the original data and then growing until all m features have been found.

$$\begin{aligned} P(X \mid Y) > P(X). \\ P(X\&Y) > P(X). \end{aligned} \tag{3.3}$$

$$\max\, D(S,c), D = I(\{x_i, i = 1, ..., m\}; c). \tag{3.4}$$

This sequential search (3.5), after finding the first feature, seeks to find features that produce the largest increase in mutual information given the present conditions of the set S. Here the search equation for the mth feature is shown in terms of the previous $m-1$ features. The initial equation is taken from 3.1 and then expanded to account for the addition of the mth feature through x_m. Te eventual equation to evaluate the newest feature is dependent on an integral that covers each of the previous features resulting in an $m+1$ ordered integral since the class c must also be integrated.

$$\begin{aligned} I(S_m; c) &= \iint p(S_m, c) \log \frac{r(S_m, c)}{p(S_m)p(c)}\, dS_m\, dc. \\ &= \iint p(S_{m-1}, x_m, c) \log \frac{p(S_{m-1}, x_m, c)}{p(S_{m-1}, x_m)p(c)}\, dS_{m-1}\, dx\, dc. \\ &= \int ... \int p(x_1, ..., x_m, c) \log \frac{p(x_1, ..., x_m, c)}{p(x_1, ..., x_m)p(c)}\, dx_1 ... dx_m dc. \end{aligned} \tag{3.5}$$

This does not take into account the raw increase in formation of each feature as the search must adapt for the features already selected, thus masking redundancy of the selected features. This is seen in the difficulty computing multivariate probabilities for $p(x_1, \ldots, x_m)$ and $p(x_1, \ldots, x_m, c)$ because they are reliant on having enough samples to find the true densities of each. If features are related to each other it will be difficult to know how accurate the resultant probability is in terms of modeling the data. The likelihoods of each case, x_n, must be found from within the data itself, but this does not guarantee an accurate model of the data or the class. If all of the examples of the target c are not used to build the target, then it will be deficient in modeling the target conditions.

Of course, results can still be achieved because the model will give the best approximation of the class or the features themselves. Given the complexity of the features and classes in multivariate data limited data sets are a common problem and they often produce deficient covariance matrices as there are not enough observations to decouple all the features. A more direct solution could be achieved by computing the inverse of the covariance matrix to find mutual information, but the problem is ill-posed due to the a proper covariance matrix.

3.1.3 Maximal Relevance

$$\max D(S, c), D = \frac{1}{|S|} \sum_{x_i \in S} I(x_i; c). \tag{3.6}$$

An alternative to the Max-Dependency criterion is an approach based on maximum relevance (Max-Relevance) where each feature is scored according to its mutual information over all the data satisfying (3.6). In this manner each feature is compared against the class target c to find each mutual information value. This provides a mapping of how well each feature functions against all the others. Not taken into account are any dependencies or redundancies amongst each feature. Features possessing minimal redundancy (Min-Redundancy) can be found through computing the mutual information amongst all features and scaling it proportionality against the magnitude of the full feature set satisfying (3.7). This redundancy score can be used to mitigate the strength of Max-Dependency scores producing the minimal-redundancy-maximal-relevance (mRMR).

$$\min R(S), R = \frac{1}{|S|^2} \sum_{x_i, x_j \in S} I(x_i; x_j). \tag{3.7}$$

3.1.4 Minimal-Redundancy-Maximal-Relevance

Individually Min-Redundancy and Max-Relevance are robust feature characteristics with short-comings. Min-Redundancy seeks to find the 'rarest' of features, but makes no

claim as to how applicable they are to the data as a whole. Max-Relevance provides insight into the strongest features, but risks bloating the feature set and biasing it against the most common feature-class combinations. The criterion developed in (3.8) that combines these two schemes "minimal-redundancy-maximal-relevance" (mRMR) pits the two against each other to find balanced features.

$$\max \Phi(D, R), \Phi = D - R. \tag{3.8}$$

By taking the difference between the two, features with high relevance are penalized for their redundancy. Removing repeated features with high relevance makes the feature set more robust by diminishing any bias associated with repeated features, it helps ensure feature balance. Building a set of features that cover all conditions in the data is important to modeling the class since most data sets contain too few observations instead of too many. This means something is always left uncharacterized by the feature set and often biased against when only using of the combined criteria. Min-Redundancy does not take into account how strongly the features match the class, it measures how spread the features are amongst each other. Max-Relevance ignores repetitive features that leads to a bias of feature in capturing the range of class conditions.

$$\max_{x_j \in X - S_{m-1}} \Big[I(x_j; c) - \frac{1}{m-1} \sum_{x_i \in S_{m-1}} I(x_j; x_i) \Big]. \tag{3.9}$$

The ensuing section reviews classifiers and feature wrappers, so it is important to address how a forward-wrapper would apply to mRMR (3.9). The initial feature is chosen when $m = 1$ which can be resolved directly through (3.8). Incrementally adding features requires operating over the feature set $X - S_{m-1}$, where X is the full set and S_{m-1} are all the prior selected features. The max mutual information score returned implies which of the unselected features is to be added next where the redundancy, shown as the summation, covers only the present feature set. As such, new features are penalized for being redundant with *any and all* features already chosen. This should effectively drive the resultant mRMR to a negative value as features are chosen; effectively a natural pruning mechanism assuming all features aren't independent.

3.2 Classifiers and Wrappers

3.2.1 Naive Bayes

Naive Bayes is a child of a classifier based upon Bayes Rule, the probability of an event given conditions potentially related to said event. The probability of $\boldsymbol{Y}$ given a random variable feature vector X_n with n attributes show in equation 3.10. If there are multiple conditions, X_1, X_2, X_3, to be met for the probability they evaluate as well by making more dependencies with $P(\boldsymbol{X} = x_1, \boldsymbol{X} = x_2, \boldsymbol{X} = x_3 \mid \boldsymbol{Y} = y_i)$ in place of given

probability of $P(\boldsymbol{X} \mid \boldsymbol{Y})$. This operates assuming that the attributes may overlap and this will impact the probability of others.

$$P(\boldsymbol{Y} = y_i \mid \boldsymbol{X} = x_k) = \frac{P(\boldsymbol{X} = x_k \mid \boldsymbol{Y} = y_i)P(\boldsymbol{Y} = y_i)}{\sum_j P(\boldsymbol{X} = x_k \mid \boldsymbol{Y} = y_j)P(\boldsymbol{Y} = y_j)}. \tag{3.10}$$

Naive Bayes assumes that all of the attributes are conditionally independent to lessen the amount of data needed to generate models of the conditional probabilities. If attempts are made with only a Bayes Rule classifier, $2(2^n - 1)$ observations are required to model the probabilities accurately. Operating under the condition of conditional independence, where each attribute is proven to be independent of all others, only $2n$ observations are required to build probability models. The resultant equation ()3.11 provides a more direct method of evaluating the probability of a given class, $\boldsymbol{Y}$, given a set of features, X_n.

$$P(\boldsymbol{Y} = y_k \mid X_1 ... X_n) = \frac{P(\boldsymbol{Y} = y_k) \prod_i p(X_i \mid \boldsymbol{Y} = y_k)}{\sum_j P(\boldsymbol{Y} = y_j) \prod_i P(X_i \mid \boldsymbol{Y} = y_i)}. \tag{3.11}$$

3.2.2 Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) is a supervised learning approach that builds Gaussian models, with means μ and variances Σ, of the labeled classes to use the distances between classes in the n dimensional space as a feature discriminator. The operational equation (3.12) takes a vector $\boldsymbol{X} = x_1 ... x_n$ of data and maps it into a transformed space $\boldsymbol{Y}$. This transformed space makes it easier to linearly separate the classes contained in the data by shifting them into a one dimensional space from whatever dimension, n, the class features occupy.

$$\boldsymbol{Y} = w^T \boldsymbol{X}. \tag{3.12}$$

Fisher's LDA solution is the general basis of this approach and finds the best w transform through equation (3.13) where S_B represents the between class scatter and S_W represents the within class scatter. Scatter is equivalent to variance and defined in equation (3.14). Using the modeled class statistics, most commonly implemented as Gaussian distributions, the optimal separation matrix w^* can be determined for the transformation.

$$w^* = \text{argmax}\Big[\frac{w^T S_B w}{w^T S_W W}\Big]. \tag{3.13}$$

$$
\begin{aligned}
S_i &= \sum_{x \in w_i} (x - \mu_i)(x - \mu_i)^T. \\
S_W &= S_1 + S_2. \\
S_B &= (\mu_1 - \mu_2)(\mu_1 - \mu_2)^T.
\end{aligned}
\tag{3.14}
$$

To resolve the probabilities various distance measurements are implemented between the classes taking into account the covariance and mean of each class and often utilizing prior probabilities if known.The distance determination equation and the extension of mapping LDA into multiple classes is not the focus of the, but is presented to illustrate the foundational mechanics of the separation scheme.

3.2.3 Support Vector Machines

Support Vector Machines (SVMs) are an advanced application of LDA as they are able to optimize the location of a hyperplane in n-dimensional space between two classes. This hyper-plane enables the classifier to resolve data that is not linearly separable, but is limited to binary classifications of one versus all. The operational equation (3.15) for the support vectors follows from LDA. Instead now there is a bias term b to go with the weights w and the random variable input x. The goal is produce a result past the thresholds of 1 or -1 which corresponds to two learned classes; SVM is another supervised algorithm.

$$
f(x) = w^T x + b. \tag{3.15}
$$

If there is prior knowledge of the data, a *kernel function* can be applied to the data to aid in separating it with a hyper plane, $\Phi(x) = k(x)$. This initial transform occurs before the SVM cycles through possible hyperplanes to find the two planes, one for each set of data, that provide the best boundary between the two. The kernel function plays a large roll in allowing SVM to tackle problems LDA is unable to address, but only with prior insight into the data.

Optimization of the classifier occurs when the distance between the two hyper-planes as large as possible. This needs w minimized given distance between the two planes is represented in equation (3.16) when found between a line equation (3.15) and a point $(0, 0)$.

$$
\begin{aligned}
Distance =& \frac{|Ax_0 + By_0 + c|}{\sqrt{A^2 + B^2}}. \\
Distance =& \frac{|w^T x + b|}{||w||}. \\
Distance =& \frac{1}{||w||}.
\end{aligned}
\tag{3.16}
$$

The distance between the two planes is double, $\frac{2}{||w||}$ given the surfaces are at 1 and -1. With these two functions, equation (3.17) a quadratic constrained optimization must be solved that maps towards the optimal value of w. The norm of w is replaced with the square of itself to remove the square root function imposed by the norm function, and a convenience term is added as well.

$$
\begin{aligned}
&\text{argmin}_{w,b} \frac{1}{2}||w||^2. \\
&y_i(w^T x_i - b) \geq 1 \text{ for any } i = 1...n.
\end{aligned}
\tag{3.17}
$$

This optimization is carried out in software so the ensuing steps are not detailed as they would not provide further insight given the context of this paper. Peng et al do not specify if a kernel was used which is why there is no further analysis of kernel applications, plus it drifts from the targeted scope of finding features from within the data itself. The optimization of classifiers is not the focus of this work.

3.2.4 Wrappers: Backwards & Forwards

To define the feature set through mRMR a two-step process is used in the paper. First an incremental mRMR selection process is carried out to collect the first n features from the data via cross-validation. Each of the cross-validation sets are compared against each other in terms of the error produced during classification, called e_n. This error should remain small in mean and variance over a given range of the sets, Ω, wherein the minimum error can be found. This minimum error set will be used as the assumed optimum feature set.

Once the feature sets are found, compact subsets of the sets must be built by searching the set for the best features. With the mRMR approach this carried out naturally by the process of incrementally adding features to the original set. Other methods, like Max-Dependency require a wrapper (feature selector with a direct goal of minimizing the classification error, naive Bayes classifier) to prune the feature set. The fact that mRMR does not directly optimize the classification error makes it a 'filter', like Max-Relevance. Filters provide the benefit of being less costly to implement than a wrapper for finding the features.

The wrappers considered in the paper focus on forward selection and backward selection. Recall that the mRMR approach is a forward selection that evaluates each additional feature as it added into the larger set. Forward approaches attempt to build up a feature set that incrementally adds new features, S_{n+1}, as long as the new features produces equivalent or better classification than the current set, Sn. In this manner additional features are added to make the set cover the largest search space as possible. In opposing fashion, backward selection attempts to prune redundant features where the new classification error is no worse than the current value. These methods both work well, but tend to produce different sets of features. Ideally these should provide one constant path to the critical feature set, but that is not always the case.

One way to address which of the resultant feature spaces, through the wrapper selection method, is better is finding the set that proves to be recursively more characteristic (RM-characteristic). A RM-characteristic set will perform better with a wrapper because the range over which the classification error, e_k, is controlled is larger than the competing set. This lends to a more stabilized feature set, but this is hard to achieve given it is not tolerant of overcoming local minimum. It is entirely possible a combination of two or more additional features could improve the resultant error rate, but with the requirement that $e_{k+1} < e_k$ if may be hard to add a slightly worse feature to resolve to the stronger second feature. The paper suggests allowing a tolerance on the score improvement for each additional feature. If each additional feature keeps $e_{k+1} < e_k$, then $\rho = 0$, but if 10% fail the check $\rho = 0.9$.

3.3 Results — The Strength of the mRMR Metric

With four distinct data sets and three classifiers being fed three unique sets of features the test platform is robust and illustrates the strengths and weaknesses of each feature set. The error rate, misclassification of data, is the optimization target and is what all parameters are evaluated against. As noted in Figure 6, the parameters of each data set vary in terms of classes present, samples present, and variables per observation. In every metric:

- time to select features seen in Figure 7,
- error rate with minimal features seen in Figure 8,
- error rate with large feature sets in Figure 11,
- and across all classifiers in Figure 12

the mRMR metric outperforms the other approaches tested. Regardless of the data being continuous, Figure 9, or discrete, Figure 10, mRMR appears to be the superior feature selection criteria shown in Figure 12.

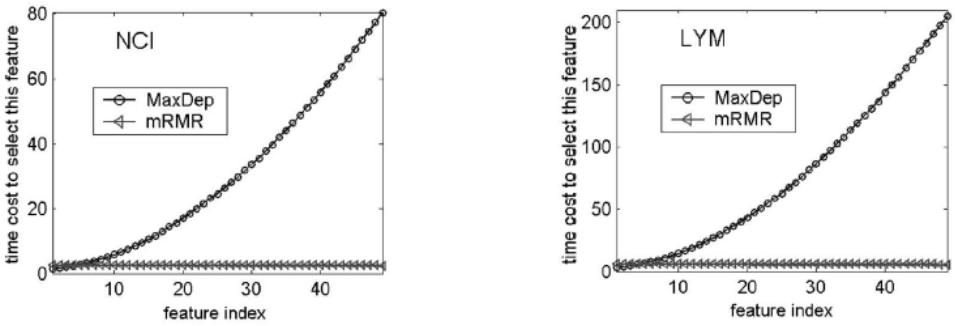

Figure 7: mRMR Classification speed on continuous data. [3]

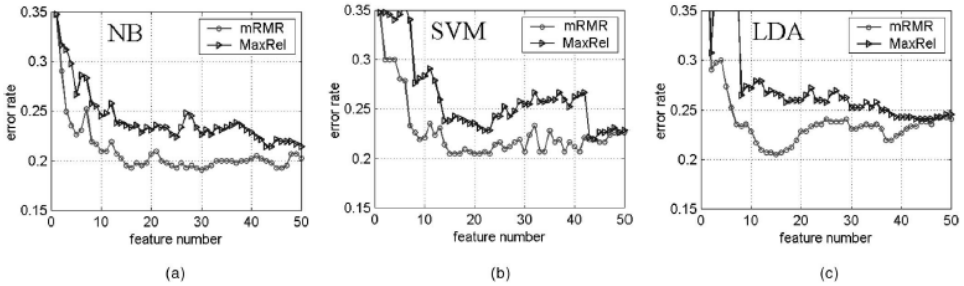

Figure 8: mRMR versus MaxRel error rates. [3]

Each algorithm is compared using (a) Naive Bayes, (b) Support Vector Machines, and (c) Linear Discriminate Analysis operating on the Arrhythmia (ARR) data.

LOOCV Error Rate (%) of NCI Data Using mRMR and MaxRel Features

Classifier	m / Method	1	5	10	15	20	25	30	35	40	45	50
NB	MaxRel	65.00	51.67	51.67	45.00	46.67	43.33	41.67	38.33	36.67	33.33	36.67
	mRMR	65.00	60.00	40.00	35.00	26.67	23.33	25.00	25.00	21.67	20.00	23.33
SVM	MaxRel	98.33	46.67	55.00	50.00	45.00	55.00	41.67	35.00	38.33	35.00	36.67
	mRMR	98.33	70.00	58.33	48.33	40.00	31.67	31.67	31.67	26.67	23.33	23.33
LDA	MaxRel	73.33	60.00	60.00	50.00	46.67	46.67	41.67	36.67	38.33	41.67	40.00
	mRMR	73.33	66.67	50.00	53.33	45.00	33.33	35.00	35.00	33.33	30.00	30.00

LOOCV Error Rate (%) of LYM Data Using mRMR and MaxRel Features

Classifier	m / Method	1	5	10	15	20	25	30	35	40	45	50
NB	MaxRel	72.92	25.00	15.63	13.54	13.54	12.50	13.54	12.50	11.46	11.46	10.42
	mRMR	72.92	17.71	16.67	10.42	11.46	9.38	10.42	9.38	9.38	7.29	8.33
SVM	MaxRel	42.71	27.08	21.88	21.88	18.75	16.67	14.58	14.58	15.63	11.46	12.50
	mRMR	42.71	11.46	10.42	7.29	5.21	7.29	7.29	5.21	5.21	5.21	4.17
LDA	MaxRel	68.75	32.29	22.92	23.96	23.96	21.88	22.92	17.71	16.67	16.67	15.63
	mRMR	68.75	18.75	15.63	12.50	6.25	7.29	5.21	3.13	4.17	3.13	3.13

Figure 9: Error rates of the continuous data. [3]

Using only Lymphoma and NCI cancer cell lines data across all classifiers.

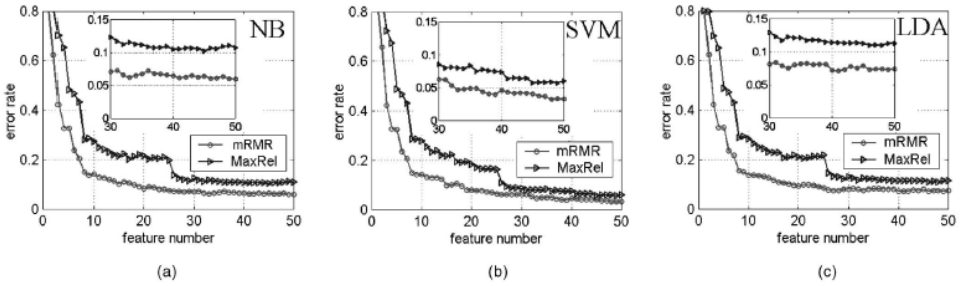

Figure 10: mRMR versus MaxRel error rates versus number of features. [3]

The classifiers tested are (a) Naive Bayes, (b) Support Vector Machines, and (c) Linear Discriminate Analysis operating on the handwritten digits (HDR) data.

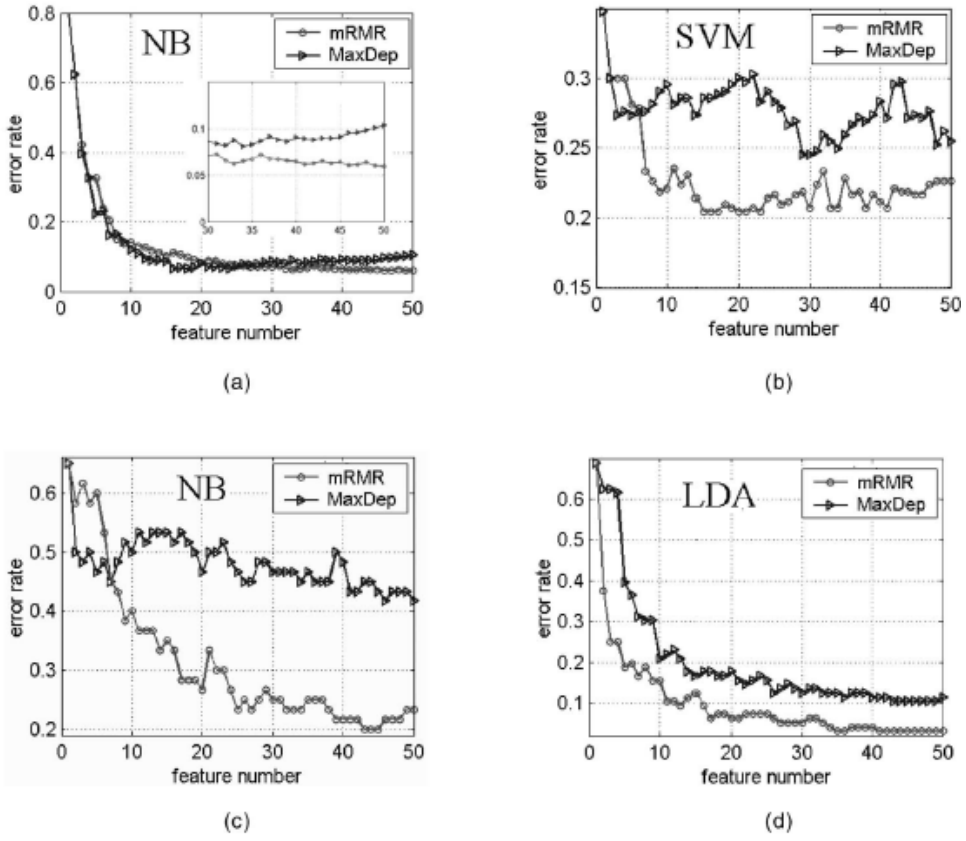

Figure 11: mRMR versus MaxRel error rates versus number of features. [3]

The classifiers tested are (a) Naive Bayes with 10-fold CV on HDR data, (b) Support Vector Machines with 10-fold CV on ARR data, and (c) Naive Bayes with LOOCV on NCI data, and (d) Linear Discriminate Analysis with LOOCV on LYM data.

The detailed results of error rates given the forward and backward wrappers fail to clearly distinguish which approach is optimal. This is not limited to mRMR, but also impacts Max-Relevance which shows a clear avenue for improvement moving forward. Even at the level of improvement seen with using mRMR the refinement of the features still plays a critical role in reducing the classification error. Incorrect implementation of mRMR is susceptible to being equal to or worse than other schemes as noted that it achieves a worse error rate than Max-Relevance when applied to the NCI data in the Forward wrapper using LDA and an equivalent error rate applied to backward wrapper with SVM on the LYM data.

3.4 Discussion — A Statistical Basis of Feature Selection

The end result, Figure 12, shows near complete dominance by the mRMR algorithm with either wrapper when compared against Max-Relevance, both of which are classified as 'filter' feature selectors. In fact the percentage of improvement of mRMR over other approaches is not consistent with any of the wrappers or classifiers. While it appears mRMR is superior to the other classifiers and feature selection methods on these four data sets, the reason why is unclear. There are many instances where additional features in the mRMR approach begin to increase the error shown in Figures 11-b, 8-b, and 8-c. It is possible there is a 'sweet spot' in terms of the number of features to be found in terms of the data sets or possibly the mRMR algorithm, but no work is done to determine the cause of these observations.

When compared with the results of the Max-Dependency feature set in Figure 11, mRMR performs slightly better across all the classifiers. Error rates of the mRMR approach are able to keep decreasing potentially because the additional features are specific to certain target classes, which Max-Relevance and Max-Dependency are unable to capture. However, not enough information is provided to contextualize how these additional features are helping the classifier. It is possible that mRMR is better at detecting and including rare features for poorly modeled features. As the authors provide no insight into the raw feature sets produced in each method, there is no way to be certain of what is driving the classification error reduction.

A possibility is that more mRMR features, really more features in general, will lead to a smaller classification error at the expense of becoming biased toward the training set. The methods for generating training and test data are ten-fold cross validation and leave out out cross validation because the original data sets are small covering samples of 2000-HDR, 420-ARR, 60-NCI, and 96-LYM. Comparing against the feature numbers show in Figure 11, NCI is given 50 features with only 60 original samples of 9 classes and LYM approaches zero error with nearly 40 features generated from only 96 samples

Data set	Wrapper	NB		SVM		LDA	
		MaxRel	mRMR	MaxRel	mRMR	MaxRel	mRMR
HDR	Forward	6.45	3.20	5.50	3.45	6.80	4.05
	Backward	5.95	3.10	4.55	2.85	6.90	4.00
ARR	Forward	18.81	17.86	20.95	19.29	19.76	18.10
	Backward	23.10	17.86	20.48	19.52	20.48	18.33
NCI	Forward	26.67	13.33	25.00	21.67	31.67	33.33
	Backward	20.00	15.00	18.33	13.33	30.00	25.00
LYM	Forward	6.25	5.21	6.25	2.08	6.25	2.08
	Backward	5.21	3.13	3.13	3.13	6.25	3.13

Figure 12: Error rates for a given data set, classifier and wrapper. [3]

of 9 classes.

With this in mind, the true improvement is how well mRMR does with as few features as possible. Evaluating the results at only 10 features, which matches the highest number of distinct classes seen in the data (the HDR has the ten hand-written digits), shows mRMR is superior in nearly ever classifier-data combination. In this capacity the algorithm is a marked improvement over the alternatives as comparing min error rate without the context of required features does not provide insight into application across data sets.

The end result is a very strong approach to aggregating features for improved classification regardless of classifier type. The method supplants the two it is based out of mathematically and in practice. Regardless of the data or the classifier, the mRMR feature sets performs the best, but there is minimal insight into predicting the performance. In many cases using the forward or backward wrapper produces unique results when all other parameters are constant. Future work in understanding any interplay between the feature sets to pinpoint the reasons for improvement would be beneficial to understanding the strength of the subtraction based mRMR in comparison to the author's suggested quotient mRMR or even to Max-Relevance and Min-Redundancy.

4 Modeling Electroencephalography Waveforms with Semi-Supervised Deep Belief Nets: Fast Classification and Anomaly Measurement

Thus far the papers have presented how features are paired with biological knowledge and how features are found and optimized. In both cases the goal is reduce the dimensionality of the data by applying a subset of knowledge about the data. The previous work touched upon the impact classifiers have and how they impact the success of a given feature set. It was seen that good features, mRMR, perform well across classifiers, but that performance is hard to track from one classifier to the next. The work in "Modeling electroencephalography waveforms with semi-supervised deep belief nets: fast classification and anomaly measurement" by Wulsin et al focuses primarily on the application of various classifiers: decision trees (DTs), support vector machines (SVMs), k-nearest neighbors (KNNs), and deep belief nets (DBNs). By feeding three sets of data to each classifier they determine how quickly the classifiers can function while validating their accuracy against the group at large.[1]

This study focuses on the need to diagnosis known classes in real time and presents a list of the clinically found objective features in Figure 13. A neurologist presented labeled data, shown in figure 13, of spike and sharp waves, generalized periodic epileptiform discharge (GPED) and triphasic waves, periodic lateralization epileptiform discharge (PLEDs), eye blink artifacts and background activity from 11 patients undergoing hypothermia treatment after cardiac arrest. Their objective is to be able to find these features again in real time as they are all clinically significant. Early detection of such

clinically significant features can help staff monitor patients in real time for anomalies, essentially any of the features noted aside from background or eye blink artifacts.

The features for the classifiers come from the annotations recordings by clustering specific hand-chosen features from Table 1 with a KNN classifier where $k = 3$. The other two feature sets are derived from the raw data and a 20 component PCA detailed further in Section 4.1. The three data sets are fed into the four different classifiers to find the processing time and accuracy for each classifier-feature combination, similar to the approach taken in Peng et al. A tool to generate a heap map of feature likelihood detection is overlaid against the known annotations to see how a real time detection system would fare against a clinician. In general the DBN classifier performs faster than its counterparts with similar accuracy, but takes considerable time to properly training before it can be deployed on data.

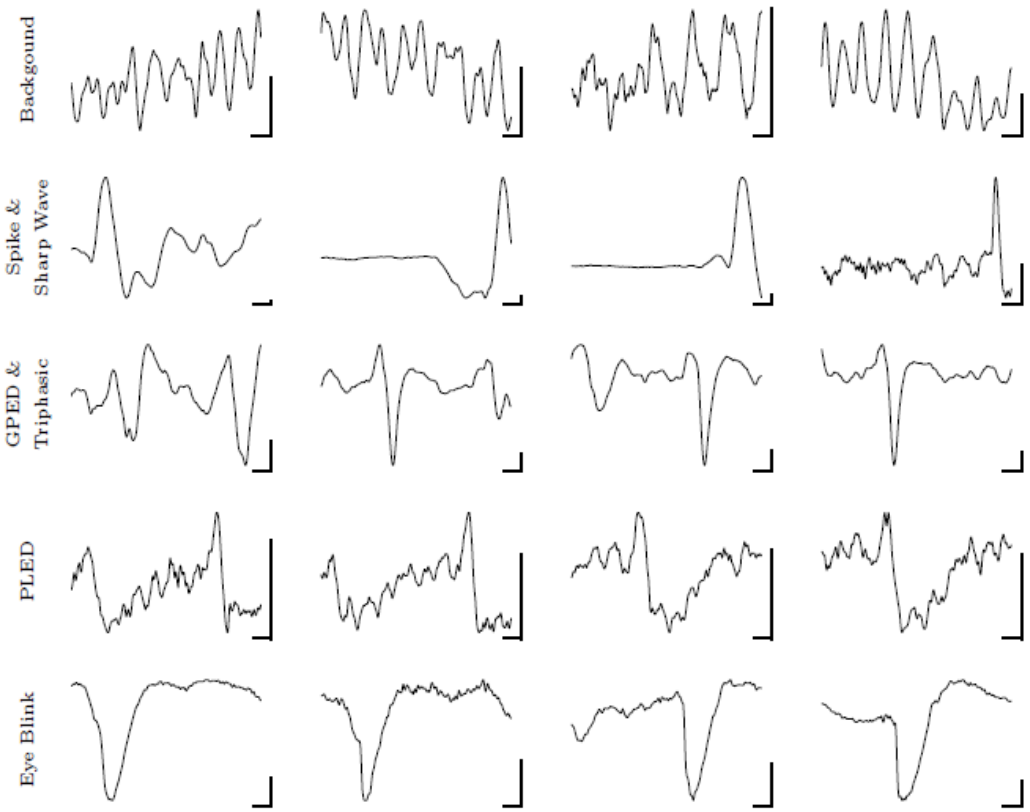

Figure 13: Neurologist-specified features of interest. [1]

Spike & Sharp Wave, GPED & Triphasic, and PLED are considered anomalous features while monitoring a patient.

Table 1: EEG Hand-Chosen Feature. [1]

Area
Normalized decay
Frequency band power
Line length
Mean energy
Average peak/valley amplitude
Normalized peak number
Peak variation
Root mean square
Wavelet energy
Zero crossings

4.1 Feature Selection

4.1.1 Domain Knowledge: Matching the Waveform

The eleven hand-chosen features in Table 1 were tested individually to ascertain independent effectiveness at classification in a KNN classifier of k = 3. Once ranked, the features were added sequentially from the strongest to the weakest to create 11 groupings of potential features. Grouping one had the strongest feature, grouping two had the two strongest features, and so on until all features were accounted for and then the classification performance of each grouping was compared using KNN with $k = 3$. The performance of these groups improved with each additional feature until the addition of the zero crossing feature. This final grouping of ten, without zero crossing, was used as the feature set in the analysis of the annotated data. The ten hand-chosen features actually represent 16 data features as the *Frequency power band* is broken into three segments, 1.5-7 Hz, 8-15 Hz and 16-29 Hz, while the *Wavelet energy* is split over four segments, 4-8 Hz, 8-16 Hz, 16-32 Hz, and 32-64 Hz. [1]

4.1.2 Statistical Knowledge: Feature Decomposition

These features, along with the raw data, and a 20 component PCA analysis are used as the three fundamental feature sets fed to the classifiers. The samples are split further into ten sub-partitions for cross-validation during classifier training. The raw data, *raw256*, contains the 256 samples of each channel-second normalized over 0 to 1. The 20 chosen PCA components are also normalized over 0 to 1, these 20 eigenvectors account for 92.75% of the variance across all sub-partitions. The hand-chosen feature set, *feat16*, extracts 16 features from each sample and normalizes over the range 0 to 1 with the top and bottom 5% being truncated to 0 or 1.

4.2 Restricted Boltzmann Machines — An Overview

Restricted Boltzmann Machines, Figure 14, are *generative stochastic artificial neural networks* that can be trained in either a supervised or unsupervised fashion. Each node functions as a boolean representation of the underlying data with the visible layer units v_i as the inputs and the hidden layer units h_j as the comparators of the visible layer and outputs. The interconnections between the layers is a matrix of weights, $W = (w_{i,j})$, that can be used to evaluate the state of a given set of node conditions, v, h as seen in equation (4.1). They deviate from their predecessor, Boltzmann Machines, in that there are no interconnections within each layer. This forces the hidden layer to interpret connections across input nodes, which is a model very similar to how the brain itself organizes information processing.

The conceptual idea of a RBM network is that unique combinations of inputs will trigger specific outputs. Since the system is binary an initial layer of 2000 visible units, given the system in Figure 15, can represent 2^{2000} possible inputs. Since the layer cannot communicate with itself to inhibit or excite within a given layer, the ensuing layer, the *hidden layer* must piece together what the activation of the *visible layer's* state represents. The probability is favorable that the data given to the visible layer does not represent 2^{2000} unique classes so the hidden layer will find that the visible layer is a finite number of classes which causes there to be redundancy amongst the hidden node activation patterns. Placing a layer on top of the first repeats the process and if the second layer is smaller in order than the first, the data will naturally be refined into a smaller set.

To ensure the visible layer permutes the data fed to it into distinct patterns, each hidden

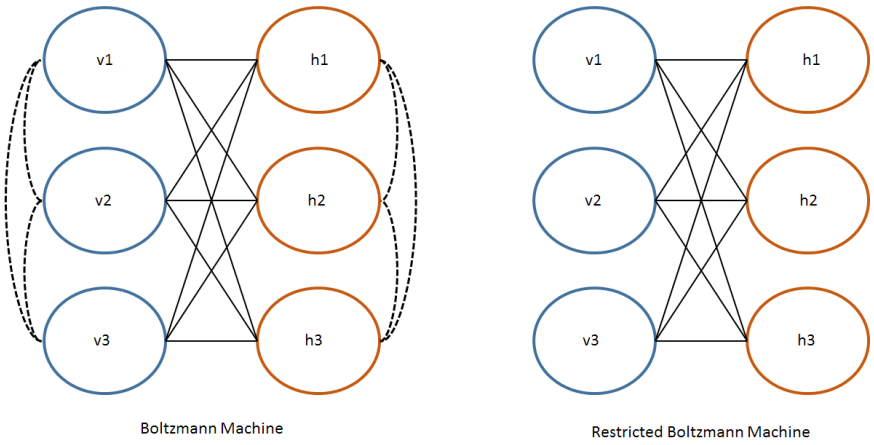

Figure 14: Layouts of Boltzmann Machine and Restricted Boltzmann Machine.

Notice the RBM lacks the intra-node connections.

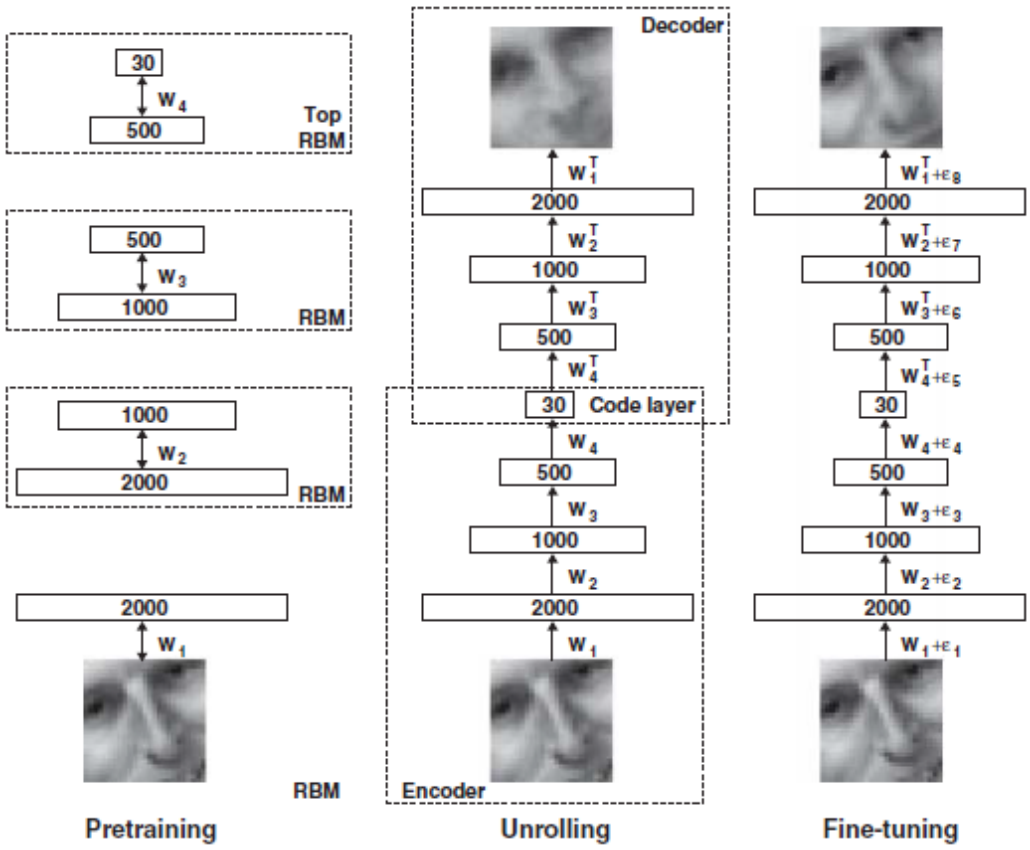

Figure 15: Functional layout and training of a DBN. [2]

This work highlights the progressive steps, and classification accuracy when apply BDNs to images.

node needs a unique bias , b_j, to ensure that some activations with each input. Initially a random bias is assigned and coupled with unique weights, $w_{i,j}$, for the links between visible and hidden layers variations in the data will provide different variations in the output of the RBM which is acting as an *autoencoder* in this capacity. In order to recover data from the hidden state back to the visible state, the visible state needs its own biases, a_i, this is important for allowing classification and reconstruction of the original data, visible in the unrolling and fine-tuning in Figure 15.

$$E(\boldsymbol{v}, \boldsymbol{h}) = -\sum_i a_i v_i - \sum_j b_j h_j - \sum_i \sum_j v_i w_{i,j} h_j. \tag{4.1}$$

To ascertain what a given hidden and visible state mean for encoding or decoding, their boolean vectors $(\boldsymbol{v}, \boldsymbol{h})$ need to be evaluated in terms of their energy configuration with

equation (4.1). The energy function is sourced from *Hopfield nets*, but the main concept is that energy of these states wants to converge to a lower level. This resembles the second law of thermodynamics, and entropy concepts, where the system works to come to rest at the lowest energy state possible which puts them at equilibrium making them stable. By computing the energy of given hidden and visible vectors the direction of change of the RBM can be understood and if it reaches a minimum then the vectors correspond to a found state. These binary nodes are represented as sigmoid functions, equation (4.2), as a summation of their bias and input vectors.

$$P(h_j = 1) = \text{Sigm}\Big(b_j + \sum_i v_i W_{ij}\Big). \tag{4.2}$$

Given an input vectors $(\boldsymbol{v}, \boldsymbol{h})$, the probabilities of all the hidden states can be determined through (4.3). Raising the energy to e ensures that resultant values are bounded given the large summation operations on the visible and hidden nodes. This helps converge on lower energy states when searching for a minimum state as sequential plots of the energy level raised to e show a substantially small slope for values less than zero, but an increasingly large slope for values greater than zero. When dealing with thousands of summations this helps prune out states are that far removed from the target state.

$$p(\boldsymbol{v}, \boldsymbol{h}) = \frac{1}{Z} e^{-E(\boldsymbol{v}, \boldsymbol{h})}. \tag{4.3}$$

The Z term corresponds to the partition function by summing over all the potential pairs of visible and hidden nodes. This weights the probability given how likely all the paths are to produce the given hidden states from the presented visible states. In a sense it is adjusting the strength of the present data set, shown in (4.4). This value should not evaluate to zero, which implies that the probability is guaranteed regardless of the present node states. As the bias values can be negative or positive for each node, and the linking matrix w values are used the function always evaluates to a positive non-zero value.

$$Z = \sum_{\boldsymbol{v}, \boldsymbol{h}} e^{-E(\boldsymbol{v}, \boldsymbol{h})}. \tag{4.4}$$

$$p(\boldsymbol{v}) = \frac{1}{Z} \sum_{\boldsymbol{h}} e^{-E(\boldsymbol{v}, \boldsymbol{h})}. \tag{4.5}$$

Given an input for each visible node, the network's probability for that input vector can be determined by summing over all the possible hidden vectors with equation (4.5). If this input vector is substantive of a targeted class, the energy of active nodes can be lowered and the energy of inactive nodes can be raised to train the layer. This will alter the partition function as well as low energy paths may be increased if they don't

agree with the present vector making it more likely that the present vector, or one very similar to it triggers an appropriate hidden layer configuration. This condition is how the system can be trained, via the log probability of the difference between the modeled data and the real data shown in equation (4.6).

$$\begin{aligned} \frac{\partial \log p(\boldsymbol{v})}{\partial w_{ij}} &= \langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{model}. \\ \Delta w_{ij} &= \epsilon(\langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{model}). \end{aligned} \tag{4.6}$$

The error between the model and the data is used to trigger updates to the weights linking the layers, w. Every component of w is updated, even those not active during the process of modeling the given input $\boldsymbol{v}$. This prunes connections between visible and hidden layers and forces them to find mappings as other connections strengthen due to repeated use, just like neurons in the brain. Implementation of this algorithm is automated, but control over the *learning rate* ϵ is critical to ensure the process converges. When learning on the weights, it is necessary to compare a histogram of the weights as they are updated to ensure the learning rate is roughly 10^-3 of the mean weight. As weight should be initialized from a zero-mean Gaussian distribution, ϵ is most likely to be on the order of hundredths.

Equation (4.6) is also applied to update the bias values when the partial derivative is taken with respect to a or b. As b is related to classification of the data it should be trained first and a being associated with reconstruction relies on the results of b if training as specified in Figure 15. Their learning rates can be larger than for the weights, but there is not a rule of thumb since the system is already operating close to ideal with assumed proper weights.

Pulling the process together, the given input data $\boldsymbol{v}$ present with an associated output $\boldsymbol{h}$ from which an overall probability can be determine through (4.3). The resultant derivative of the log probability of $p(\boldsymbol{v})$ returns the optimization parameter equation (4.6). For the data present, the vectors need only be held high to a value of 1 for all indexes to obtain an unbiased response in (4.7). The probabilities are computed using the logistic sigmoid function, $\sigma(x) = 1/(1 + e^{-x})$ to provide a binary result for the conditional state of each node.

$$\begin{aligned} p(h_j = 1 \mid \boldsymbol{v}) &= \sigma(b_j + \sum_i v_i w_i j). \\ p(v_i = 1 \mid \boldsymbol{h}) &= \sigma(a_j + \sum_j h_j w_i j). \end{aligned} \tag{4.7}$$

If the partial is taken with respect to a or b instead, then the update rate to the biases can be computed and implemented in the next iteration of training. The factor ϵ represents a learning rate the user can control prior to the algorithm's activation. The trouble arises when it comes to finding unbiased samples to be used for the realization of the

model. The original solution required *Gibbs Sampling* to permute a selection of the given visible layer and use that to seed a process by which a new hidden layer is generated. This process requires each entry in vector $\boldsymbol{v}$ to be selected iteratively until all have been updated in accordance with their probability.

To mitigate this time sink, the use of a *training vector* can be used for the visible layer from which the states of the hidden units can be generated. This *reconstruction* hidden layer, from the test vector, is used to build the corresponding *reconstruction* of the visible layer by use of (4.6) when v_i is set to 1. Now the *reconstruction* estimation takes the place of the model in (4.6). It is strongly advised to incorporate some iterations of the Gibbs Sampling to develop *Contrastive Divergence* where the the resultant reconstruction would be generated through n iterations labeled as such via CD_n.

Gibbs Sampling, equation (4.8), works to develop a unique state from randomly iterating through the system to obtain a deserved sequence of observations that are unbiased. As the input vector $\boldsymbol{v}$ is given, it cannot be used for the model to calculate the training update in equation (4.6). Instead a new vector needs to be found that also satisfies some part of a randomly selected subset of the found hidden vector $\boldsymbol{h}$. It start by picking a random state of the visible units and then calculating the resultant hidden units activity. This search continues until a hidden vector is created that could be made given the probabilities of the known hidden state vector and thus is created from an unbiased visible state vector.

$$\frac{\partial \log P((v))}{\partial W_{ij}} \approx \langle v_i h_j \rangle^0 - \langle v_i h_j \rangle^k. \tag{4.8}$$

4.3 Classifiers

Each algorithm is run under various conditions to ensure a fair showing of capabilities and faults. The results are taken from best case operation scenarios where the lesser configuration results are not reported. The optimal configuration is found via the ten-fold cross-validation which leads to the best $F1$ value (4.9). Where $F1$ combines the sensitivity and precision together represent the harmonic mean giving a rating between the two values when they differ that is tied to the weaker of the two scores. If the scores are the same value, the $F1$ will be identical, but as the values drift farther apart the distance between them drives the score to the lower value.

$$F_1 = 2\frac{\text{sensitivity} \cdot \text{precision}}{\text{sensitivity} + \text{precision}}. \tag{4.9}$$

Of primary interest is the approach to training and developing the DBN layers and weights. The search parameter in the network is the number of nodes per layer, but four layers is chosen given the available computational resources despite the parent work being based on only three layers. It is stated that additional layers improve the accuracy

of the algorithm, which is the reason for the increase in layers proven in Le Roux et al. [9]

The network is developed by training on the plethora of unlabeled data for each layer of the Restricted Boltzmann Machines (RBMs), labeled pretrainning in figure 15. With each layer being developed incrementally, the output nodes (hidden) become the visible nodes to the next tier of RBMs, but the first layer does take in the true visible data. Once all the layers are trained they are 'rolled out' or linked together and allowed to further update their weights through any desired scheme, usually forward-propagation or back-propagation. This enables the data to tune itself without labels since the initial pretrainning layers can be reversed to reconstruct the input from the classification.

4.3.1 Decision Trees

DTs create a hierarchical tree structure that is optimized with the Gini diversity index at each branch. Each attribute of the given data is processed to determine the strength of the Gini diversity index, which evaluates how likely it is for two items from the set R will be of the same class with replacement. This requires knowledge about the probability p of each item in the set. In equation 4.10, a lower value indicates high diversity so that feature should be higher up in the tree as the top of the tree will be the most traversed, those nodes need to be the easiest to evaluate.

$$I_G = 1 - \sum_{i=1}^{R} p_i^2. \tag{4.10}$$

The pruning criterion used in Wulsin's work was limiting the minimum number of samples in a node prior to splitting. The tree's binary structure allows any point to be the starting node once the Gini index is used to map the tree. A high, or low, diversity index does not make the best decision surface it it is not splitting the data into equally large sets. Ideally the splits would be 50/50, but all that can be specified are the minimum sizes for a split which were set at [2,4,6,8,10,12,14,16] during the search.

4.3.2 Support Vector Machines

Previously discussed in section 3.2.3.

4.3.3 K-Nearest Neighbors

K-Nearest Neighbors (KNNs) are a nonparametric approach to classification. This technique is supervised because it assigns each class a mean across all the dimensions of the features. This mean is used to determine where new points will be classified by

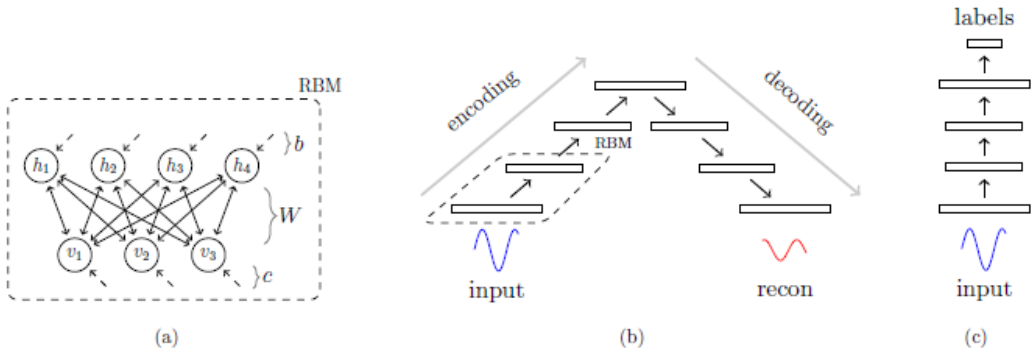

Figure 16: Configuration of DBN during training process. [1]

DBN layouts: (a) RBM model with visible layers v_i, hidden layers h_i, visible biases c, hidden biases b, and symmetric weights W, (b) Stacked RBM layers to implement a feed-forward network that breaks down the data via encoding and then reconstructs the input via decoding, and (c) an alternative representation of classification once the network has been sufficiently trained to produce classifications from the final layer.

comparing all new points to their k nearest neighbors which vote on the new points label. Distance between class centers and new points can be calculated by a variety of measures, most common for continuous variables is Euclidean distance and discrete variables use Hamming distance.

Prior knowledge can be brought into play by choosing an appropriate distance metric. As Euclidean distance only relies on the mean values of the classes it is best if data is sparse. However, with sufficient data a metric like Mahalanobis distance can be used if there are enough data points to make a full covariance matrix. Similar to SVMs kernels, KNNs performance can be manipulated given proper domain knowledge and a health sample size.

4.3.4 Deep Belief Networks

The intent of the research is to compare the above classifiers to DBNs and determine their effectiveness for real time classification. They are a layered neural network that learns patterns in high dimensional data through progressively decreasing sized layers that lead to one 'label' layer. The learning can then be reversed from the label layer to build the resultant feature. The strength of this process comes from the Restricted Boltzmann Machines acting as the connected neural network. The layers are trained one at a time and then stacked together to fine tune the full algorithm via back-propagation. Figure 16 shows a brief overview of the individual layer structure followed by its deployment as a a full system.

Wulsin trains his network first on the unlabeled data by performing layer specific RBM

optimization. Once the layers are set, they are linked together and the weights are further adjusted via back-propagation on the same unlabeled data. With the network essentially primed, it can now be optimized with the labeled data in fine-tuning back-propagation training. This is found to produce the best results and aligns with the results of Hinton's work that DBNs work best when able to start their convergence from a reasonably correct set of weights and biases. The final step is shifting the labels to account for the DBNs sensitivity to class imbalance to improve detection of less common classes, which account for the majority of classes as background is 91.6% of the overall samples.

The $F1$ measure is used to drive the optimization comparing sensitivity to precision. Each classifier is setup to run as one-against-the-others which means these measures are initial recording specific to a single class. To find the mean of a given classifier across all classes, the results are taken over each of the ten partitions and then averaged across the given classifier. The resultant measure provides an $F1$ score for each classifier over each of the three data sets. This masks the ability to discern where the improvement in classification is happening when comparing classifiers, a similar situation from the previous paper's work.

To account for the interest of real-time processing, 100 trials of each classifier for each data set were run. The average time was used to determine the relative speed of each classifier. To determine how effective different feature types, *raw256* versus *feat16*, are at seeding the search the non-background classes were compiled together to test the DBNs ability to distinguish between anomaly and background. The RMSE between the conditionals of the two classes were used to generate a heat map for likelihood of anomaly detection. This heat map would then be shown in ten second windows where the original channel-seconds overlap by 62.5 milliseconds to give a 'live' update of anomaly detection.

4.4 Results — Real Time Detection

Conclusively DBNs are not the fastest for classification time nor are they the most accurate over the majority of datasets. However, their averaged $F1$ score, over 100 trials, and computation time make them at least as good as the KNN or SVM algorithms on average. As shown in the figures, the KNN, SVM and DBN algorithms vie for $F1$ score with each ranking first on a different data set. The downfall of the others is that their computation time varies with each data set, while the DBN produces a very consistent classification time.

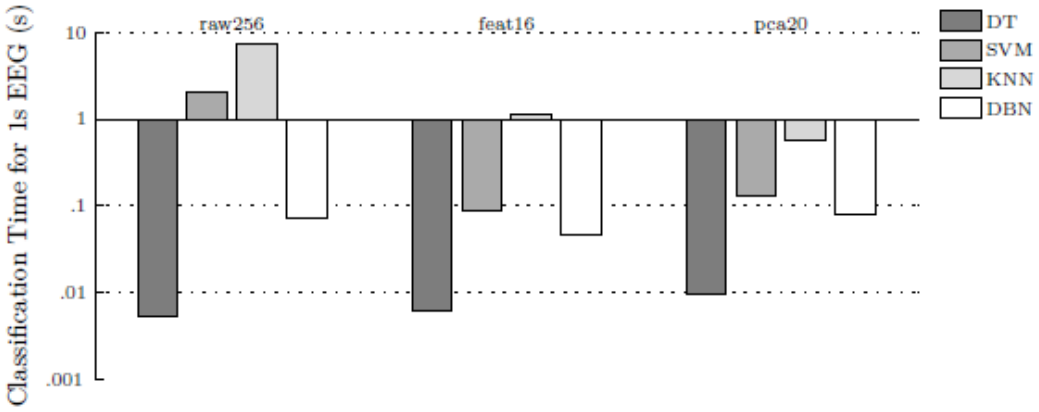

Figure 17: Mean classification times of each classifier for given data. [1]

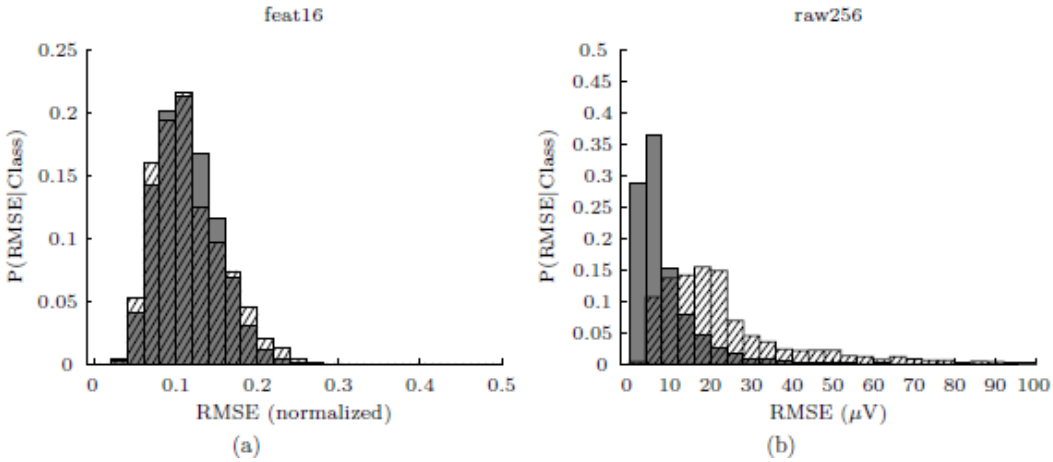

Figure 18: Histogram distributions of class-conditional probabilities. [1]

The two data sets are (a) *feat16* and (b) *raw256*. Solid backing signifies background features and hatched signifies non-background features.

The results, Figure 18, for finding the RMSE of the *feat16* and *raw256* data sets show that the null hypothesis that the differences in the two errors between the median RMSE values of the background and non-background samples is not significant ($p \ll 0.001$). It shows that the DBN returns better results when operating on the raw data as opposed to the feature based data. Plot b of Figure 18 shows the hatched, non-background classes, have a different distribution than the solid, background classes, despite them being fully overlapped in plot a. This is a critical step for vetting the functionality of the intended real time detection heat map, Figure 20, based upon the RMSE of each class.

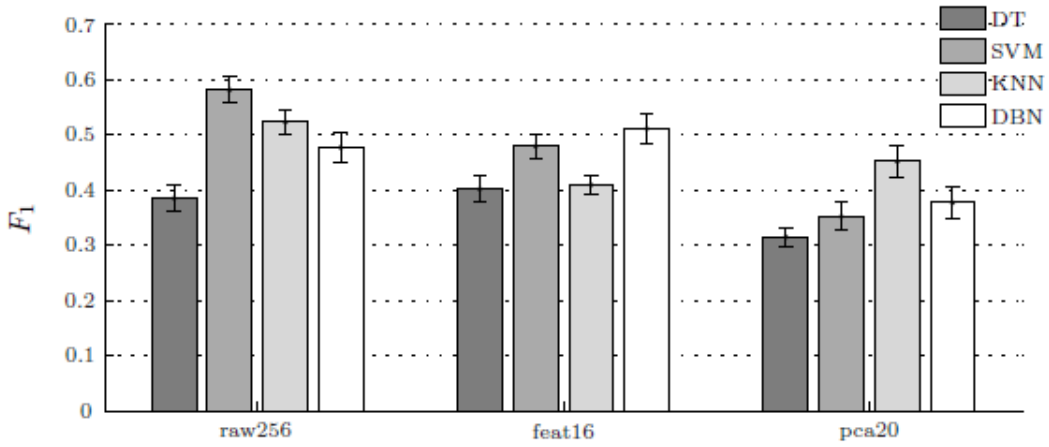

Figure 19: F_1 scores of each approach. [1]

4.5 Discussion — Classifier Effectiveness

The BND waveform classifier was noted to be a first of its kind and remains competitive with other novel approaches of in use at the time. Their work succeeds in bring competitive diagnosis into real-time across data sets. GPEDs and triphasic waveforms, along with PLEDS had not previously been incorporated into a detector prior to this work which adds to the clinical utility of the resultant classifiers. The only drawback of this work is that time it takes to fully train the algorithms to perform. The DBNs took from days to more than a week to fully develop and the KNN and SVM approaches took on the order of hours to days. Understandably if the system can be assured to function at a high level the training time is justifiable, but could be greatly reduced if tailored to one specific patient. The sample data for this study comes from a variety of subjects which increases complexity and total sample count. Adapting a present a system could also be achieved quickly assuming the new subject's median values are aligned with the weights of the present algorithm. Hinton cautioned that systems do not resolve well if they are not seeded with reasonably estimates of weights and biases.

An interesting benefit of the research highlights that all the algorithms performed within range of their optimal results with raw data when compared to feature based data. This suggests raw data may be the best option for the algorithms to operate on when performing supervised learning. While not in opposition to the work of Peng et al, it keeps the debate open about which methodology is best for developing robust feature sets. The PCA seeded algorithm actually performs the worst overall, but the authors noted they did not test all possible variants of the classification algorithms. It is likely that with finer tuning, as was done to the DBN algorithm, their results could be improved at the risk of potentially over training.

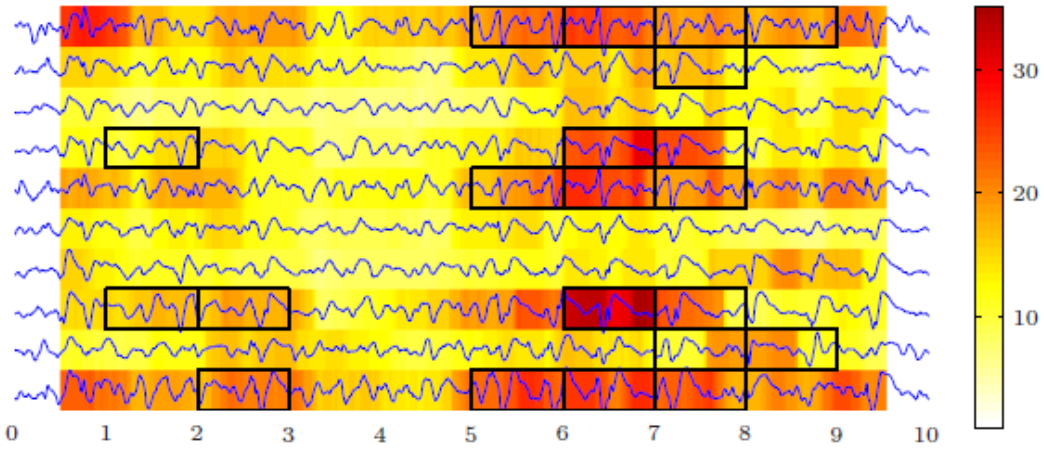

Figure 20: Classification accuracy heat map.[1]

The RMSE informs the background color, shifting to red when it is most likely to detect a feature of interest.

5 Conclusion

Makeig's work illustrated that the static state of the brain is critical in understanding responses from controlled sensory stimulus. Prior work saw the stimulus response as only ERPs and failed to link them with other similar brain state events to form a complete diagram of how a brain response to auditory and/or visual stimulus. A combination of features, ITC, ERSP and frequency, rooted in physiological knowledge enabled them to improve the classification and accuracy of stimulus response. Determining if the ERPs were the true response or a byproduct of the response by the brain to other influences sheds light on the way signals propagate throughout the brain.

It underscores the difficulty in using sensors that are mixing all the generated signals together. Critical headway was made by using ICA to spatially filter the raw electrode recordings to determine location and timing of various stimulus responses addressing the issued of mixed signals. These mappings naturally cluster over a range of patients and conditions to produce a definitive guide for what to anticipate given the state of ERSP, ITC and frequency measurements. Understanding what type of response matches a given input provides a way to establish a better model from known inputs and known outputs through a very dynamic system.

In this case, a better model of the brain could lead to understanding how other features compare and on what basis they are able to improve classification. With a proven link between inputs and outputs it should be possible to develop transforms to model the brain and use these models to compare across subjects. The other works show varying success with different feature schemes, but there is nothing specific to the chosen features

aside from improved classification accuracy. Properly developed models should be able to leverage domain knowledge to indicate how the those improved schemes' features are physiological different from others in the set. If the results cannot be mapped to a known biological process it shows glaring problems in neuroscience or a failure of the feature to properly identify the class condition.

To improve classification, methods for finding strong features were reviewed in Peng's work in the development of mRMR. Here a new features was tested against its predecessors on four data sets over a range of frequently used classifiers. The proposed mRMR feature showed statistically significant improvement over its competitors, but it did not universally win out. In fact its results were not even consistent with their being a definitive application in terms of 'with SVMs always use a forward wrapper', because the path to the lowest classification error seemed to shift with each classifier and data set. The nuance of building the feature set forward (via addition of features) or backwards (via pruning of features) came up split with error rates ranging from tightly bound (identical in some cases) to very loose (nearly 100% increase in error).

While the proposed feature creation algorithm is superior to the competition, the limited consistency of results leaves much to be understood about the approach across each presented data set. It may be that each data set requires a unique approach, but this would undermine its use if each state/person combination required extensive training to discern the best application of mRMR. This issue was outside the scope of the paper, but the work failed to show how the mRMR features differed in direct comparison to the alternatives.

This leads into the work of Wulsin where an effort is made develop a robust classifier that can succeed well in a real-time environment with extensive variability. A five class classifier is developed that successfully processes EEGs streams in real time (inside of a second) while maintaining effective detection rates comparable to commonly used approaches. This results in the best 'features' being the raw data for each of the classifiers, DTs, SVMs, KNN, and DBNs. Attempts to improve upon the raw data with a PCA or a hand-drawn feature set results in diminished results across the classifiers. Indicating that the features themselves are not the limiting factor to interpreting the data, but the chosen classifier.

Wulsin's data may be misrepresenting the strength of features given that the sample size is 11 patients and that the learning is supervised with tightly controlled labeled data. The advantages of knowing precisely what the targeted class is and allowing for large amounts of training time ranging from a few hours to over a week were not present in the other works. In truth, being able to detect EEG anomalies inside of a second is impressive, but appears impractical if it takes 24 hours of data and 5 days to train the detection algorithm. This is similar to Peng et al's resulst that showed excellent classification as the number of features began to approach the number of classes in the data. Given large amounts of time and sufficient computational power most problems become trivial, but Peng et al strove to show that mRMR always evaluated quickly and

always performed as well as if not better than its competitors. Two criteria unable to be achieved in Wulsin's work. Features may not provide the best results, but they may provide the best results when given limited resources.

The eventual realization is that all three facets of methodology for discerning the data present in an EEG can be expanded upon greatly in the future. First, the selection and cultivation of features is maturing, but at the expense of hours spent by clinicians to prime the algorithms. Then the algorithms themselves are now taking longer and longer to implement the training process due in part to more complex algorithms, but also due to the increasing complexity and amount of data on hand. Wulsin made no attempt to test the bounds of his detection, but the $F1$ scores show difficulty resolving both sensitivity and accuracy which could be a problem in the original annotations or in the algorithm. It is hard to determine which is at fault without also putting the clinician through testing.

Secondly, an alphabet is coming into focus through work based in high volume as Wulin's paper illustrates with its over 800,000 samples. The high sample count enables effective unsupervised training to calibrate his DBN to a point where it produced reasonable results. This is followed up by detailed supervised training to fine tune with labeled data and is credited with putting the effectiveness to classification performance not seen with only supervised training data. This lends credence to the notion that the original annotator could be a limiting factor if the performance with only labeled data isn't able to complete with a combined training paradigm.

This ultimately comes back to Peng in understand how the qualities of the data relate to one another. The mRMR criterion starts with features possessing high Max-Relevance scores and penalized them for being overly redundant. Ideally this penalty helps cover a wider range of cases within each class regardless of the causes of those rare events. This could potentially make mRMR able to better classify noisy signals, but the work does not discuss the quality of the data in terms of anomalies like Wulsin's work.

All of these approaches fail to develop ways to adapt if the features are rated highly, but ineffective at during classification. Linear independence and distance from a common probability density exhibited by background waveforms provide the most computational sound way to resolve information from EEGs. However unlike speech which can present with multiple distinct speakers, the brain is presenting with one speaker operating with multiple languages. Each language is important, but feature overlap between them can be clouding the resolution since numerous highly rated features must now compete in a classifier to explain phenomenon. If the classifier knew something about the class in question, as seen in Makeig's work, the classifier could actively ignore features not related to its query. Wulsin dodges this with the SVM classifier by making five versions of it, one for each feature, but the DBN was able to sort all the features through one classifier.

The ability to match signals lacking contextual information is the eventual goal of a search-by-signal database, but first searching with context must be achieved. It is clear

that specific aspects of context are understood, but uniting that understanding with feature selection and classifier operation is still developing. All of the works presented speak towards these issues highlighting developments that provide insight in the underlying scientific processes, both biological and mathematical, that make it possible to glean insight into the realm of EEG recognition.

6 Acknowledgments

I would like to thank my advisor, Dr Iyad Obeid, for his help in the preparation of this paper. As well as gratitude to the exam panel volunteers of Dr Joseph Picone, Dr Yimin Zhang, and Dr Ying Zhu. And to Dr Alessandro Napoli for providing perspective on the exam process.

References

[1] D.F. Wulsin, *et al.* "Modeling electroencephalography waveforms with semi-supervised deep belief nets: fast classification and anomaly measurement," *Journal of Neural Engineering* **8-3** (2011).

[2] G.E. Hinton and Salakhutdinoc R.R. "Reducing the dimensionality of data with neural networks," *Science* **313** 504-7 (2006).

[3] Hanchuan Peng, *et al.* "Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **27-8** (2006).

[4] Scott Makeig *et al.* "Mining event-related brain dynamics," *TRENDS in Cognitive Sciences* **8-5** (2005).

[5] Scott Makeig *et al.* "Dynamic brain sources of visual evoked responses," *Science* **295** 690-4 (2002)

[6] Scott Makeig. "Auditory event-related dynamics of the EEG spectrum and effects of exposure tones," *Electroencephalogr. Clin. Neurophsiol.* **86** 283-93 (1993).

[7] Scott Makeig *et al.* "Independent component analysis of electroencephalographic data," *Advances in Neural Information Processing Systems* **8** 145 (1996).

[8] Nader Karamzadeh *et al.* "Capturing dynamic patterns of task-based functional connectivity," *J. of NeuroImage* **66** 311-7 (2013).

[9] Nicolas Le Roux and Yoshua Bengio. "Representational Power of Restricted Boltzmann Machines and Deep Belief Networks," *Neural Comput.* **20** 1631-49 (2008).