A. COVER PAGE

Project Title: Automatic discovery and processing of EEG cohorts from clinical records	
Grant Number: 5U01HG008468-03	**Project/Grant Period:** 06/01/2015 - 05/31/2019
Reporting Period: 06/01/2017 - 05/31/2019	**Requested Budget Period:** 06/01/2017 - 05/31/2019
Report Term Frequency: Annual	**Date Submitted:** 12/17/2019
Program Director/Principal Investigator Information: JOSEPH PICONE , PHD MS **Phone number:** 2152044841 **Email:** joseph.picone@temple.edu	**Recipient Organization:** TEMPLE UNIV OF THE COMMONWEALTH TEMPLE UNIVERSITY 1801 N Broad Street, 401 Conwell Hall PHILADELPHIA, PA 191226003 **DUNS:** 057123192 **EIN:** 1231365971A1 **RECIPIENT ID:**
Change of Contact PD/PI: N/A	
Administrative Official: SHERRI GIBBS Temple Administration Services Building (TASB) 1852 North 10th Street Philadelphia, PA 191226023 **Phone number:** 215-707-3106 **Email:** sherri.gibbs@temple.edu	**Signing Official:** KAREN DENISE MITCHELL TASB 1852 North 10th Street Philadelphia, PA 191226023 **Phone number:** 215-707-7547 **Email:** grantsmanagement@temple.edu
Human Subjects: No	**Vertebrate Animals:**
hESC: No	**Inventions/Patents:** No

B. ACCOMPLISHMENTS

B.1 WHAT ARE THE MAJOR GOALS OF THE PROJECT?

The specific aims have not changed since the start of the project. The percentages listed are relative to the overall goals of the multi-year project since many of the sub-tasks span multiple years.

Specific Aim 1: Automatically recognize and time-align EEG events that contribute to a diagnosis: We will develop automated techniques to discover and align the underlying EEG events that led to a diagnosis using data-driven approaches and semi-supervised learning. Five classes of events will be identified: spike and sharp wave; generalized periodic epileptiform discharges; periodic lateralized epileptiform discharges, eye blink and artifact. Everything else is considered background. This will make the data more useful to a wide range of clinical research, and support a new form of biomedical knowledge derived from BigData repositories.

YR1 Sub-Tasks: Annotation Development (50%), Iterative Training and Bootstrapping (50%)

Specific Aim 2: Automatically recognize critical concepts in the EEG reports: We will automatically recognize clinical events (e.g. “intermittent bursts of paroxysmal high amplitude activity”) and their types: EEG-specific ACTIVITY (e.g. “beta frequency activity”), EEG-specific PATTERN (e.g. “burst suppression pattern”), or CLINICAL DEPARTMENT (e.g. “coded for 30 minutes in the emergency room”). In addition we shall automatically distinguish the clinical events’ polarity (POSITIVE, NEGATIVE) and modality (e.g. CONDITIONAL, POSSIBLE). In EEG reports, mentions of clinical events also have dense spatial and temporal information associated with them that will be mined automatically. Spatial expressions (e.g. “bilateral”, “diffuse”) and their spatial roles to the clinical events shall be discovered. Similarly, temporal expressions (e.g. “every ten seconds”) and their temporal links to the clinical events shall be automatically mined. In addition, because EEG reports describe also the clinical picture of patients, we shall identify automatically several types of medical concepts in the form of medical problems (e.g. “epilepsy”), tests and treatments.

YR1: Clinical Events (80%), Medical Concepts (75%)

Specific Aim 3: Automatic patient cohort retrieval: We shall develop a patient cohort retrieval system that will identify patients having EEGs relevant to a query or similar to a given EEG. Central to the patient cohort retrieval system is a qualified medical knowledge graph, generated automatically by using a BigData solution based on MapReduce operating on the knowledge automatically extracted in aims 1 and 2. In this way, the patient cohort retrieval system will be designed to search both free-text chart notes and EEG signals. Searching both areas will enhance retrieval for those medical events or concepts recorded in only one place. In addition, a spatial and temporal characterization of the way in which events in an EEG are narrated by physicians and the validation of these across a BigData resource are important contributions to basic science.

YR1: Generation of QMKG with MapReduce (33%), Query Expansion (33%), Index Generation (95%), Learning to Rank Based on Feedback (100%)

Specific Aim 4: Evaluation and analysis of the results of the patient cohort retrieval: To evaluate the cohort identification system clinicians and medical students shall design sets of queries that model inclusion criteria that describe the kinds of patients desired for comparative studies on EEG data. In addition, the experts will select subsets of EEGs to retrieve similar EEG data automatically from the cohort identification system. Relevance judgements produced by clinical experts shall be used to qualify the degrees of relevance of the patients identified. For each query, medical experts shall examine the top-ranked cohorts for common precision errors (false positives), and the bottom five ranked common recall errors (false negatives). User validation testing will be performed using live clinical data and the feedback will enable an analysis of the errors that will be used to better rank EEG reports. This will enhance the quality of the cohort identification system. User acceptance studies shall also be conducted and information about the perceived value of the system shall be collected.

An annotated big data archive of EEGs will greatly increase accessibility for non-experts in neuroscience, bioengineering and medical informatics who would like to study EEG data and demonstrate that a much wider range of big data bioengineering applications are now tractable. The cohort retrieval system and annotated EEG signals will greatly reduce training times for medical students pursing careers in neuroscience.

YR 1: Generation of Queries (33%), Evaluation of Patient Cohort System (33%), Analysis of Results (33%), Component Evaluation (33%), Demonstration / Feedback (25%)

B.1.a Have the major goals changed since the initial competing award or previous report?

Yes

Revised goals:

The specific aims have not changed since the start of the project. The percentages listed are relative to the overall goals of the multi-year project since many of the sub-tasks span multiple years.

Specific Aim 1: Automatically recognize and time-align EEG events that contribute to a diagnosis: We will develop automated techniques to discover and align the underlying EEG events that led to a diagnosis using data-driven approaches and semi-supervised learning. Five classes of events will be identified: spike and sharp wave; generalized periodic epileptiform discharges; periodic lateralized epileptiform

discharges, eye blink and artifact. Everything else is considered background. This will make the data more useful to a wide range of clinical research, and support a new form of biomedical knowledge derived from BigData repositories.

Specific Aim 2: Automatically recognize critical concepts in the EEG reports: We will automatically recognize clinical events (e.g. “intermittent bursts of paroxysmal high amplitude activity”) and their types: EEG-specific ACTIVITY (e.g. “beta frequency activity”), EEG-specific PATTERN (e.g. “burst suppression pattern”), or CLINICAL DEPARTMENT (e.g. “coded for 30 minutes in the emergency room”). In addition we shall automatically distinguish the clinical events’ polarity (POSITIVE, NEGATIVE) and modality (e.g. CONDITIONAL, POSSIBLE). In EEG reports, mentions of clinical events also have dense spatial and temporal information associated with them that will be mined automatically. Spatial expressions (e.g. “bilateral”, “diffuse”) and their spatial roles to the clinical events shall be discovered. Similarly, temporal expressions (e.g. “every ten seconds”) and their temporal links to the clinical events shall be automatically mined. In addition, because EEG reports describe also the clinical picture of patients, we shall identify automatically several types of medical concepts in the form of medical problems (e.g. “epilepsy”), tests and treatments.

Specific Aim 3: Automatic patient cohort retrieval: We shall develop a patient cohort retrieval system that will identify patients having EEGs relevant to a query or similar to a given EEG. Central to the patient cohort retrieval system is a qualified medical knowledge graph, generated automatically by using a BigData solution based on MapReduce operating on the knowledge automatically extracted in aims 1 and 2. In this way, the patient cohort retrieval system will be designed to search both free-text chart notes and EEG signals. Searching both areas will enhance retrieval for those medical events or concepts recorded in only one place. In addition, a spatial and temporal characterization of the way in which events in an EEG are narrated by physicians and the validation of these across a BigData resource are important contributions to basic science.

Specific Aim 4: Evaluation and analysis of the results of the patient cohort retrieval: To evaluate the cohort identification system clinicians and medical students shall design sets of queries that model inclusion criteria that describe the kinds of patients desired for comparative studies on EEG data. In addition, the experts will select subsets of EEGs to retrieve similar EEG data automatically from the cohort identification system. Relevance judgements produced by clinical experts shall be used to qualify the degrees of relevance of the patients identified. For each query, medical experts shall examine the top-ranked cohorts for common precision errors (false positives), and the bottom five ranked common recall errors (false negatives). User validation testing will be performed using live clinical data and the feedback will enable an analysis of the errors that will be used to better rank EEG reports. This will enhance the quality of the cohort identification system. User acceptance studies shall also be conducted and information about the perceived value of the system shall be collected.

An annotated big data archive of EEGs will greatly increase accessibility for non-experts in neuroscience, bioengineering and medical informatics who would like to study EEG data and demonstrate that a much wider range of big data bioengineering applications are now tractable. The cohort retrieval system and annotated EEG signals will greatly reduce training times for medical students pursing careers in neuroscience.

B.2 WHAT WAS ACCOMPLISHED UNDER THESE GOALS?

File uploaded: accomplishments_v01.pdf

B.3 COMPETITIVE REVISIONS/ADMINISTRATIVE SUPPLEMENTS

For this reporting period, is there one or more Revision/Supplement associated with this award for which reporting is required?

No

B.4 WHAT OPPORTUNITIES FOR TRAINING AND PROFESSIONAL DEVELOPMENT HAS THE PROJECT PROVIDED?

File uploaded: training.pdf

B.5 HOW HAVE THE RESULTS BEEN DISSEMINATED TO COMMUNITIES OF INTEREST?

Throughout the course of this project, we have maintained an extensive web presence using two sites:

(1) Data and Resources: https://www.isip.piconepress.com/projects/tuh_eeg/
(2) Project-related Information: https://www.isip.piconepress.com/projects/nih_cohort/

The first site is most significant because has been used to disseminate our databases, annotated data, tagged reports and software. We currently have over 2,250 subscribers of these resources and we average about three customer contacts per day.

We currently distribute five major resources related to this project: (1) the TUH EEG Corpus, (2) the TUH EEG Normal/Abnormal Corpus, (3) the TUH EEG Artifact Corpus, (4) the TUH EEG Seizure Corpus, and (5) the TUH EEG Slowing Corpus. These are all subsets of the main TUH EEG Corpus. We have released manual and automatically generated annotations of the data.

We also have released various types of support software and documents that educate users about the data. We provide personalized customer support as a courtesy to our users as well.

B.6 WHAT DO YOU PLAN TO DO DURING THE NEXT REPORTING PERIOD TO ACCOMPLISH THE GOALS?

Not Applicable

Cohort retrieval systems can be a powerful clinical tool to aid diagnosis and training if we can harness the untapped potential of electronic medical records that include unstructured text, temporally constrained measurements (e.g., vital signs), multichannel signal data (e.g., EEGs), and image data (e.g., MRIs). In this talk, we will demonstrate a system that automatically ingests and organizes medical reports so that unstructured queries can be answered. Clinicians are able to retrieve relevant EEG signals and EEG reports using standard queries (e.g. "Young patients with focal cerebral dysfunction who were treated with Topamax") from a large open source repository of clinical EEG data known as the TUH EEG Corpus. The system automatically annotates EEG signal data for events such as seizures and overall assessments such as abnormality using a multi-stage deep learning approach that integrates temporal and spatial context. Key clinical concepts are extracted from unstructured EEG reports using a novel recurrent neural network approach and used to understand semantic composition. The system also infers underspecified information and normalizes information across reports. Signal data annotations are combined with clinical concepts in a single unified representation based on a Qualified Medical Knowledge Graph. A novel learning-to-rank framework was developed to improve cohort ranking and usability based on relevance judgements produced by neurologists. The annotated data and knowledge representations are available as open source resources.

In this project, there were four aims directed at the development of two major subsystems: automatic labeling of EEG signal and automatic interpretation of electronic medical records. EEG processing was the primary subject of Aim 1, which the Temple University team was responsible for. Automatic processing of electronic medical records was the primary focus of Aims 2 and 3, which the University of Texas at Dallas team was responsible. Aim 4 was directed towards the integration of these two technologies into a Cohort Retrieval system.

1. Automated Processing of EEG Records

Scalp electroencephalograms (EEGs) are the primary means by which physicians diagnose brain-related illnesses such as epilepsy and seizures. Automated seizure detection using clinical EEGs is a very difficult machine learning problem due to the low fidelity of a scalp EEG signal. Nevertheless, despite the poor signal quality, clinicians can reliably diagnose illnesses from visual inspection of the signal waveform. Commercially available automated seizure detection systems, however, suffer from unacceptably high false alarm rates. Deep learning algorithms that require large amounts of training data have not previously been effective on this task due to the lack of big data resources necessary for building such models and the complexity of the signals involved. The evolution of big data science, most notably the release of the Temple University EEG (TUEG) Corpus, has motivated renewed interest in this problem.

In this section, we discuss the application of a variety of deep learning architectures to automated seizure detection. Architectures explored include multilayer perceptrons, convolutional neural networks (CNNs), long short-term memory networks (LSTMs), gated recurrent units and residual neural networks. We use the TUEG Corpus, supplemented with data from Duke University, to evaluate the performance of these hybrid deep structures. Since TUEG contains a significant amount of unlabeled data, we also discuss unsupervised pre-training methods used prior to training these complex recurrent networks.

Exploiting spatial and temporal context is critical for accurate disambiguation of seizures from

artifacts. We explore how effectively several conventional architectures are able to model context and introduce a hybrid system that integrates CNNs and LSTMs. The primary error modalities observed by this state-of-the-art system were false alarms generated during brief delta range slowing patterns such as intermittent rhythmic delta activity. A variety of these types of events have been observed during inter-ictal and post-ictal stages. Training models on such events with diverse morphologies has the potential to significantly reduce the remaining false alarms. This is one reason we are continuing our efforts to annotate a larger portion of TUEG. Increasing the data set size significantly allows us to leverage more advanced machine learning methodologies.

1.1. Background

An EEG records the electrical activity along the scalp and measures spontaneous electrical activity of the brain. The signals measured along the scalp can be correlated with brain activity, which makes it a primary tool for diagnosis of brain-related illnesses. Electroencephalograms (EEGs) are used in a broad range of health care institutions to monitor and record electrical activity in the brain. EEGs are essential in the diagnosis of clinical conditions such as epilepsy, depth of anesthesia, coma, encephalopathy, brain death and even in the progression of Alzheimer's disease.

Manual interpretation of EEGs is time-consuming since these recordings may last hours or days. It is also an expensive process as it requires highly trained experts. Therefore, high performance automated analysis of EEGs can reduce time of diagnosis and enhance real-time applications by flagging sections of the signal that need further review. Many methods have been developed over the years, including time-frequency digital signal processing techniques, autoregressive spectral analysis, wavelet analysis, nonlinear dynamical analysis, multivariate techniques based on simulated leaky integrate-and-fire neurons and expert systems that attempt to mimic a human observer. In spite of recent research progress in this field, the transition of automated EEG analysis technology to commercial products in operational use in clinical settings has been limited, mainly because of unacceptably high false alarm rates.

In recent years, progress in machine learning and big data resources has enabled a new generation of technology that is approaching acceptable levels of performance for clinical applications. The main challenge in this task is to operate with an extremely low false alarm rate. A typical critical care unit contains 12 to 24 beds. Even a relatively low false alarm rate of 5 false alarms (FAs) per 24 hours per patient, which translates to between 60 and 120 false alarms per day, would overwhelm healthcare staff servicing these events. This is especially true when one considers the amount of other equipment that frequently trigger alerts. In this section, we discuss the application of deep learning technology to the automated EEG interpretation problem and introduce several promising architectures that deliver performance close to the requirements for operational use in clinical settings.

1.1.1. Leveraging Recent Advances in Deep Learning

Machine learning has made tremendous progress over the past three decades due to rapid advances in low-cost highly-parallel computational infrastructure, powerful machine learning algorithms, and, most importantly, big data. Although contemporary approaches for automatic interpretation of EEGs have employed more modern machine learning approaches such as neural networks [19, 20] and support vector machine, state-of-the-art machine learning algorithms have not previously been utilized in EEG analysis because of a lack of big data resources. A significant big data

resource, known as the TUH EEG Corpus (TUEG) is now available creating a unique opportunity to evaluate high performance deep learning approaches. This database includes detailed physician reports and patient medical histories, which are critical to the application of deep learning. However, transforming physicians' reports into meaningful information that can be exploited by deep learning paradigms is proving to be challenging because the mapping of reports to underlying EEG events is nontrivial.

Though modern deep learning algorithms have generated significant improvements in performance in fields such as speech and image recognition, it is far from trivial to apply these approaches to new domains, especially applications such as EEG analysis that rely on waveform interpretation. Deep learning approaches can be viewed as a broad family of neural network algorithms that use a large number of layers of nonlinear processing units to learn a mapping between inputs and outputs. These algorithms are usually trained using a combination of supervised and unsupervised learning. The best overall approach is often determined empirically and requires extensive experimentation for optimization. There is no universal theory on how to arrive at the best architecture, and the results are almost always heavily data dependent. Therefore, in this section we will present a variety of approaches and establish some well-calibrated benchmarks of performance. We explore two general classes of deep neural networks in detail.

The first class is a Convolutional Neural Network (CNN), which is a class of deep neural networks that have revolutionized fields like image and video recognition, recommender systems, image classification, medical image analysis, and natural language processing through end to end learning from raw data. An interesting characteristic of CNNs that was leveraged in these applications is their ability to learn local patterns in data by using convolutions, more precisely cross-correlation, as their key component. This property makes them a powerful candidate for modeling EEGs which are inherently multichannel signals. Each channel in an EEG possesses some spatial significance with respect to the type and locality of a seizure event. EEGs also have an extremely low signal to noise ratio and events of interest such as seizures are easily confused with signal artifacts (e.g., eye movements) or benign variants (e.g., slowing). The spatial property of the signal is an important cue for disambiguating these types of artifacts from seizures. These properties make modeling EEGs more challenging compared to more conventional applications like image recognition of static images or speech recognition using a single microphone. In this study, we adapt well-known CNN architectures to be more suitable for automatic seizure detection. Leveraging a high-performance time-synchronous system that provides accurate segmentation of the signal is also crucial to the development of these kinds of systems. Hence, we use a hidden Markov model (HMM) based approach as a non-deep learning baseline system.

Optimizing the depth of a CNN is crucial to achieving state-of-the-art performance. Best results are achieved on most tasks by exploiting very deep structures (e.g., thirteen layers are common). However, training deeper CNN structures is more difficult since they are prone to degradation in performance with respect to generalization and suffer from convergence problems. Increasing the depth of a CNN incrementally often saturates sensitivity and also results in a rapid decrease in sensitivity. Often increasing the number of layers also increases the error on the training data due to convergence issues, indicating that the degradation in performance is not created by overfitting. We address such degradations in performance by designing deeper CNNs using a deep residual learning framework (ResNet).

We also extend the CNN approach by introducing an alternate structure, a deep convolutional generative adversarial network (DCGAN) to allow unsupervised training. Generative adversarial networks (GANs) have emerged as powerful techniques for learning generative models based on game theory. Generative models use an analysis by synthesis approach to learn the essential features of data required for high performance classification using an unsupervised approach. We introduce techniques to stabilize the training of DCGAN for spatio-temporal modeling of EEGs.

The second class of network that we discuss is a Long Short-Term Memory (LSTM) network. LSTMs are a special kind of recurrent neural network (RNN) architecture that can learn long-term dependencies. This is achieved by introducing a new structure called a memory cell and by adding multiplicative gate units that learn to open and close access to the constant error flow. It has been shown that LSTMs are capable of learning to bridge minimal time lags in excess of 1,000 discrete time steps. To overcome the problem of learning long-term dependencies in modeling of EEGs, we describe a few hybrid systems composed of LSTMs that model both spatial relationships (e.g., cross-channel dependencies) and temporal dynamics (e.g., spikes). In an alternative approach for sequence learning of EEGs, we propose a structure based on gated recurrent units (GRUs). A GRU is a gating mechanism for RNNs that is similar in concept to what LSTMs attempt to accomplish. It has been shown that GRUs can outperform many other RNNs, including LSTM, in several datasets.

1.1.2. Big Data Enables Deep Learning Research

Recognizing that deep learning algorithms require large amounts of data to train complex models, especially when one attempts to process clinical data with a significant number of artifacts using specialized models, we have developed a large corpus of EEG data to support this kind of technology development. The TUEG Corpus is the largest publicly available corpus of clinical EEG recordings in the world. The most recent release, v1.1.0, includes data from 2002 – 2015 and contains over 23,000 sessions from over 13,500 patients – over 1.8 years of multichannel signal data in total. This dataset was collected at the Department of Neurology at Temple University Hospital. The data includes sessions taken from outpatient treatments, Intensive Care Units (ICU) and Epilepsy Monitoring Units (EMU), Emergency Rooms (ER) as well as several other locations within the hospital. Since TUEG consists entirely of clinical data, it contains many real-world artifacts (e.g., eye blinking, muscle artifacts, head movements). This makes it an extremely challenging task for machine learning systems and differentiates it from most research corpora currently available in this area. Each of the sessions contains at least one EDF file and one physician report. These reports are generated by a board-certified neurologist and are the official hospital record. These reports are comprised of unstructured text that describes the patient, relevant history, medications, and clinical impression. The corpus is publicly available from the Neural Engineering Data Consortium (*www.nedcdata.org*).

EEG signals in TUEG were recorded using several generations of Natus Medical Incorporated's NicoletTM EEG recording technology. The raw signals consist of multichannel recordings in which the number of channels varies between 20 and 128 channels. A 16-bit A/D converter was used to digitize the data. The sample frequency varies from 250 Hz to 1024 Hz. In our work, we resample all EEGs to a sample frequency of 250 Hz. The Natus system stores the data in a proprietary format that has been exported to EDF with the use of NicVue v5.71.4.2530. The original EEG records are split into multiple EDF files depending on how the session was annotated

by the attending technician. For our studies, we use the 19 channels associated with a standard 10/20 EEG configuration and apply a Transverse Central Parasagittal (TCP) montage.

A portion of TUEG was annotated manually for seizures. This corpus is known as the TUH EEG Seizure Detection Corpus (TUSZ). TUSZ is also the world's largest publicly available corpus of annotated data for seizure detection that is unencumbered. No data sharing or IRB agreements are needed to access the data. TUSZ contains a rich variety of seizure morphologies. Variation in onset and termination, frequency and amplitude, and locality and focality protect the corpus from a bias towards one type of seizure morphology. TUSZ, which reflects a seizure detection task, is the focus of the experiments presented in this section. For related work on six-way classification of EEG events, see.

We have also included an evaluation on a held-out data set based on the Duke University Seizure Corpus (DUSZ). The DUSZ database is collected solely from the adult ICU patients exhibiting non-convulsive seizures. These are continuous EEG (cEEG) records where most seizures are focal and slower in frequency. TUSZ in contrast contains records from a much broader range of patients and morphologies. A comparison of these two corpora is shown in Table 1. The evaluation sets are comparable in terms of the number of patients and total amount of data, but TUSZ contains many more sessions collected from each patient.

It is important to note that TUSZ was collected using several generations of Natus Incorporated EEG equipment, while DUSZ was collected at a different hospital, Duke University, using a Nihon Kohden system. Hence, using DUSZ as a held-out evaluation set is an important benchmark because it tests the robustness of the models to variations in the recording conditions. Deep learning systems are notoriously prone to overtraining, so this second data set represents important evidence that the results presented here are generalizable and reproducible on other tasks.

1.2. Temporal Modeling of Sequential Signals

The classic approach to machine learning, shown in Fig. 1, involves an iterative process that begins with the collection and annotation of data and ends with an open set, or blind, evaluation. Data is usually sorted into training (train), development test set (dev_test) and evaluation (eval). Evaluations on the dev_test data is used to guide system development. One cannot adjust system parameters based on the outcome of evaluations on the eval set but can use these results to assess overall system performance. We typically iterate on all aspects of this approach, including expansion and repartitioning of the training and dev_test data, until overall system performance is optimized.

Table 1. An overview of the corpora used to develop the technology described in this section

Description	TUSZ		DUSZ
	Train	Eval	Eval
Patients	64	50	45
Sessions	281	229	45
Files	1,028	985	45
Seizure (secs)	17,686	45,649	48,567
Non-Seizure (secs)	596,696	556,033	599,381
Total (secs)	614,382	601,682	647,948

We often leverage previous stages of technology development to seed, or initialize, models used in a new round of development. Further, there is often a need to temporally segment the data, for example automatically labeling events of interest, to support further explorations of the problem space. Therefore, it is common when exploring new applications to begin with a familiar technology. As previously mentioned, EEG signals have a strong temporal component. Hence, a likely candidate for establishing good baseline results is an HMM approach, since this algorithm is particularly strong at automatically segmenting the data and localizing events of interest.

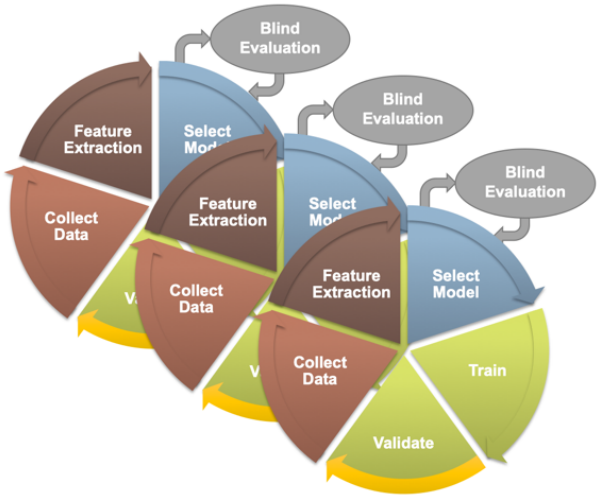

Fig. 1. An overview of a typical design cycle for machine learning

HMM systems typically operate on a sequence of vectors referred to as features. In this section, we briefly introduce the feature extraction process we have used, and we describe a baseline system that integrates hidden Markov models for sequential decoding of EEG events with deep learning for decision-making based on temporal and spatial context.

1.2.1. A Linear Frequency Cepstral Coefficient Approach to Feature Extraction

The first step in our machine learning systems consists of converting the signal to a sequence of feature vectors. Common EEG feature extraction methods include temporal, spatial and spectral analysis. A variety of methodologies have been broadly applied for extracting features from EEG signals including a wavelet transform, independent component analysis and autoregressive modeling. In this study, we use a methodology based on mel-frequency cepstral coefficients (MFCC) which have been successfully applied to many signal processing applications including speech recognition. In our systems, we use linear frequency cepstral coefficients (LFCCs) since a linear frequency scale provides some slight advantages over the mel scale for EEG signals. A block diagram summarizing the feature extraction process used in this work is presented in Fig. 2. Though it is increasingly popular to operate directly from sampled data in a deep learning system, and let the system learn the best set of features automatically, for applications in which there is limited annotated data, it is often more beneficial to begin with a specific feature extraction algorithm. Experiments with different types of features or with using sampled data directly have not shown a significant improvement in performance.

We did an extensive exploration of many of the common parameters associated with feature extraction and optimized the process for six-way event classification. We have found this approach, which leverages a popular technique in speech recognition, is remarkably robust across many types of machine learning applications. The LFCCs are computed by dividing raw EEG signals into shorter frames using a standard overlapping window approach. A high resolution Fast Fourier Transform (FFT) is computed next. The spectrum is downsampled with a filter bank composed of an array of overlapping bandpass filters. Finally, the cepstral coefficients are derived

by computing a discrete cosine transform of the filter bank's output [44]. In our experiments, we discarded the zeroth-order cepstral coefficient, and replaced it with a frequency domain energy term which is calculated by adding the output of the oversampled filter bank after they are downsampled:

$$E_f = \log\left(\sum_{k=0}^{N-1} |X(k)|^2\right). \quad (1)$$

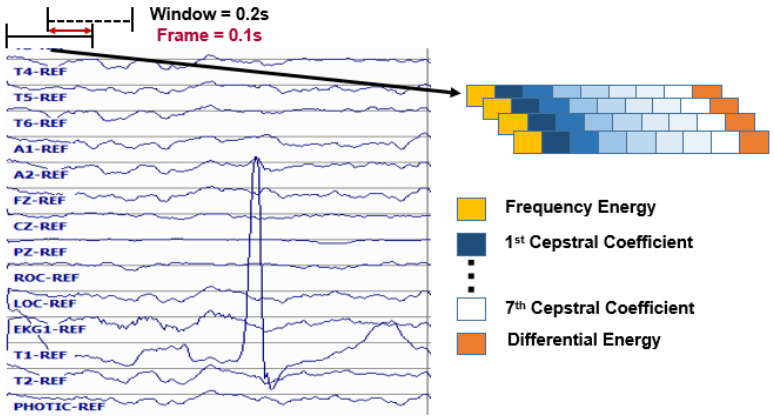

Fig. 2. Base features are calculated using linear frequency cepstral coefficients

We also introduce a new feature, called differential energy, that is based on the long-term differentiation of energy. Differential energy can significantly improve the results of spike detection, which is a critical part of seizure detection, because it amplifies the differences between transient pulse shape patterns and stationary background noise. To compute the differential energy term, we compute the energy of a set of consecutive frames, which we refer to as a window, for each channel of an EEG:

$$E_d = \max_m \left(E_f(m)\right) - \min_m \left(E_f(m)\right). \quad (2)$$

We have used a window of 9 frames which is 0.1 secs in duration, corresponding to a total duration of 0.9 secs, to calculate differential energy term. Even though this term is a relatively simple feature, it resulted in a statistically significant improvement in spike detection performance.

Our experiments have also shown that using regression-based derivatives of features, which is a popular method in speech recognition, is effective in the classification of EEG events. We use the following definition for the derivative:

$$d_t = \frac{\sum_{n=1}^{N} n(c_{t+n} - c_{t-n})}{2\sum_{n=1}^{N} n^2}. \quad (3)$$

Eq. (3) is applied to the cepstral coefficients, c_t, to compute the first derivatives, which are referred to as delta coefficients. Eq. (3) is then reapplied to the first derivatives to compute the second derivatives, which are referred to as delta-delta coefficients. Again, we use a window length of 9 frames (0.9 secs) for the first derivative and a window length of 3 (0.3 secs) for the second derivative. The introduction of derivatives helps the system discriminate between steady-state behavior, such as that found in a periodic lateralized epileptiform discharges (PLED) event, and impulsive or nonstationary signals, such as that found in spikes (SPSW) and eye movements (EYEM).

Through experiments designed to optimize feature extraction, we found best performance can be achieved using a feature vector length of 26. This vector includes nine absolute features consisting of seven cepstral coefficients, one frequency-domain energy term, and one differential energy term. Nine deltas are added for these nine absolute features. Eight delta-deltas are added because we exclude the delta-delta term for differential energy.

1.2.2. Temporal and Spatial Context Modeling

HMMs are among the most powerful statistical modeling tools available today for signals that have both a time and frequency domain component. HMMs have been used quite successfully in sequential decoding tasks like speech recognition, cough detection and gesture recognition to model signals that have sequential properties such as temporal or spatial evolution. Automated interpretation of EEGs is a problem like speech recognition since both time domain (e.g., spikes) and frequency domain information (e.g., alpha waves) are used to identify critical events. EEGs have a spatial component as well.

A left-to-right channel-independent GMM-HMM, as illustrated in Fig. 3, was used as a baseline system for sequential decoding. HMMs are attractive because training is much faster than comparable deep learning systems, and HMMs tend to work well when moderate amounts of annotated data are available. We divide each channel of an EEG into 1 sec epochs, and further subdivide these epochs into a sequence of 0.1 sec frames. Each epoch is classified using an HMM trained on the subdivided epoch. These epoch-based decisions are postprocessed by additional statistical models in a process that parallels the language modeling component of a speech recognizer. Standard three state left-to-right HMMs with 8 Gaussian mixture components per state were used. The covariance matrix for each mixture component was assumed to be a diagonal matrix – a common assumption for cepstral-based features. Though we evaluated both channel-dependent and channel-independent models, channel-independent models were ultimately used because channel-dependent models did not provide any improvement in performance.

Supervised training based on the Baum-Welch reestimation algorithm was used to train two models – seizure and background. Models were trained on segments of data containing seizures based on manual annotations. Since seizures comprise a small percentage of the overall data (3% in the training set; 8% in the evaluation set), the amount of non-seizure data was limited to be comparable to the amount of seizure data, and non-seizure data was selected to include a rich variety of artifacts such as muscle and eye movements. Twenty iterations of Baum-Welch were used though performance is not very sensitive to this value. Standard Viterbi decoding (no beam search) was used in recognition to estimate the model likelihoods for every epoch of data. The entire file was not decoded as one stream because of the imbalance between the seizure and background classes – decoding was restarted for each epoch.

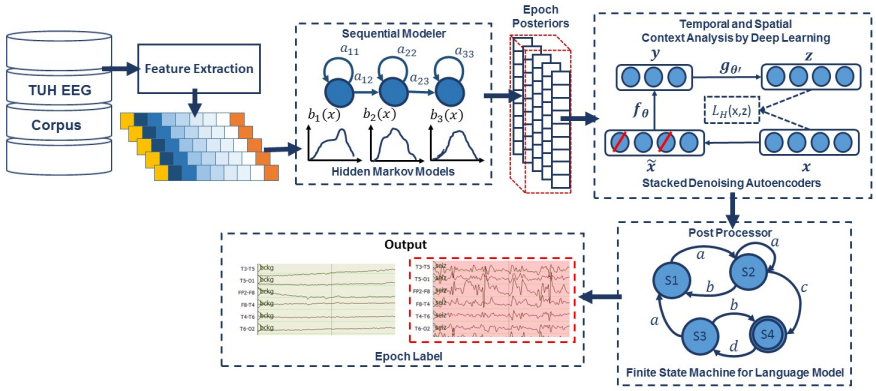

Fig. 3. A hybrid architecture based on HMMs

The output of the epoch-based decisions was postprocessed by a deep learning system. Our baseline system used a Stacked denoising Autoencoder (SdA) as shown in Fig. 3. SdAs are an extension of stacked autoencoders and are a class of deep learning algorithms well-suited to learning knowledge representations that are organized hierarchically. They also lend themselves to problems involving training data that is sparse, ambiguous or incomplete. Since inter-rater agreement is relatively low for seizure detection, it made sense to evaluate this type of algorithm as part of a baseline approach.

An N-channel EEG was transformed into N independent feature streams. The hypotheses generated by the HMMs were postprocessed using a second stage of processing that examines temporal and spatial context. We apply a third pass of postprocessing that uses a stochastic language model to smooth hypotheses involving sequences of events so that we can suppress spurious outputs. This third stage of postprocessing provides a moderate reduction in false alarms.

Training of SdA networks are done in two steps: (1) pre-training in a greedy layer-wise approach and (2) fine-tuning by adding a logistic regression layer on top of the network. The output of the first stage of processing is a vector of two likelihoods for each channel at each epoch. Therefore, if we have 22 channels and 2 classes (seizure and background), we will have a vector of dimension 2 x 22 = 44 for each epoch.

Each of these scores is independent of the spatial context (other EEG channels) or temporal context (past or future epochs). To incorporate context, we form a supervector consisting of N epochs in time using a sliding window approach. We find it beneficial to make N large – typically 41. This results in a vector of dimension 41 x 44 = 1,804 that needs to be processed each epoch. The input dimensionality is too high considering the amount of manually labeled data available for training and the computational requirements. To deal with this problem we used Principal Components Analysis (PCA) to reduce the dimensionality to 20 before applying the SdA postprocessing.

The parameters of the SdA model are optimized to minimize the average reconstruction error using a cross-entropy loss function. In the optimization process, a variant of stochastic gradient descent is used called “Minibatch stochastic gradient descent” (MSGD). MSGD works identically to stochastic gradient descent, except that we use more than one training example to make each estimate of the gradient. This technique reduces variance in the estimate of the gradient, and often makes better use of the hierarchical memory organization in modern computers.

The SdA network has three hidden layers with corruption levels of 0.3 for each layer. The number of nodes per layer are: 1st layer (connected to the input layer) = 800, 2nd layer = 500, 3rd layer (connected to the output layer) = 300. The parameters for pre-training are: learning rate = 0.5, number of epochs = 150, batch size = 300. The parameters for fine-tuning are: learning rate = 0.1, number of epochs = 300, batch size = 100. The overall result of the second stage is a probability vector of dimension two containing a likelihood that each label could have occurred in the epoch. A soft decision paradigm is used rather than a hard decision paradigm because this output is smoothed in the third stage of processing.

1.3. Improved Spatial Modeling Using CNNs

Convolutional Neural Networks (CNNs) have delivered state of the art performance on highly challenging tasks such as speech and image recognition. These early successes played a vital role in stimulating interest in deep learning approaches. In this section we explore modeling of spatial

information in the multichannel EEG signal to exploit our knowledge that seizures occur on a subset of channels. The identity of these channels also plays an important role localizing the seizure and identifying the type of seizure.

1.3.1. Deep Two-Dimensional Convolutional Neural Networks

CNN networks are usually composed of convolutional layers and subsampling layers followed by one or more fully connected layers. Consider an image of dimension W × H × N, where W and H are the width and height of the image in pixels, and N is the number of channels (e.g. in an RGB image, N = 3 since there are three colors). Two-dimensional (2D) CNNs commonly used in sequential decoding problems such as speech or image recognition typically consist of a convolutional layer that will have K filters (or kernels) of size M × N × Q where M and N are smaller than the dimension of the data and Q is smaller than the number of channels. The image can be subsampled by skipping samples as you convolve the kernel over the image. This is known as the stride, which is essentially a decimation factor. CNNs have a large learning capacity that can be controlled by varying their depth and breadth to produce K feature maps of size (W – M + 1) × (H – N + 1) for a stride of 1, and proportionally smaller for larger strides. Each map is then subsampled using a technique known as max pooling, in which a filter is applied to reduce the dimensionality of the map. An activation function, such as a rectified linear unit (ReLU), is applied to each feature map either before or after the subsampling layer to introduce nonlinear properties to the network. Nonlinear activation functions are necessary for learning complex functional mappings.

In Fig. 4, a system that combines CNN and a multi-layer perceptron (MLP) is shown. Drawing on our image classification analogy, each image is a signal where the width of the image (W) is the window length multiplied by the number of samples per second, the height of the image (H) is the number of EEG channels and the number of image channels (N) is the length of the feature vector. This architecture includes six convolutional layers, three max pooling layers and two fully-connected layers. A rectified linear unit (ReLU) nonlinearity is applied to the output of every convolutional and fully-connected layer.

In our optimized version of this architecture, a window duration of 7 secs is used. The first convolutional layer filters the input of size of 70 × 22 × 26 using 16 kernels of size 3 × 3 with a stride of 1. The input feature vectors have a dimension of 26, while there are 22 EEG channels.

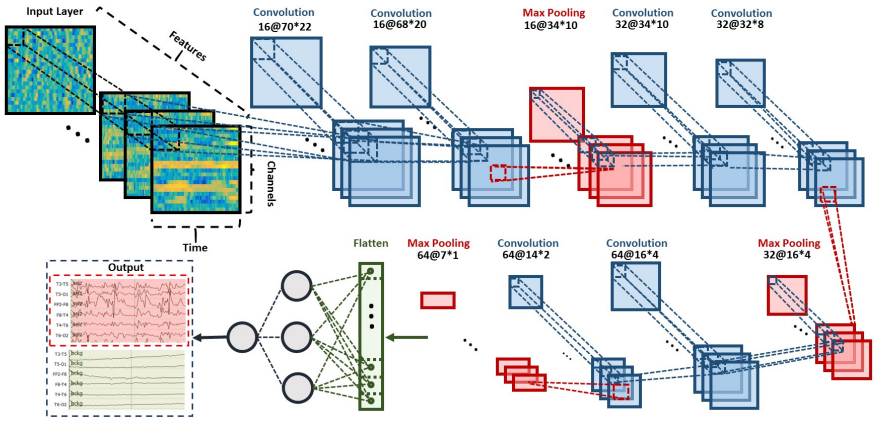

Fig. 4. A two-dimensional decoding of EEG signals using a CNN/MLP hybrid architecture

The window length is 70 because the features are computed every 0.1 secs, or 10 times per second, and the window duration is 7 sec. These kernel sizes and strides were experimentally optimized.

The second convolutional layer filters its input using 16 kernels of size 3 × 3 with a stride of 1. The first max pooling layer takes as input the output of the second convolutional layer and applies a pooling size of 2 × 2. This process is repeated two times with kernels of size 32 and 64. Next, a fully-connected layer with 512 neurons is applied and the output is fed to a 2-way sigmoid function which produces a two-class decision. This two-class decision is the final label for the given epoch, which is 1 sec in duration. Neurologists usually review EEGs using 10 sec windows, so we attempt to use a similar amount of context in this system. Pattern recognition systems often subdivide the signal into small segments during which the signal can be considered quasi-stationary. A simple set of preliminary experiments determined that a reasonable tradeoff between computational complexity and performance was to split a 10 sec window, which is what neurologists use to view the data, into 1 sec epochs.

In our experiments, we found structures that are composed of two consecutive convolutional layers before a pooling layer perform better than structures with one convolutional layer before a pooling layer. Pooling layers decrease the dimensions of the data and thereby can result in a loss of information. Using two convolutional layers before pooling mitigates the loss of information. We find that using very small fields throughout the architecture (e.g., 3 x 3) performs better than larger fields (e.g. 5 × 5 or 7 × 7) in the first convolutional layer.

1.3.2. Augmenting CNNs with Deep Residual Learning

The depth of a CNN plays an instrumental role in its ability to achieve high performanc. As many as thirteen layers are used for challenging problems such as speech and image recognition. However, training deeper CNN structures is more difficult since convergence and generalization become issues. Increasing the depth of CNNs, in our experience, tends to increase the error on evaluation dataset. As we add more convolutional layers, sensitivity first saturates and then degrades quickly. We also see an increase in the error on the training data when increasing the depth of a CNN, indicating that overfitting is actually not occurring. Such degradations in performance can be addressed by using a deep residual learning framework known as a ResNet. ResNets introduce an "identity shortcut connection" that skips layers. Denoting the desired underlying mapping as $H(x)$, we map the stacked nonlinear layers using $F(x) = H(x) - x$, where x is the input. The original mapping is recast into $F(x) + x$. It can be shown that it is easier to optimize the residual mapping than to optimize the original, unreferenced mapping.

The deep residual learning structure mitigates two important problems: vanishing/exploding gradients and saturation of accuracy when the number of layers is increased. As the gradient is backpropagated to earlier layers, repeated multiplication of numbers less than one often makes the gradient infinitively small. Performance saturates and can rapidly degrade due to numerical precision issues. Our structure addresses these problems by reformulating the layers as learning residual functions with reference to the layer inputs instead of learning unreferenced functions.

An architecture for our ResNet approach is illustrated in Fig. 5. The shortcut connections between the convolutional layers make training of the model tractable by allowing information to propagate effectively through this very deep structure. The network consists of 6 residual blocks with two 2D convolutional layers per block. These convolutional layers are followed by a fully connected

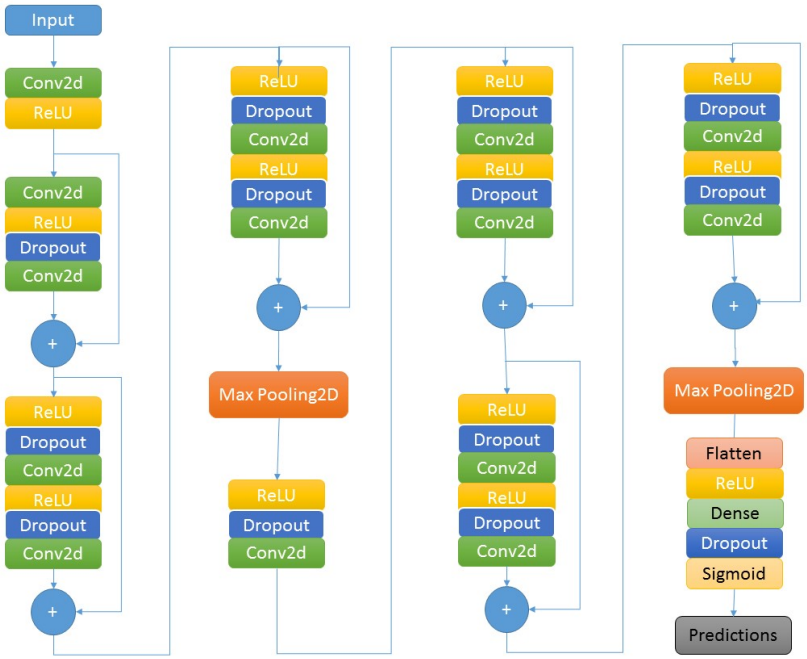

Fig. 5. A deep residual learning framework, ResNet, is shown.

layer and a single dense neuron as the last layer. This brings the total number of layers in this modified CNN structure to 14. The 2D convolutional layers all have a filter length of (3, 3). The first 7 layers of this architecture have 32 filters while the last layers have 64 filters. We increment the number of filters from 32 to 64, since the initial layers represent generic features, while the deeper layers represent more detailed features. In other words, the richness of the data representations increases because each additional layer forms new kernels using combinations of the features from the previous layer.

Except for the first and last layers of the network, before each convolutional layer we apply a Rectified Linear Unit (ReLU) as an activation function. ReLU is the most commonly used activation function in deep learning models. The function returns 0 if it receives any negative input, but for any positive value it returns that value (e.g., $f(x) = \max(0, x)$). To overcome the problem of overfitting in deep learning structures with a large number of parameters, we use dropout as our regularization method between the convolutional layers and after ReLU. Dropout is a regularization technique for addressing overfitting by randomly dropping units along with their connections from the deep learning structures during training. We use the Adam optimizer which is an algorithm for first-order gradient-based optimization of stochastic objective functions, based on adaptive estimates of lower-order moments. After parameter tuning, we apply Adam optimization using the following parameters (according to the notation in their original paper): $\alpha = 0.00005, \beta_1 = 0.9, \beta_2 = 0.999, \varepsilon = 10^{-8}$, and $decay = 0.0001$.

The deep learning systems described thus far have incorporated fully supervised training and discriminative models. Next, we introduce a generative deep learning structure based on convolutional neural networks that leverages unsupervised learning techniques. These are important for biomedical applications where large amounts of fully annotated data are difficult to find.

1.3.3. Unsupervised Learning

Machine learning algorithms can generally be split into two categories: generative and discriminative. A generative model learns the joint probability distribution of $P(X, Y)$ where X is an observable variable and Y is the target variable. These models learn the statistical distributions of the input data rather than simply classifying the data as one of C output classes. Hence the name, generative, since these methods learn to replicate the underlying statistics of the data. GMMs trained using a greedy clustering algorithm or HMMs trained using the Expectation Maximization (EM) algorithm are well-known examples of generative models. A discriminative model, on the other hand, learns the conditional probability of the target Y, given an observation X, which we denote $P(Y|X)$. Support Vector Machine and Maximum Mutual Information Estimation (MMIE) are two well-known discriminative models.

Generative adversarial networks (GANs) have emerged as a powerful learning paradigm technique for learning generative models for high-dimensional unstructured data. GANs use a game theory approach to find the Nash equilibrium between a generator and discriminator network. A basic GAN structure consists of two neural networks: a generative model G that captures the data distribution, and a discriminative model D that estimates the probability that a sample came from the training data rather than G. These two networks are trained simultaneously via an adversarial process. In this process, the generative network, G, transforms the input noise vector z to generate synthetic data $G(z)$. The training objective for G is to maximize the probability of D making a mistake about the source of the data.

The output of the generator is a synthetic EEG – data that is statistically consistent with an actual EEG but is fabricated entirely by the network. The second network, which is the discriminator, D, takes as input either the output of G or samples from real world data. The output of D is a probability distribution over possible input sources. The output of the discriminator in GAN determines if the signal is a sample from real world data or synthetic data from the generator.

The generative model, G, and the discriminative model, D, compete in a two-player minimax game with a value function, $V(G; D)$, in a way that D is trained to maximize the probability of assigning the correct label to both the synthetic and real data from G. The generative model G is trained to fool the discriminator by minimizing $log(1 - D(G(z)))$:

$$\min_G \max_D V(D, G) = \mathbb{E}_{x \sim p_{data}(x)}[log\, D(x)] + \mathbb{E}_{z \sim p_z(z)}[log(1 - D(G(z))] \,. \quad (4)$$

During the training process, our goal is to find a Nash equilibrium of a non-convex two-player game that minimizes both the generator and discriminator's cost functions.

A deep convolutional generative adversarial network (DCGAN) is shown in Fig. 6. The generative model takes 100 random inputs and maps them to a matrix with size of [21, 22, 250], where 21 is the window length (corresponding to a 21 sec duration), 22 is number of EEG channels and 250 is number of samples per sec. Recall, in our study, we resample all EEGs to a sample frequency of

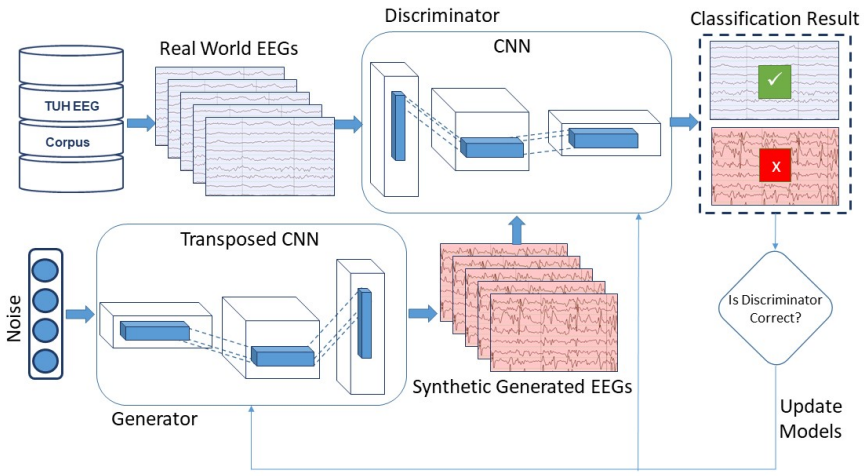

Fig. 6. An unsupervised learning architecture is shown that uses DCGANs

250 Hz. The generator is composed of transposed CNNs with upsamplers. Transposed convolution, also known as fractionally-strided convolution, can be implemented by swapping the forward and backward passes of a regular convolution. We need transposed convolutions in the generators since we want to go in the opposite direction of a normal convolution. For example, in this case we want to compose the vector of [21, 22, 250] from 100 random inputs. Using transposed convolutional layers, we can transform feature maps to a higher-dimensional space. Leaky ReLUs are used for the activation function and dropout layers are used for regularization. Adam is used as the optimizer and binary cross-entropy is used as the loss function.

In this architecture, the discriminative model accepts vectors from two sources: synthetic data generators and real data (raw EEGs in this case). It is composed of strided convolutional neural networks. Strided convolutional neural networks are like regular CNNs but with a stride greater than one. In the discriminator we replace the usual approach of convolutional layers with max pooling layers with strided convolutional neural networks. This is based on our observations that using convolutional layers with max pooling makes the training of DCGAN unstable. This is due to the fact that using strided convolutional layers, the network learns its own spatial downsampling, and convolutional layers with max pooling tend to conflict with striding.

Finding the Nash equilibrium, which is a key part of the GAN approach, is a challenging problem that impacts convergence during training. Several recent studies address the instability of GANs and suggest techniques to increase the training stability of GANs. We conducted a number of preliminary experiments and determined that these techniques were appropriate:

In the discriminator:

- pretraining of the discriminator;
- one-sided label smoothing;
- eliminating fully connected layers on top of convolutional features;
- replacing deterministic spatial pooling functions (such as max pooling) with strided convolutions.

In the generator:

- using an ReLU activation for all layers except for the output;
- normalizing the input to [-1, 1] for the discriminator;
- using a $tanh()$ activation in the last layer except for the output;

- using leaky ReLU activations in the discriminator for all layers except for the output;
- freezing the weights of discriminator during adversarial training process;
- unfreezing weights during discriminative training;
- eliminating batch normalization in all the layers of both the generator and discriminator.

The GAN approach is attractive for a number of reasons including creating an opportunity for data augmentation. Data augmentation is common in many state-of-the-art deep learning systems today, allowing the size of the training set to be increased as well as exposing the system to previously unseen patterns during training.

1.4. Learning Temporal Dependencies

The duration of events such as seizures can vary dramatically from a few seconds to minutes. Further, neurologists use significant amounts of temporal context and adaptation in manually interpreting EEGs. They are very familiar with their patients and often can identify the patient by examining the EEG signal, especially when there are certain types of anomalous behaviors. In fact, they routinely use the first minute or so of an EEG to establish baseline signal conditions, or normalize their expectations, so that they can more accurately determine anomalous behavior. Recurrent neural networks (RNN), which date back to the late 1980's, have been proposed as a way to learn such dependencies. Prior to this, successful systems were often based on approaches such as hidden Markov models, or used heuristics to convert frame-level output into longer-term hypotheses. In this section, we introduce several architectures that model long-term dependencies.

1.4.1. Integration of Incremental Principal Component Analysis with LSTMs

In the HMM/SdA structure proposed in Section 1.2.2, PCA was used prior to SdA for dimensionality reduction. Unlike HMM/SdA, applying LSTM networks directly to features requires more memory efficient approaches than PCA, or the memory requirements of the network can easily exceed the available computational resources (e.g., low-cost graphics processing units such as the Nvidia 1080ti have limited amount of memory – typically 8 Gbytes). Incremental principal components analysis (IPCA) is an effective technique for dimensionality reduction. This algorithm is often more memory efficient than PCA. IPCA has constant memory complexity proportional to the batch size, and it enables use of large datasets without a need to load the entire file or dataset into memory. IPCA builds a low-rank approximation for the input data using an amount of memory which is independent of the number of input data samples. It is still dependent on the dimensionality of the input data features but allows more direct control of memory usage by changing the batch size.

In PCA, the first k dominant principal components, $y_1(n), y_2(n), \ldots, y_k(n)$, are computed directly from the input, $x(n)$ as follows:

$For\ n\ =\ 1, 2, \ldots, do\ the\ following$:

1) $x_1(n)\ =\ x(n)$.

2) $For\ i\ =\ 1, 2, \ldots, \min(k, n), do$:

$a)\ if\ i\ =\ n, initialize\ the\ i\ principal\ component\ as\ y_i(n)\ =\ x_i(n)$;

$b)\ otherwise\ compute$:

$$y_i(n) = \frac{n-1-p}{n} y_i(n-1) + \frac{1+p}{n} x_i(n) x_i^T(n) \frac{y_i(n-1)}{||y_i(n-1)||}, \quad (5)$$

$$x_{i+1}(n) = x_i(n) - x_i^T(n) \frac{y_i(n)}{||y_i(n)||} \frac{y_i(n)}{||y_i(n)||}, \quad (6)$$

where the positive parameter p is called the amnesic parameter. Typically, p ranges from 2 to 4. Then the eigenvector and eigenvalues are given by:

$$e_i = \frac{y_i(n)}{||y_i(n)||} \; and \; \lambda_i = ||y_i(n)|| . \quad (7)$$

In Fig. 7, we present an architecture that integrates IPCA and LSTM. In this system, samples are converted to features and the features are delivered to an IPCA layer that performs spatial context analysis and dimensionality reduction. The output of the IPCA layer is delivered to a one-layer LSTM for seizure classification task. The input to the IPCA layer is a vector whose dimension is the product of the number of channels, the number of features per frame and the number of frames of context. Preliminary experiments have shown that 7 seconds of temporal context performs well. The corresponding dimension of the vector input to IPCA is 22 channels × 26 features × 7 seconds × 10 frames/second, or a total of 4004 elements. A batch size of 50 is used and the dimension of its output is 25 elements per frame at 10 frames/second. In order to learn long-term dependencies, one LSTM with a hidden layer size of 128 and batch size of 128 is used along with Adam optimization and a cross-entropy loss function.

1.4.2. End-to-End Sequence Labeling Using Deep Architectures

In machine learning, sequence labeling is defined as assigning a categorial label to each member of a sequence of observed values. In automatic seizure detection, we assign one of two labels: seizure or non-seizure. This decision is made every epoch, which is typically a 1 sec interval. The proposed structures are trained in an end-to-end fashion, requiring no pre-training and no pre-processing, beyond the feature extraction process that was explained in Section 1.2.1. For example, for an architecture composed of a combination of CNN and LSTM, we do not train CNN independently from LSTM, but we train both jointly. This is challenging because there are typically convergence issues when attempting this.

In Fig. 8, we integrate 2D CNNs, 1-D CNNs and LSTM networks, which we refer to as CNN/LSTM, to better exploit long-term dependencies. Note that the way that we handle data in CNN/LSTM is different from the CNN/MLP system presented in Fig. 4. The input EEG features vector sequence can be thought of as being composed of frames distributed in time where each frame is an image of width (W) equal to the length of a feature vector. The height (H) equals the number of EEG channels and the number of image channels (N) equals one. The input to the network consists of T frames where T is equal to the window length multiplied by the number of

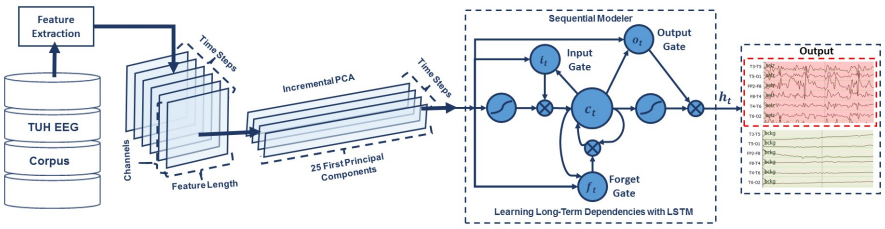

Fig. 7. An architecture that integrates IPCA and LSTM

frames per second. In our optimized system, where features are available 10 times per second, a window duration of 21 seconds is used. The first 2D convolutional layer filters 210 frames (T = 21 × 10) of EEGs distributed in time with a size of 26 × 22 × 1 (W = 26, H = 22, N = 1) using 16 kernels of size 3 × 3 with a stride of 1. The first 2D max pooling layer takes as input a vector which is 260 frames distributed in time with a size of 26 × 22 × 16 and applies a pooling size of 2 × 2. This process is repeated two times with two 2D convolutional layers with 32 and 64 kernels of size 3 × 3 respectively and two 2D max pooling layers with a pooling size 2 × 2.

The output of the third max pooling layer is flattened to 210 frames with a size of 384 × 1. Then a 1D convolutional layer filters the output of the flattening layer using 16 kernels of size 3 which decreases the dimensionality in space to 210 × 16. Next, we apply a 1D max pooling layer with a size of 8 to decrease the dimensionality to 26 × 16. This is the input to a deep bidirectional LSTM network where the dimensionality of the output space is 128 and 256. The output of the last bidirectional LSTM layer is fed to a 2-way sigmoid function which produces a final classification of an epoch. To overcome the problem of overfitting and force the system to learn more robust features, dropout and Gaussian noise layers are used between layers. To increase nonlinearity, Exponential Linear Units (ELU) are used. Adam is used in the optimization process along with a mean squared error loss function.

More recently, researchers proposed another type of recurrent neural network, known as a gated recurrent unit (GRU). A GRU architecture is similar to an LSTM but without a separate memory cell. Unlike LSTM, a GRU does not include output activation functions and peep hole connections. It also integrates the input and forget gates into an update gate to balance between the previous activation and the candidate activation. The reset gate allows it to forget the previous state. It has been shown that the performance of a GRU is on par with an LSTM, but a GRU can be trained faster. The architecture is similar to that Fig. 8, but we simply replace LSTM with GRU, in a way that the output of 1D max pooling is the input to a GRU where the dimensionality of the output space is 128 and 256. The output of the last GRU is fed to a 2-way sigmoid function which produces a final classification of an epoch. These two approaches, LSTM and GRU, are evaluated as part of a hybrid architecture that integrates CNNs with RNNs.

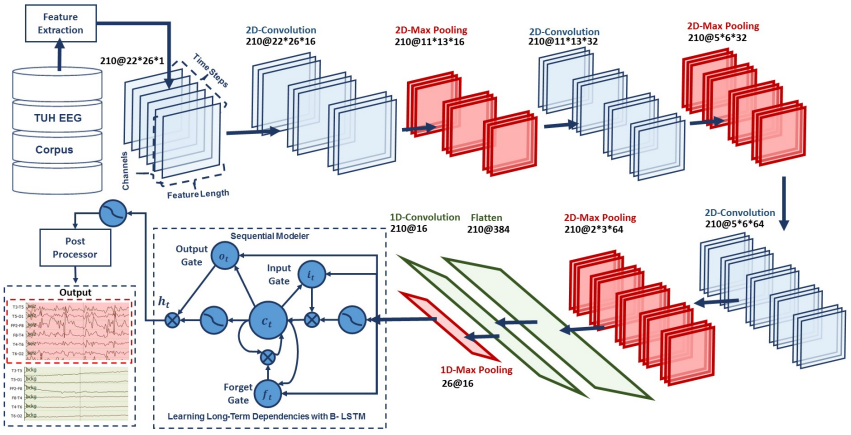

Fig. 8. A deep recurrent convolutional architecture

1.4.3. Temporal Event Modeling Using LSTMs

A final architecture we wish to consider is a relatively straightforward variation of an LSTM network. LSTMs are a special type of recurrent neural network which contains forget and output gates to control the information flow during its recurrent passes. LSTM networks have proven to be outperform conventional RNNs, HMMs and other sequence learning methods in numerous applications such as speech recognition and handwriting recognition. Our first implementation of LSTM was a hybrid network of both HMM and LSTM networks. A block diagram of HMM/LSTM system is shown in Fig. 9. Similar to the HMM/SdA model discussed before, the input to the second layer of the system, which is the first layer of LSTMs, is a vector of dimension 2 × 22 × window length. We use PCA to reduce the dimensionality of the input vector to 20 and pass it to the LSTM model. A window size of 41 secs (41 epochs at 1 sec per epoch) is used for a 32-node single hidden layer LSTM network. The final layer uses a dense neuron with a sigmoid activation function. The parameters of the models are optimized to minimize the error using a cross-entropy loss function and Adam.

Next, we use a 3-layer LSTM network model. Identification of a seizure event is done based on an observation of a specific type of epileptiform activity called “spike and wave discharges”. The evolution of these activities across time helps identify a seizure event. These events can be observed on individual channels. Once observed, the seizures can be confirmed based on their focality, signal energy and its polarity across spatially close channels. The architecture is shown in Fig. 10.

In the preprocessing step, we extract a 26-dimensional feature vector for an 11-frame context centered around the current frame. The output dimensionality for each frame is 10 x 26 (left) + 26 (center) + 10 x 26 (right) = 546. The static LSTM cells are used with a fixed batch size of 64 and a window size of 7 seconds. The data is randomly split into subsets where 80% is used for training and 20% is used for cross-validation during optimization. The features are normalized and scaled down to a range of [0, 1] on a file basis, which helps the gradient descent algorithm (and its variants) to converge much faster. Shuffling was performed on batches to avoid training biases.

The network includes 3 LSTM layers with (256, 64, 16) hidden layers followed by a 2-cell dense layer. The activation function used for all LSTM layers is a hyperbolic tangent function, tanh(), except for the final layer, which uses a softmax function to compress the range of output values to

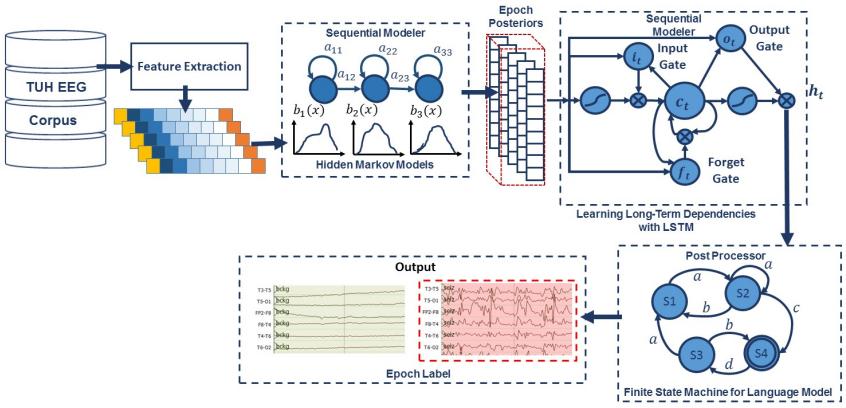

Fig. 9. A hybrid architecture that integrates HMM and LSTM.

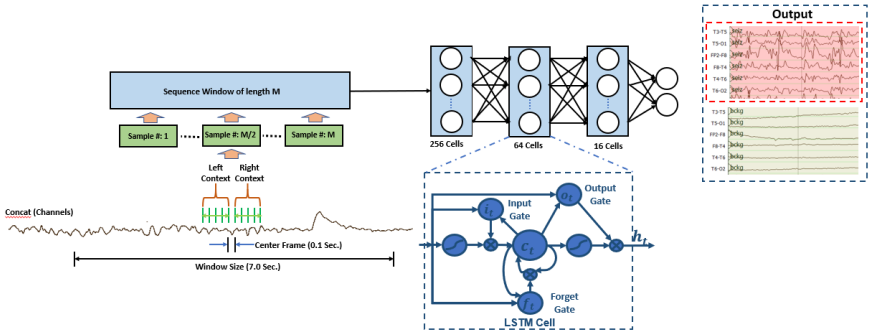

Fig. 10. A channel-based long short-term memory (LSTM) architecture

[0,1] so they resemble posterior probabilities. A cross-entropy function is used for calculating loss. Stochastic gradient descent with Nesterov momentum is used for optimization. Nesterov momentum attempts to increase the speed of training by introducing a momentum term based on accumulated gradients of its previous steps and a correction term in the direction of the current gradient. This tends to reduce the amount of overshoot during optimization.

The optimization is performed on the training data at a very high learning rate of 1.0 for the first five epochs. Cross-validation is performed after each epoch. After five epochs, if the cross-validation loss stagnates for three consecutive epochs (referred to as "patience = 3"), learning rates are halved after each iteration until it anneals to zero. If the model fails to show consistent performance on a cross-validation set, then it reverts to the previous epoch's weights and restarts training until convergence. This method helps models avoid overfitting on the training data as long as the training and cross-validation sets are equally diverse.

The outputs of the models are fed to a postprocessor which is described in more detail in Section 1.5. This postprocessor is designed based on domain knowledge and observed system behavior to remove spurious and misleading detections. This is implemented to incorporate spatial context. The postprocessor sets a threshold for hypothesis confidence, the minimum number of channels for target event detection and a duration constraint which must be satisfied for detection. For example, if multiple channels consistently detected spike and wave discharges in the same 9-second interval, this event would be permitted as a valid output. Outputs from a fewer number of channels or over a smaller duration of time would be suppressed.

We have now presented a considerable variety of deep learning architectures. It is difficult to predict which architecture performs best on a given task without extensive experimentation. Hence, in the following section, we review a wide-ranging study of how these architectures perform on the TUSZ seizure detection task.

1.5. Experimentation

Machine learning is at its core an experimental science when addressing real-world problems of scale. Real world data is complex and poses many challenges that require a wide variety of technologies to solve and can mask the benefits of one specific algorithm. Therefore, it is important that a rigorous evaluation paradigm be used to guide architecture decisions. In this section, we are focusing on the TUSZ Corpus because it is a very comprehensive dataset and it offers a very challenging task.

The evaluation of machine learning algorithms in biomedical fields for applications involving sequential data lacks standardization. Common quantitative scalar evaluation metrics such as sensitivity and specificity can often be misleading depending on the requirements of the application. Evaluation metrics must ultimately reflect the needs of users yet be sufficiently sensitive to guide algorithm development. Feedback from critical care clinicians who use automated event detection software in clinical applications has been overwhelmingly emphatic that a low false alarm rate, typically measured in units of the number of errors per 24 hours, is the single most important criterion for user acceptance. Though using a single metric is not often as insightful as examining performance over a range of operating conditions, there is a need for a single scalar figure of merit. Shah et al. discuss the deficiencies of existing metrics for a seizure detection task and propose several new metrics that offer a more balanced view of performance. In this section, we compare the architectures previously described using one of these measures, the Any-Overlap Method (OVLP). We also provide detection error tradeoff (DET) curves.

1.5.1. Evaluation Metrics

Researchers in biomedical fields typically report performance in terms of sensitivity and specificity. In a two-class classification problem such as seizure detection, we can define four types of errors:

- True Positives (TP): the number of 'positives' detected correctly
- True Negatives (TN): the number of 'negatives' detected correctly
- False Positives (FP): the number of 'negatives' detected as 'positives'
- False Negatives (FN): the number of 'positives' detected as 'negatives'

Sensitivity (TP/(TP+FN)) and specificity (TN/(TN+FP)) are derived from these quantities. There are a large number of auxiliary measures that can be calculated from these four basic quantities that are used extensively in the literature. For example, in information retrieval applications, systems are often evaluated using accuracy ((TP+TN)/(TP+FN+TN+FP)), precision (TP/(TP+FP)), recall (another term for sensitivity) and F1 score ((2•Precision•Recall)/(Precision + Recall)). However, none of these measures address the time scale on which the scoring must occur or how you score situations where the mapping of hypothesized events to reference events is ambiguous. These kinds of decisions are critical in the interpretation of scoring metrics such as sensitivity for many sequential decoding tasks such as automatic seizure detection.

In some applications, it is preferable to score every unit of time. With multichannel signals, such as EEGs, scoring for each channel for each unit of time might be appropriate since significant events such as seizures occur on a subset of the channels present in the signal. However, it is more common in the literature to simply score a summary decision per unit of time, such as every 1 sec, that is based on an aggregation of the per-channel inputs (e.g., a majority vote). We refer to this type of scoring as epoch-based. An alternative, that is more common in speech and image recognition applications, is term-based, in which we consider the start and stop time of the event, and each event identified in the reference annotation is counted once. There are fundamental differences between the two conventions. For example, one event containing many epochs will count more heavily in an epoch-based scoring scenario. Epoch-based scoring generally weights the duration of an event more heavily since each unit of time is assessed independently.

Term-based metrics score on an event basis and do not count individual frames. A typical approach for calculating errors in term-based scoring is the Any-Overlap Method (OVLP). This approach is

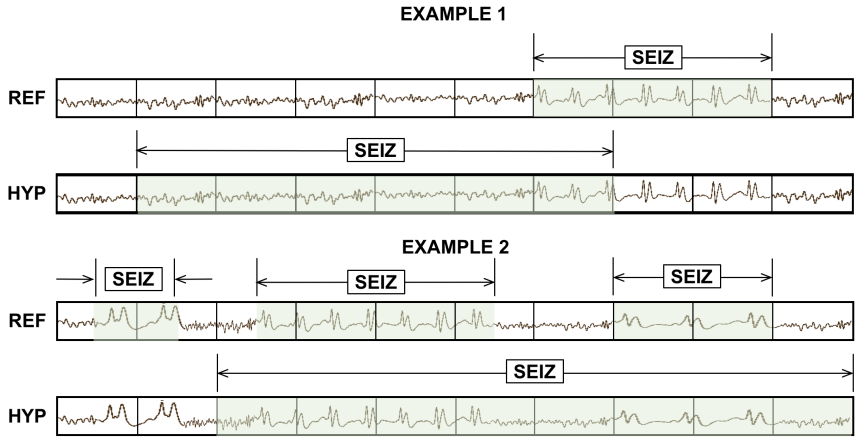

Fig. 11. OVLP scoring is very permissive about the degree of overlap between the reference and hypothesis. For example, in Example 1, the TP score is 1 with no false alarms. In Example 2, the system detects 2 out of 3 seizure events, so the TP and FN scores are 2 and 1 respectively.

illustrated in Fig. 11. TPs are counted when the hypothesis overlaps with the reference annotation. FPs correspond to situations in which a hypothesis does not overlap with the reference.

OVLP is a more permissive metric that tends to produce much higher sensitivities. If an event is detected in close proximity to a reference event, the reference event is considered correctly detected. If a long event in the reference annotation is detected as multiple shorter events in the hypothesis, the reference event is also considered correctly detected. Multiple events in the hypothesis annotation corresponding to the same event in the reference annotation are not typically counted as FAs. Since the FA rate is a very important measure of performance in critical care applications, this is another cause for concern. However, since OVLP metric is the most popular choice in the neuroengineering community, we present our results in terms of OVLP.

Note that results are still reported in terms of sensitivity, specificity and false alarm rate. But, as previously mentioned, how one measures the errors that contribute to these measures is open for interpretation. Shah et al. studied this problem extensively and showed that many of these measures correlate and are not significantly different in terms of the rank ordering and statistical significance of scoring differences for the TUSZ task. We provide a software package that allows researchers to replicate our metrics and reports on many of the most popular metrics.

1.5.2. Postprocessing with Heuristics Improves Performance

Because epoch-based scoring produces a hypothesis every epoch (1 sec in this case), and these are scored against annotations that are essentially asynchronous, there is an opportunity to improve performance by examining sequences of epochs and collapsing multiple events into a single hypothesis. We have experimented with heuristic approaches to this as well as deep learning-based approaches and have found no significant advantage for the deep learning approaches. As is well known in machine learning research, a good heuristic can be very powerful. We apply a series of heuristics, summarized in Fig. 12, to improve performance. These heuristics are very important in reducing the false alarm rate to an acceptable level.

The first heuristic we apply is a popular method that focuses on a model's confidence in its output. Probabilistic filters are implemented to only consider target events which are detected above a

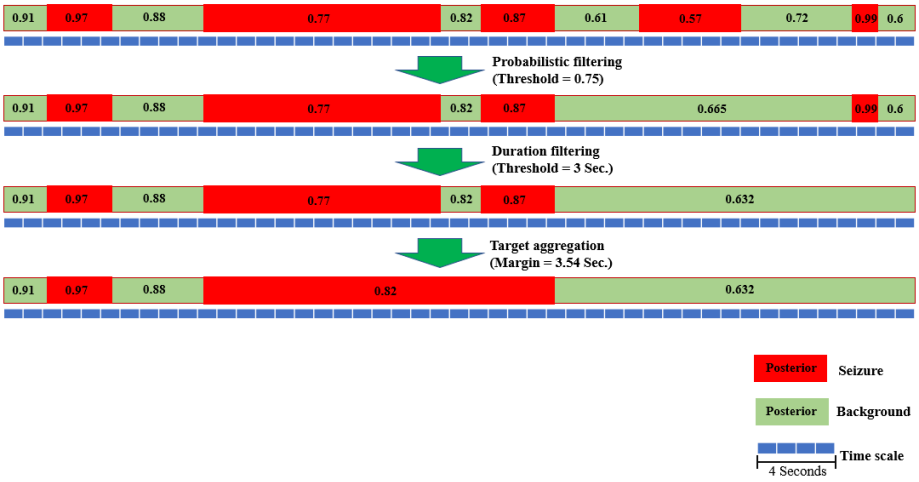

Fig. 12. An illustration of the postprocessing algorithms used to reduce the FA rate

specified probability threshold. This method tends to suppress spurious long duration events (e.g., slowing) and extremely short duration events (e.g., muscle artifacts). This decision function is applied on the seizure (target) labels only. We compare each seizure label's posterior with the threshold value. If the posterior is above the threshold, the label is kept as is; otherwise, it is changed to the non-seizure label, which we denote "background."

Our second heuristic was developed after performing extensive error analysis. The most common types of errors we observed were false detections of background events as seizures (FPs) which tend to occur in bursts. Usually these erroneous bursts occur for a very small duration of time (e.g., 3 to 7 seconds). To suppress these, any seizure event whose duration is below a specified threshold is automatically considered as a non-seizure, or background, event.

Finally, we also implement a smoothing method that collapses sequences of two seizure events separated by a background event into one long seizure event. This is typically used to eliminate spurious background events. If seizures are observed in clusters separated by small intervals of time classified as background events, these isolated events are most likely part of one longer seizure event. In this method, we apply a nonlinear function that computes a pad time to extend the duration of an isolated event. If the modified endpoint of that event overlaps with another seizure event, the intervening background event is eliminated. We used a simple regression approach to derive a quadratic function that produces a padding factor: $w(x) = -0.0083d^2 + 0.45d - 0.66$, were d is the duration of the event. This method tends to reduce isolated background events when they are surrounding by seizure events, thereby increasing the specificity.

The combination of these three postprocessing methods tends to decrease sensitivity slightly and reduce false alarms by two orders of magnitude, so their impact is significant. The ordering in which these methods is applied is important. We apply them in the order described above to achieve optimal performance.

1.5.3. A Comprehensive Evaluation of Hybrid Approaches

A series of experiments was conducted to optimize the feature extraction process. Subsequent attempts to replace feature extraction with deep learning-based approaches have resulted in a slight

degradation in performance. A reasonable tradeoff between computational complexity and performance was to split the 10 sec window, popular with neurologists who manually interpret these waveforms, into 1 sec epochs, and to further subdivide these into 0.1 sec frames. Hence, features were computed every 0.1 sec using a 0.2 sec overlapping analysis window. The output of the feature extraction system is 22 channels of data, where in each channel, a feature vector of dimension 26 corresponds to every 0.1 secs. This type of analysis is very compatible with the way HMM systems operate, so it was a reasonable starting point for this work.

We next evaluated several architectures using these features as inputs on TUSZ. These results are presented in Table 2. The related DET curve is illustrated Fig. 13. An expanded version of the DET curve in Fig. 13 that compares the performance of these architectures in a region of the DET curve where the false positive rate, also known as the false alarm (FA) rate, is low is presented in Fig. 14. Since our focus is achieving a low false alarm rate, behavior in this region of the DET curve is very important. As previously mentioned, these systems were evaluated using the OVLP method, though results are similar for a variety of these metrics.

It is important to note that the accuracy reported here is much lower than what is often published in the literature on other seizure detection tasks. This is due to a combination of factors including (1) the neuroscience community has favored a more permissive method of scoring that tends to produce much higher sensitivities and lower false alarm rates; and (2) TUSZ is a much more difficult task than any corpus previously released as open source. The evaluation set was designed to be representative of common clinical issues and includes many challenging examples of seizures. We have achieved much higher performance on other publicly available tasks, such as the Children's Hospital of Boston MIT (CHB-MIT) Corpus, and demonstrated that the performance of these techniques exceeds that of published or commercially-available technology. TUSZ is simply a much more difficult task and one that better represents the clinical challenges this technology faces.

Also, note that the HMM baseline system, which is shown in the first row of Table 2, and channel-based LSTM, which is shown in the last row of Table 2, operate on each channel independently. The other methods consider all channels simultaneously by using a supervector that is a concatenation of the feature vectors for all channels. The baseline HMM system only classifies epochs (1 sec in duration) using data from within that epoch. It does not look across channels or across multiple epochs when performing epoch-level classification.

Table 2. Performance of the proposed architectures on TUSZ

System	Sensitivity	Specificity	FA/24 Hrs.
HMM	30.32%	80.07%	244
HMM/SdA	35.35%	73.35%	77
HMM/LSTM	30.05%	80.53%	60
IPCA/LSTM	32.97%	77.57%	73
CNN/MLP	39.09%	76.84%	77
CNN/GRU	30.83%	91.49%	21
ResNet	30.50%	94.24%	13
CNN/LSTM	30.83%	97.10%	6
Channel-Based LSTM	39.46%	95.20%	11

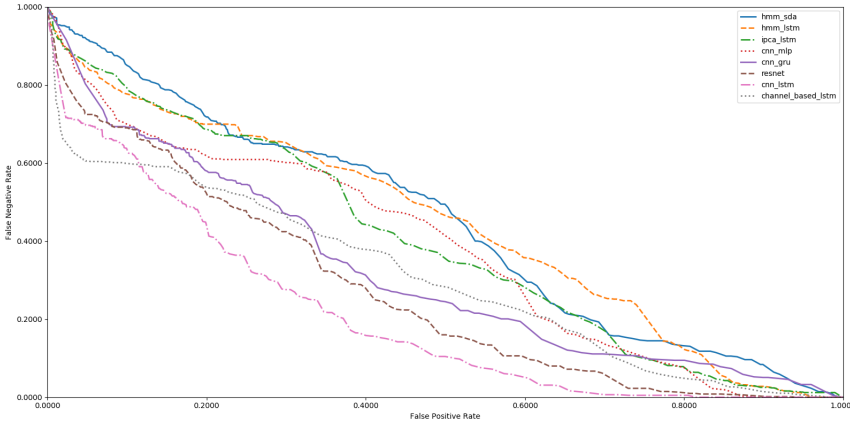

Fig. 13. A DET curve comparison of the proposed architectures on TUSZ.

From Table 2 we can see that adding a deep learning structure for temporal and spatial analysis of EEGs can decrease the false alarm rate dramatically. Further, by comparing the results of HMM/SdA with HMM/LSTM, we find that a simple one-layer LSTM performs better than 3 layers of SdA due to LSTM's ability to explicitly model long-term dependencies. Note that in this case the complexity and training time of these two systems is comparable.

The best overall systems shown in Table 2 are CNN/LSTM and channel-based LSTM. CNN/LSTM is a doubly deep recurrent convolutional structure that models both spatial relationships (e.g., cross-channel dependencies) and temporal dynamics (e.g., spikes). For example, CNN/LSTM does a much better job rejecting artifacts that are easily confused with spikes because these appear on only a few channels, and hence can be filtered based on correlations between channels. The depth of the convolutional network is important since the top convolutional layers tend to learn generic features while the deeper layers learn dataset specific features. Performance degrades if a single convolutional layer is removed. For example, removing any of the middle convolutional layers results in a loss of about 4% in the sensitivity. However, it is important to note that the computational complexity of the channel-based systems is significantly higher than the systems that aggregate channel-based features into a single vector, since the channel-based systems are decoding each channel independently.

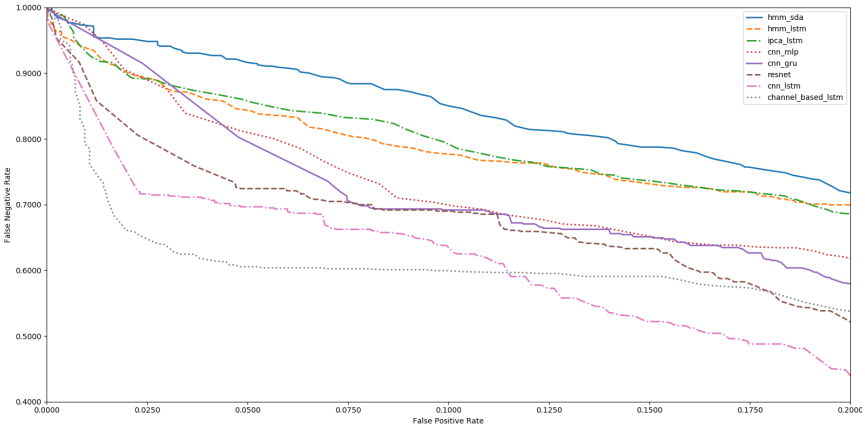

Fig. 14. An expanded comparison of performance in a region where the FP rate is low.

As shown in Fig. 13 and Fig. 14, we find that CNN/LSTM has a significantly lower FA rate than CNN/GRU. We speculate that this is due to the fact that while a GRU unit controls the flow of information like the LSTM unit, it does not have a memory unit. LSTMs can remember longer sequences better than GRUs. Since seizure detection requires modeling long distance relationships, we believe this explains why there is a difference in performance between the two systems.

The time required for training for CNN/GRU was 10% less than CNN/LSTM. The training time of these two systems is comparable since most of the cycles are spent training the convolutional layers. We also observe that the ResNet structure improves the performance of CNN/MLP, but the best overall system is still CNN/LSTM.

We have also conducted an open-set evaluation of the best systems, CNN/LSTM and channel-based LSTM, on a completely different corpus – DUSZ. These results are shown in Table 3. A DET curve is shown in Fig. 15. This is an important evaluation because none of these systems were exposed to DUSZ data during training or development testing. Parameter optimizations were performed only on TUSZ data. As can be seen, at high FA rates, performance between the two systems is comparable. At low FA rates, however, CNN/LSTM performance on TUSZ is lower than on DUSZ. For channel-based LSTM, in the region of low FA rate, performance on TUSZ and DUSZ is very similar. This is reflected by the two middle curves in Fig. 15. The differences in performance for channel-based LSTM when the data changes are small. However, for CNN/LSTM, which gives the best overall performance on TUSZ, performance decreases rapidly on DUSZ. Recall that we did not train these systems on DUSZ – this is true open set testing. Hence, we can conclude in this limited study that channel-based LSTM generalized better than the CNN/LSTM system.

Table 3. A comparison of several CNN and LSTM architectures on DUSZ

System	Data	Sensitivity	Specificity	FA/24 Hrs.
CNN/LSTM	TUSZ	30.83%	97.10%	6
CNN/LSTM	DUSZ	33.71%	70.72%	40
Channel-Based LSTM	TUSZ	39.46%	95.20%	11
Channel-Based LSTM	DUSZ	42.32%	86.93%	14

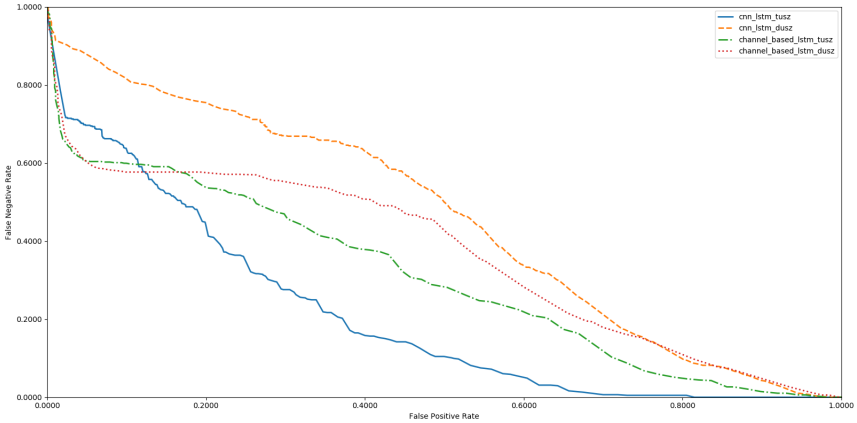

Fig. 15. Performance of CNN/LSTM and channel-based LSTM on TUSZ and DUSZ.

1.5.4. Optimization of Core Components

Throughout these experiments, we observed that the choice of optimization method had a considerable impact on performance. The CNN/LSTM system was evaluated using a variety of optimization methods, including Stochastic gradient descent (SGD), RMSprop, Adagrad, Adadelta, Adam, Adamax and Nadam. These results are shown in Table 5. The best performance is achieved with Adam, a learning rate of $\alpha = 0.0005$, a learning rate decay of 0.0001, exponential decay rates of $\beta_1 = 0.9$ and $\beta_2 = 0.999$ for the moment estimates and a fuzz factor of $\epsilon = 10^{-8}$. Table 5 also illustrates that Nadam delivers comparable performance to Adam. Adam combines the advantages of AdaGrad which works well with sparse gradients, and RMSProp which works well in non-stationary settings.

Similarly, we evaluated our CNN/LSTM using different activation functions, as shown in Table 4. ELU delivers a small but measurable increase in sensitivity, and more importantly, a reduction in false alarms. The ELU activation function is defined as:

$$\mathrm{f(x)} = \begin{cases} x & x > 0 \\ \alpha.(e^x - 1) & x \leq 0 \end{cases}, \qquad (8)$$

where α is slope of negative section. The derivative of the ELU activation function is:

$$\acute{\mathrm{f}}\mathrm{(x)} = \begin{cases} 1 & x > 0 \\ \alpha.e^x & x \leq 0 \end{cases}. \qquad (9)$$

The ReLU activation function is defined as:

$$\mathrm{f(x)} = \begin{cases} x & x > 0 \\ 0 & x \leq 0 \end{cases}. \qquad (10)$$

The corresponding derivative is:

Table 5. Comparison of optimization algorithms

System	Sensitivity	Specificity	FA/24 Hrs.
SGD	23.12%	72.24%	44
RMSprop	25.17%	83.39%	23
Adagrad	26.42%	80.42%	31
Adadelta	26.11%	79.14%	33
Adam	30.83%	97.10%	6
Adamax	29.25%	89.64%	18
Nadam	30.27%	92.17%	14

Table 4. A comparison of activation functions

System	Sensitivity	Specificity	FA/24 Hrs.
Linear	26.46%	88.48%	25
Tanh	26.53%	89.17%	21
Sigmoid	28.63%	90.08%	19
Softsign	30.05%	90.51%	18
ReLU	30.51%	94.74%	11
ELU	30.83%	97.10%	6

$$\acute{f}(x) = \begin{cases} 1 & x > 0 \\ 0 & x \leq 0 \end{cases}. \qquad (11)$$

ELU is very similar to ReLU except for negative inputs. ReLUs and ELUs accelerate learning by decreasing the gap between the normal gradient and the unit natural gradient. ELUs push the mean towards zero but with a significantly smaller computational footprint. In the region where the input is negative ($x < 0$), since an ReLU's gradient is zero, the weights will not get adjusted. Those neurons which connect into that state will stop responding to variations in error or input. This is referred to as the dying ReLU problem. But unlike ReLUs, ELUs have a clear saturation plateau in their negative region, allowing them to learn a more robust and stable representation.

Determining the proper initialization strategy for the parameters in the model is part of the difficulty in training. Hence, we investigated a variety of initialization methods using the CNN/LSTM structure introduced in Fig. 8. These results are presented in Table 6. The related DET curve is illustrated in Fig. 16. In our experiments, we observed that proper initialization of weights in a convolutional recurrent neural network is critical to convergence. For example, initialization of tensor values to zero or one completely stalled the convergence process. Also, as we can see in Table 6, the FA rate of the system in the range of 30% sensitivity can change from 7 to 40, for different initialization methods. This decrease in performance and deceleration of convergence arises because some initializations can result in the deeper layers receiving inputs with small variances, which in turn slows down back propagation, and retards the overall convergence process.

Best performance is achieved using orthogonal initialization. This method is a simple yet effective way of combatting exploding and vanishing gradients. In orthogonal initialization, the weight matrix is chosen as a random orthogonal matrix, i.e., a square matrix W for which $W^T W = I$. Typically, the orthogonal matrix is obtained from the QR decomposition of a matrix of random numbers drawn from a normal distribution. Orthogonal matrices preserve the norm of a vector and their eigenvalues have an absolute value of one. This means that no matter how many times we perform repeated matrix multiplication, the resulting matrix doesn't explode or vanish. Also, in orthogonal matrices, columns and rows are all orthonormal to one another, which helps the weights to learn different input features. For example, if we apply orthogonal initialization on a CNN architecture, in each layer, each channel has a weight vector that is orthogonal to the weight vectors of the other channels.

Table 6. A comparison of initialization methods

System	Sensitivity	Specificity	FA/24 Hrs.
Orthogonal	30.8%	96.9%	7
Lecun Uniform	30.3%	96.5%	8
Glorot Uniform	31.0%	94.2%	13
Glorot Normal	29.5%	92.4%	18
Variance Scaling	31.8%	92.1%	19
Lecun Normal	31.8%	92.1%	19
He Normal	31.3%	91.1%	22
Random Uniform	30.2%	90.0%	25
Truncated Normal	31.6%	87.8%	31
He Uniform	29.2%	85.1%	40

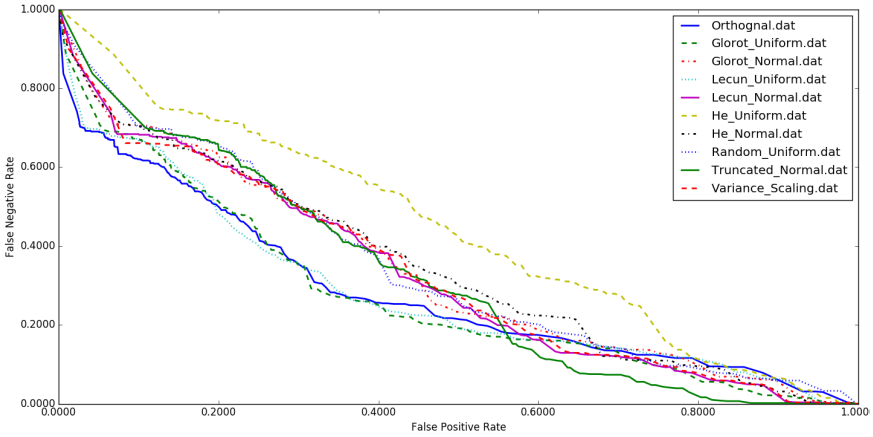

Fig. 16. A comparison of different initialization methods for CNN/LSTM

Overfitting is a serious problem in deep neural nets with many parameters. We have explored five popular regularization methods to address this problem. The techniques collectively known as L1, L2 and L1/L2 prevent overfitting by adding a regularization term to the loss function. The L1 regularization technique, also known as Lasso regression, is defined as adding the sum of weights to the loss function:

$$Cost\ Function = Loss\ Function + \lambda \sum_{i=1}^{k} |w_i| \ ,(12)$$

where w is the weight vector and λ is a regularization parameter. The L2 technique, also known as ridge regression, is defined as adding the sum of the square of the weights to the loss function:

$$Cost\ Function = Loss\ Function + \lambda \sum_{i=1}^{k} w_i^2 \ .(13)$$

The L1/L2 technique is a combination of both techniques:

$$Cost\ Function = Loss\ Function + \lambda \sum_{i=1}^{k} |w_i| + \lambda \sum_{i=1}^{k} w_i^2 \ . \qquad (14)$$

In an alternative approach, we used dropout to prevents units from excessively co-adapting by randomly dropping units and their connections from the neural network during training.

We also studied the impact of introducing zero-centered Gaussian noise to the network. In this regularization method, which is considered a random data augmentation method, we add zero-centered Gaussian noise with a standard deviation of 0.2 to all hidden layers in the network as well as the visible or input layer. The results of these experiments are presented in Table 7 along with a DET curve in Fig. 17. While L1/L2 regularization has the best overall performance, in the region where FA rates are low, the dropout method delivers a lower FA rate. The primary error modalities observed were false alarms generated during brief delta range slowing patterns such as intermittent rhythmic delta activity. Our closed-loop experiments demonstrated that all regularization methods presented in Table 7, unfortunately, tend to increase the false alarm rate for slowing patterns.

Finally, in Fig. 18, an example of an EEG that is generated by the DCGAN structure of Fig. 6 is shown. Note that to generate these EEGs, we use a generator block in DCGAN in which each EEG signal has a 7 sec duration. We apply a 25 Hz low pass filter on the output of DCGAN, since most of the cerebral signals observed in scalp EEGs fall in the range of 1–20 Hz (in standard clinical recordings, activity below or above this range is likely to be an artifact). Unfortunately, in a simple

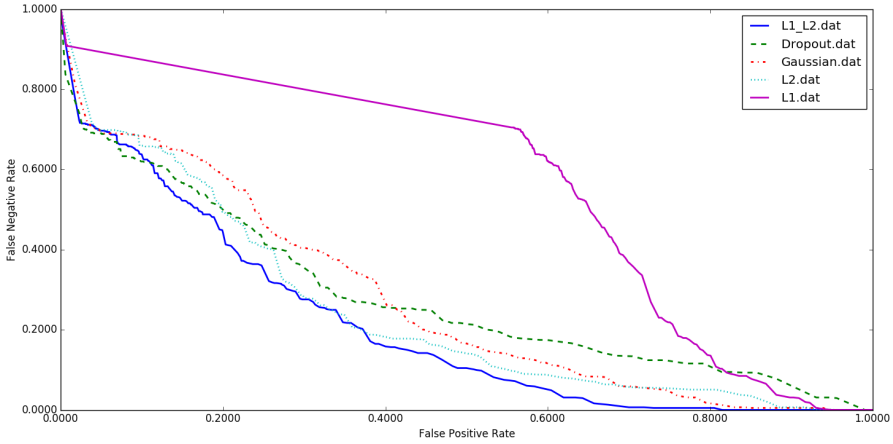

Fig. 17. A comparison of different regularization methods for CNN/LSTM

pilot experiment in which we randomly mixed actual EEGs with synthetic EEGs, expert annotators could easily detect the synthetic EEGs, which was a bit discouraging. Seizures in the synthetic EEGs were sharper and more closely resembled a slowing event. Clearly, more work is needed with this architecture.

However, our expert annotators also noted that the synthetic EEGs did exhibit focality. An example of focality is when activity is observed on the CZ-C4 channel, we would expect to observe the inverse of this pattern on the C4-T4 channel. As can be seen in Fig. 18, in last two seconds of the generated EEG, we observe slowing activity on the CZ-C4 channel and the inverse pattern of the same slowing activity on the C4-T4 channel. Hence, it is possible to generate synthetic multi-channel EEG signals with DCGAN that resemble clinical EEGs. However, DCGAN is not yet at the point where it is generating data that is resulting in an improvement in the performance of our best systems.

1.6. Summary

EEGs remain one of the main clinical tools that physicians use to understand brain function. New applications of EEGs are emerging including diagnosis of head trauma-related injuries which offer the potential to vastly expand the market for EEGs. A board-certified EEG specialist is required by law to interpret an EEG and produce a diagnosis. Since it takes several years of additional training post-medical school for a physician to qualify as a clinical specialist, the ability to generate data far exceeds the available expertise to interpret these data, creating a critical bottleneck. Despite rapid advances in deep learning in recent years, automatic interpretation of EEGs is still a very challenging problem.

Table 7. A comparison of performance for different regularizations

System	Sensitivity	Specificity	FA/24 Hrs.
L1/L2	30.8%	97.1%	6
Dropout	30.8%	96.9%	7
Gaussian	30.8%	95.8%	9
L2	30.2%	95.6%	10
L1	30.0%	43.7%	276

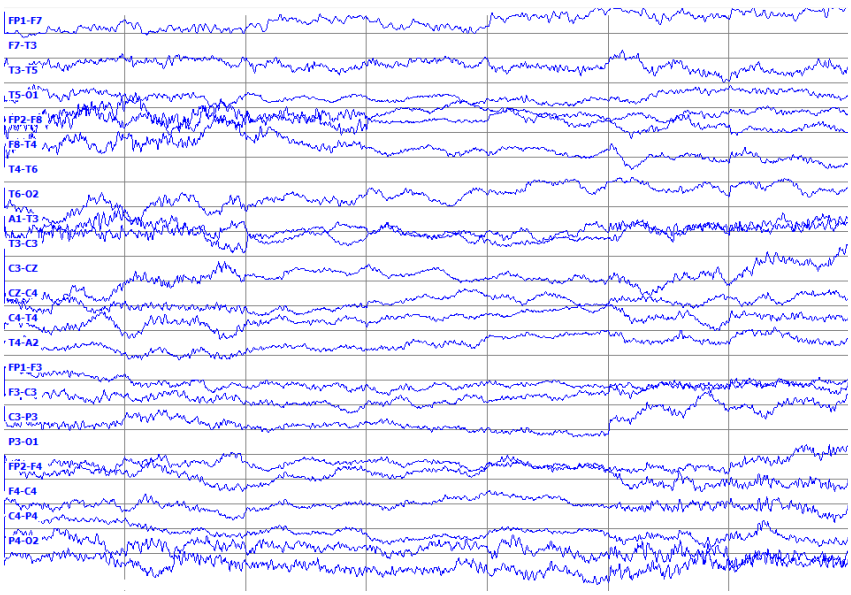

Fig. 18. Synthetic EEG waveforms generated using DCGAN.

We have introduced a variety of deep learning architectures for automatic classification of EEGs including a hybrid architecture that integrates CNN and LSTM technology. Two systems are particularly promising: CNN/LSTM and channel-based LSTM. While these architectures deliver better performance than other deep structures, their performance still does not meet the needs of clinicians. Human performance on similar tasks is in the range of 75% sensitivity with a false alarm rate of 1 per 24 hours. The false alarm rate is particularly important to critical care applications since it impacts the workload experienced by healthcare providers.

The primary error modalities for our deep learning-based approaches were false alarms generated during brief delta range slowing patterns such as intermittent rhythmic delta activity. A variety of these types of artifacts have been observed during inter-ictal and post-ictal stages. Training models on such events with diverse morphologies is potentially one way to reduce the remaining false alarms. This is one reason we are continuing our efforts to annotate a larger portion of TUSZ.

We are also exploring the potential of supervised GAN frameworks for spatio-temporal modeling of EEGs. Most of the research on GANs is focused on either unsupervised learning or supervised learning using conditional GANs. Given that the annotation process to produce accurate labels is expensive and time-consuming, we are exploring semi-supervised learning in which only a small fraction of the data has labels. GANs can be used to perform semi-supervised classification by using a generator-discriminator pair to learn an unconditional model of data and then tune the discriminator using the small amount of labeled data for prediction.

We are also continuing to manually label EEG data. We invite you to register at our project web site, *www.isip.piconepress.com/projects/tuh_eeg*, to be kept aware of the latest developments.

2. Automatically recognize critical concepts in an EMR

Goals Specific to Aim 2 in the main project: *Automatically recognize critical concepts in an EMR:* (1) EEG activities and their attributes, (2) EEG events, (3) medical problems, (4) medical treatments and (5) medical tests mentioned in the narratives of the EEG reports, along with their inferred forms of *modality* and *polarity*. When we considered the recognition of the modality, we took advantage of the definitions used in the 2012 i2b2 challenge on evaluating temporal relations in medical text. In that challenge, modality was used to capture whether a medical event discerned from a medical record actually happened, is merely proposed, mentioned as conditional, or described as possible. We extended this definition such that the possible modality values of "factual", "possible", and "proposed" indicate that medical concepts mentioned in the EEGs are actual findings, possible findings and findings that may be true at some point in the future, respectively. For identifying polarity of medical concepts in EEG reports, we relied on the same definition used in the 2012 i2b2 challenge, considering that each concept can have either a "positive" or a "negative" polarity, depending on any absent or present negation of its finding. Through the identification of modality and polarity of the medical concepts, we aimed to capture the neurologist's beliefs about the medical concepts mentioned in the EEG report. Some of the medical concepts mentioned in the EEG reports that describe the clinical picture of a patient are similar to those evaluated in the 2010 i2b2 challenge, as they represent medical problems, tests and treatments, thus we could take advantage of our participation in that challenge and use many of the features we have developed for automatically recognizing such medical concepts. However, EEG reports also contain a substantial number of mentions of EEG activities and EEG events, as they discuss the EEG test.

In the third year of the project, the team from the University of Texas at Dallas has developed the ability to automatically infer missing and unspecified information from the EEG reports. More specifically, we developed inference methods capable to generate the impression sections and the clinical correlations sections. When these sections are absent from the EEG report, critical concepts are missing and cannot be indexed in the patient cohort retrieval system. After we generated the missing information, we used the Multi-task Active Deep Learning (MTADL) framework for annotating (1) EEG activities and their attributes, (2) EEG events, (3) medical problems, (4) medical treatments and (5) medical tests mentioned in the narratives of the reports, along with their inferred forms of *modality* and *polarity*. The MTDADL framework was developed in the second year of the project.

Inferring the over-all impression from EEG reports is a challenging problem because the over-all impression is informed by the neurologist's subjective interpretation of the EEG recording as well as his or her neurological expertise and accumulated experience. In fact, automatically inferring the over-all impression requires accounting for the role of neurological knowledge and experience. Our deep learning model is able to automatically infer such knowledge by processing the natural language within EEG reports. The model operates in the following steps:

[Step 1] word-level features are automatically extracted based on their context by incorporating the skip-gram model popularized by the Word2Vec framework;

[Step 2] report-level features are automatically extracted using either (i) a deep averaging network (DAN), or (ii) a recurrent neural network (RNN); and

[Step 3] the most likely over-all impression is predicted from word- and report-level features through densely-connected "deep" neural layers.

When writing an EEG report, the neurologist typically documents their over-all impression of the EEG: whether it indicates normal or abnormal brain activity. However, this information is not always explicitly stated in the impression section of an EEG report and must sometimes be inferred by the reader. Figure 1 illustrates three EEG reports indicating (a) an over-all impression of NORMAL, (b) an over-all impression of ABNORMAL, and (c) an underspecified over-all impression. Note, in Figure 1, we have normalized the order and titles of the sections in each EEG report; in reality, however, we observed a total of 1,176 unique section titles in our collection.

INTRODUCTION: The EEG was performed using the standard 10/20 electrode placement system with an EKG electrode and anterior temporal electrodes. The EEG was recorded during wakefulness and photic stimulation, as well as hyperventilation, activation procedures were performed.	**INTRODUCTION:** Digital video EEG is performed at the bedside using standard 10-20 system of electrode placement with one channel of EKG. The patient is sitting out of her bed. She is very confused and poorly cooperative.	**INTRODUCTION:** Digital video EEG is performed at bedside using standard 10-20 system of electrode placement with 1 channel of EKG. The patient is agitated.
MEDICATIONS: Depakote ER	**MEDICATIONS:** Keppra.	**MEDICATIONS:** Keppra, Aricept, Senna, Aricept, ASA, famotidine
HISTORY: A 21-year-old man with a history of seizures since age 15. Has had five episodes since 2005, all tonic-clonic seizures with loss of consciousness lasting one to two minutes and postictal confusion.	**HISTORY:** An elderly woman with change in mental status, waxing and waning mental status, COPD, morbid obesity, and markedly abnormal EEG. Digital 3EG was done on June 27, 2011.	**HISTORY:** 84-year-old woman of unknown handedness with advanced dementia, failure to thrive, change in mental status, TIA, dementia.
DESCRIPTION: The EEG opens to a well-formed 9 to 10Hz posterior dominant rhythm, which is symmetrically reactive to eye opening and eye closing. There is a normal amount of frontal central beta rhythm seen. The recording is only seen during wakefulness and he has normal response to hyperventilation and photic stimulation.	**DESCRIPTION:** Much of the EEG includes muscle artifact. When she Is cooperative, there is a theta pattern with bursts of frontal delta. Muscle artifact is remarkable when the patient becomes a bit more agitated. As she goes off to sleep, the deltas slowed considerably. There are handful of triphasic waves noted. Heart rate 84 BPM.	**DESCRIPTION:** As the tracing opens, the patient has a lot of muscle activity. She seems to have facial twitching and grimacing and it almost looks like she has a suck or snout reflexes. Although the patient does not appear to interact with the physician in any way, this produces an alerting response with an increase in 5-7 hertz theta activity in the background. The overall background is 1 of shifting asymmetries with theta from side as with beta sometimes better represented on either side, shifting arrhythmic delta and intermittent, subtle attenuations in the background. Following admission of the Ativan, the EEG becomes somewhat more discontinuous.
IMPRESSION: Normal EEG in wakefulness.	**IMPRESSION:** This is an abnormal EEG due to 1. Prominent versus frontally predominant rhythmic delta. 2 Excess beta. 3. Excess theta.	**IMPRESSION:** This EEG is similar to the 2 previous studies this year which demonstrated a slow background. Each recording seems to demonstrate an increase in slowing. The administration of Ativan produced a somewhat discontinuous pattern as may be anticipated in a patient with advanced dementia.
CLINICAL CORRELATION: This awake EEG is normal. Please note that a normal EEG does not exclude the diagnosis of epilepsy.		**CLINICAL CORRELATION:** No epileptiform features were seen.
(a)	(b)	(c)

Figure 1. Examples of EEG reports with (a) an over-all impression of NORMAL, (b) an over-all impression of ABNORMAL, and (c) an *underspecified* over-all impression which does not state whether the EEG was normal or abnormal.

When producing an over-all impression, the neurologist interprets the EEG signal as well as the patient's clinical history, medications, and the setting of the EEG. For example, consider report

(b) from Figure 1: determining that the EEG was abnormal required identifying, among other findings, the frontal delta rhythm, while in report (c) the impression involves the drug Ativan and the patient's prior diagnoses of dementia. These examples show that automatically inferring the over-all impression requires accounting for high-level semantic information in EEG reports capturing the characteristics of the patient and the described EEG signal. Moreover, we observed that not all EEG reports included an impression section.

The UTD team developed an approach for automatically inferring the overall impression from an EEG report even when the impression section is omitted. To accomplish this, we combined deep neural learning with the largest collection of publicly available EEG reports – the Temple University Hospital (TUH) EEG Corpus. The TUH EEG Corpus contains 16,495 de-identified EEG reports generated at TUH between 2002 and 2013. We found that 15,313 reports contained a clear over-all impression, while 1,029 reports had a missing or underspecified over-all impression. To train and evaluate our model, we considered only the reports with a clear over-all impression and (1) *identified* the over-all impression (which was used as the gold-standard) and (2) *removed* the impression section from the report. This allowed us to design a deep neural network to predict the over-all impression for EEG reports without relying on the impression section. We used a standard 3:1:1 split for training, development, and testing.

When designing our deep neural network, we noticed that the natural language content of each EEG report was far from uniform. The number of sections, the title of sections, the number of sentences in each section, and the lengths of each sentence all varied between individual neurologists and individual reports. Moreover, when describing an EEG recording, each neurologist wrote in a different style: while some neurologists preferred terse economical language, others provided meticulous multi-paragraph discussions. Thus, it was necessary to design the deep neural network to be independent of the length (and style) of the language used by the neurologist. Our approach for determining the over-all impression from EEG reports takes advantage of recent advances in deep learning in order to (1) automatically preform high-level feature extraction from EEG reports and (2) determine the most likely overall impression based on trends observed in a large collection of EEG reports. High-level feature extraction was performed automatically and was accomplished in two steps. In the first step, we learned word-level features for every word used in any EEG report. In the second step, we learned how to combine and compose the word-level features to produce high-level features characterizing the report itself.

Formally, we represent each EEG report as a tensor, $\boldsymbol{R} \in \mathbb{R}^{N \times V}$, where N is the number of words in the report and V is the size of the *vocabulary* or number of unique words across all EEG reports in the training set (in our case, $V = 39{,}131$). Each row $\boldsymbol{R}_i$ is known as a *one-hot* vector which indicates that the i^{th} word in the report corresponds to the j^{th} word in the vocabulary in by assigning a value of one to $\boldsymbol{R}_{ij}$ and a value of zero to all elements. The overall impression of an EEG report (obtained from the removed *impression* section) is represented as $c \in C$ where $C = \{\text{NORMAL}, \text{ABNORMAL}\}$. The goal of the deep neural network presented in this paper is to determine the optimal parameters θ which are able to predict the correct assignment of c for a report $\boldsymbol{R}$:

$$\theta = \underset{\theta'}{\operatorname{argmax}} \sum_{(c,\boldsymbol{R}) \in \mathcal{X}} \log P(c \mid \boldsymbol{R}; \theta') \tag{1}$$

where $\mathcal{X}$ indicates the training set of EEG reports. Unfortunately, determining the over-all impression directly from the words in each report is difficult. For example, *spikes* and *sharp waves* typically indicate abnormal brain activity but can be non-pathologic if they occur in the temporal regions of the brain during sleep: small sharp spikes in the temporal region of the brain during sleep are known as *benign epileptiform transients of sleep* (BETS) and do not indicate an abnormal EEG. Consequently, to correctly predict the overall impression c, it is important to consider high-level features characterizing the content each report rather than individual words. We extract these features automatically as part of our deep learning architecture. Specifically, we factorize the distribution used in Equation 1 into three factors:

$$P(c \mid \boldsymbol{R}; \theta) = P(\boldsymbol{W} \mid \boldsymbol{R}) \cdot P(\boldsymbol{e} \mid \boldsymbol{W}) \cdot P(c \mid \boldsymbol{e}; \theta) \quad (2)$$

The three factors in Equation 2 correspond to the three steps used to train our deep learning model:

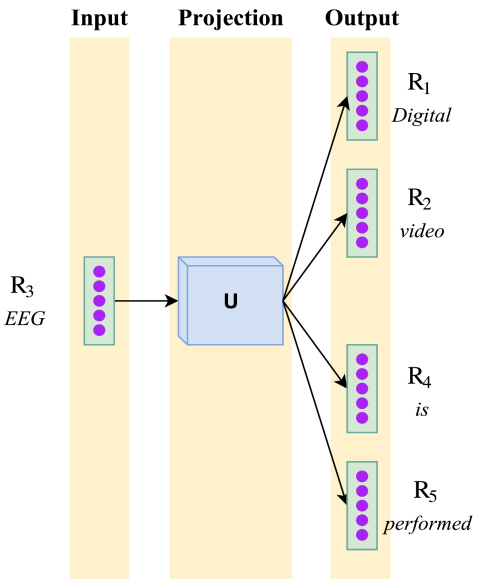

Figure 2. The skip-gram model used to learn word-level features for each word in an EEG report, shown on report (c) from Figure 1.

(1) produce a high-level feature representation $\boldsymbol{W}$ of every word in $\boldsymbol{R}$, i.e. $P(\boldsymbol{W} \mid \boldsymbol{R})$;

(2) create a single high-level feature representation $\boldsymbol{e}$ for the report itself by combining and composing the high-level feature representations of every word in the report, i.e. $P(\boldsymbol{e} \mid \boldsymbol{W})$; and

(3) determine the most likely over-all impression c for the report based on its high-level feature representation $\boldsymbol{e}$, i.e. $P(c \mid \boldsymbol{e}; \theta)$.

Next, we will describe each of these steps in detail followed by a description of the training and application of our model to infer underspecified over-all impressions from EEG reports, as well as details on how model parameters were selected and the model parameters used in our experiments.

Learning Word-Level Features from EEG Reports

We determined a high-level feature representation for each possible word $v \in [1, V]$ by examining the context around that word in each report, where V is the size of the vocabulary (described above). To do this, we adapt the *skip-gram model*. The skip-gram model learns a single feature representation for every word in the vocabulary based on all of its *contexts* across all EEG reports in the training set. Specifically, we learn the projection matrix $\boldsymbol{U} \in \mathbb{R}^{V \times K}$ where each row $\boldsymbol{U}_v$ is the high-level feature representation of the v^{th} word in the vocabulary. Figure 2 shows the architecture of the skip-gram model when considering the word *EEG* from the context *Digital video EEG is performed (*from report (c) in Figure 1). The goal of the skip-gram model is to learn the projection matrix $\boldsymbol{U}$ which, when multiplied with the one-hot vector for *EEG,* is best able to predict the one-hot vectors associated with each context word, e.g., *Digital, video, is,* and *performed*. In this way, the skip-gram model is able to learn a representation for the word *EEG* which captures the facts that (1) an EEG can be *performed* and that (2) *digital video* is a type of

EEG. We learn the optimal project matrix $\boldsymbol{U}$ by training a separate neural network in which the input is every word $\boldsymbol{R}_i \in \boldsymbol{R}$ in every report $\boldsymbol{R} \in \mathcal{X}$, and the goal is to predict the n previous and n following words using the projection matrix $\boldsymbol{U}$:

$$\boldsymbol{U} = \max_{\boldsymbol{U}'} \sum_{\boldsymbol{R} \in \mathcal{X}} \sum_{i=1}^{N} \left[\sum_{t=-n}^{-1} P(\boldsymbol{R}_{i+t} | \boldsymbol{R}_i; \boldsymbol{U}') + \sum_{t=1}^{t=n} P(\boldsymbol{R}_{i+t} | \boldsymbol{R}_i; \boldsymbol{U}') \right] \quad (3)$$

where

$$P(\boldsymbol{R}_{i+t} | \boldsymbol{R}_i; \boldsymbol{U}') = \frac{\exp(\boldsymbol{R}_{i+t} \boldsymbol{U}' \cdot \boldsymbol{R}_i \boldsymbol{U}')}{\sum_{v=1}^{V} \exp(\boldsymbol{R}_i \boldsymbol{U}'_v)} \quad (4)$$

In our experiments, we used $n = 2$. Learning the optimal projection matrix $\boldsymbol{U}$ allows the model to produce a high-level feature representation of every word in the report, $\boldsymbol{W} \in \mathbb{R}^{N \times K}$, by simply multiplying $\boldsymbol{R}$ with $\boldsymbol{U}$:

$$\boldsymbol{W} = \boldsymbol{R}\boldsymbol{U} \quad (5)$$

where each $\boldsymbol{W}_i$ indicates the word-level feature vector associated with $\boldsymbol{R}_i$. The word-level feature vectors ($\boldsymbol{W}$) learned by the skip-gram model have a number of useful algebraic properties. Of particular note is their ability to capture semantic similarity, for example, closest feature vector to the word *generalized* is that of the word *diffuse*, and the closest feature vector to *focal* is that of the word *localized*. This highlights the ability of the skip-gram model to capture the fact that both *generalized* and *diffuse* refer to activity spread across a large area of the brain (e.g. both hemispheres, multiple lobes), while *focal* and *localized* describe activity concentrated in one or two regions of the brain.

Learning EEG Report-Level Features

Representing each word in a report as an independent feature vector is not sufficient to predict the overall impression. Instead, it is necessary to learn how to combine and compose the word-level feature vectors $\boldsymbol{W}$ to create a single high-level feature vector for the report, $\boldsymbol{e}$. We considered two different neural architectures for learning $\boldsymbol{e}$. The first model is based on a Deep Averaging Network (DAN), while the second uses a Recurrent Neural Network (RNN). Both architectures enable the model to learn a semantic composition but in different ways. Specifically, a DAN learns an un-ordered composition of each word in the document, while an RNN learns an ordered composition. However, the representation learned by an RNN often struggles to account for long-distance interactions and favors the latter half of each document. Consequently, we evaluated both models in order to determine the most effective architecture for learning report-level features from EEG reports.

Deep Averaging Network for Inferring Underspecified Information. The Deep Averaging Network (DAN) learns the report-level feature representation $\boldsymbol{e}$ of a report based on its word-level features $\boldsymbol{W}$. To understand the need for report-level features, consider the excerpt:

Ex_1: *...a well-formed 9 to 10Hz posterior dominant rhythm, which is symmetrically reactive to eye opening and eye closing.*

Interpreting Ex_1 requires understanding (1) that the words *posterior dominant rhythm* describe a single EEG activity, and (2) that the *posterior dominant rhythm* is *well-formed*. Clearly, word-level features are not sufficient to capture this information. Instead we would like to extract high-level semantic features encoding information across words, sentences, and even sections of the report. The DAN used in our model accomplishes these goals using five layers, as shown in Figure 3. The first two layers learn an encoding of each word $\boldsymbol{W}_i$ associated with the report, and the third layer combines the resulting encodings to produce an encoding for the report itself. The final two layers refine this encoding to produce $\boldsymbol{e}$. To learn an encoding for each word, we apply two densely-connected Rectified Linear Units (ReLUs). The rectifying activation functions used in ReLUs have several notable advantages, in particular the ability to allow for sparse activation. This enables learning which words in an EEG report have the largest impact the over-all impression. By using a ReLU for the first layer of our encoder, each word represented by feature vector $\boldsymbol{W}_i$ is projected onto an initial encoding vector $\boldsymbol{r}_i^{(1)}$. The ReLU used in the second layer of the encoder produces a more expressive[14] encoding $\boldsymbol{r}_i^{(2)}$. Both encodings are generated as:

$$\boldsymbol{r}_i^{(1)} = \max(\boldsymbol{S}_1 \cdot \boldsymbol{R}_i + \boldsymbol{b}_1, 0) \tag{6}$$

$$\boldsymbol{r}_i^{(2)} = \max\left(\boldsymbol{S}_2 \cdot \boldsymbol{r}_i^1 + \boldsymbol{b}_2, 0\right) \tag{7}$$

where $\boldsymbol{S}_1, \boldsymbol{S}_2 \in \theta$ are the learned weights of the connections between the neurons in layers 1 and 2, and $\boldsymbol{b}_1, \boldsymbol{b}_2 \in \theta$ are bias vectors. While the encoding $\boldsymbol{r}_i^{(2)}$ represents information obtained from each word vector $\boldsymbol{W}_i \in \boldsymbol{W}$, we are interested in producing a single representation that captures the information about the entire EEG report. This is accomplished by layers 3 through 5. In layer 3, the piece-wise average of all word vector encodings is produced:

$$\boldsymbol{a} = \frac{1}{N}\sum_{i=0}^{N} \boldsymbol{r}_i^{(2)} \tag{8}$$

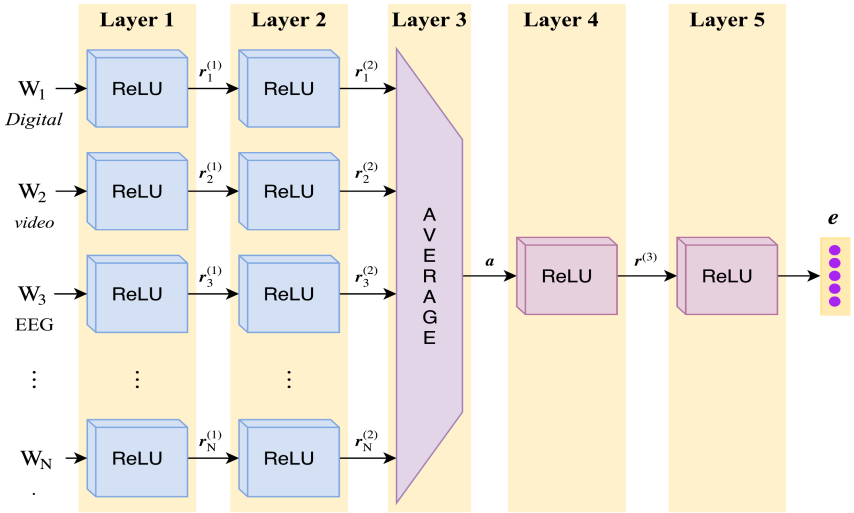

Figure 3. Architecture of the Deep Averaging Network (DAN) used to combine and compose word-level features $\boldsymbol{W}_1 \cdots \boldsymbol{W}_N$ extracted from the EEG Report shown in Figure 1(c).

Layers 4 and 5 act as additional "deep" layers which enhance the quality of the encoding. To implement layers 4 and 5 we used two additional ReLUs:

$$\boldsymbol{r}^{(3)} = \max(\boldsymbol{S}_3 \cdot \boldsymbol{a} + \boldsymbol{b}_3, 0) \quad (9)$$

$$\boldsymbol{e} = \max\left(\boldsymbol{S}_e \cdot \boldsymbol{r}^{(3)} + \boldsymbol{b}_e, 0\right) \quad (10)$$

where $\boldsymbol{S}_3, \boldsymbol{b}_3, \boldsymbol{S}_e, \boldsymbol{b}_e \in \theta$ are the learned weights and biases used by each ReLU layer. Equations 6 through 10 enable our model to generate a fixed-length high-level vector, $\boldsymbol{e}$, which encodes semantic information about the entire EEG report.

Recurrent Neural Network for Inferring Underspecified Information. In contrast to the DAN, the recurrent neural network (RNN) used in our model jointly learn how to (1) map a sequence of word-feature vectors $(\boldsymbol{W}_1, \cdots \boldsymbol{W}_N)$ to a sequence of hidden memory states $(\boldsymbol{m}_1, \cdots \boldsymbol{m}_N)$ as well as to (2) map the hidden memory states to a sequence of output vectors $(\boldsymbol{y}_1, \cdots, \boldsymbol{y}_N)$, as illustrates in Figure 4. Formally, for each word $i \in [1, N]$ where N is the length of the EEG report:

$$\boldsymbol{m}_i = \sigma(\boldsymbol{S}_m \cdot [\boldsymbol{W}_i + \boldsymbol{m}_{i-1}]) \quad (11)$$

$$\boldsymbol{y}_i = \sigma\left(\boldsymbol{S}_y \cdot \boldsymbol{m}_i\right) \quad (12)$$

where $\boldsymbol{S}_m, \boldsymbol{S}_y \in \theta$ are the learned weights connecting the neurons in each layer. Unfortunately, RNNs are known to have difficulties learning long-range dependencies between words. For example, consider the excerpt:

Ex$_2$: *periodic delta with associated periodic paroxysmal fast activity identified from the left hemisphere with a generous field of spread including the centrotemporal and frontocentral region*.

A standard RNN would be unlikely to infer that the *periodic delta* activity was observed in the *centrotemporal* and *frontocentral* regions of the brain due to the significant number of words between them. In order to enable our RNLM to overcome this barrier, we implement each of our RNNs as a stacked series of long short-term memory units (LSTMs) which are able to learn long-range dependencies by accumulating an internal memory.

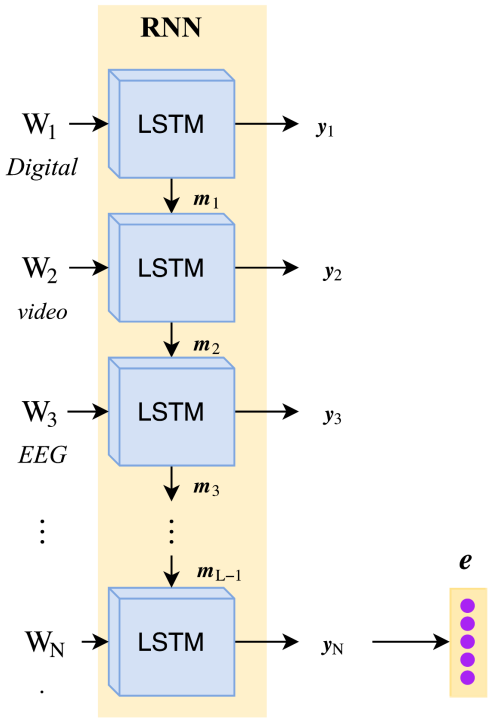

Figure 4. Architecture of the Recurrent Neural Network (RNN) used to combine and compose word-level features $W_1 \cdots W_N$ extracted from an EEG Report, shown on report (c) from Figure 1.

Inferring the Over-all Impression from EEG Reports

The learned high-level feature vector $\boldsymbol{e}$ is used to determine the most likely over-all impression associated with the EEG report. Given $\boldsymbol{e}$, we

approximated the likelihood of assigning the over-all impression c to the EEG report associated with $\boldsymbol{e}$, i.e. $P(c \mid \boldsymbol{e};\ \theta)$, with a densely connected logistic sigmoid layer. The sigmoid layer computes a floating point number $\tilde{c} \in [0, 1]$ such that $\tilde{c} \leq 0.5$ if $c = \text{NORMAL}$, and $\tilde{c} > 0.5$ if $c = \text{ABNORMAL}$:

$$\tilde{c} = \sigma(\boldsymbol{S}_c \cdot \boldsymbol{e} + \boldsymbol{b}_c) \tag{13}$$

where $\boldsymbol{S}_c, \boldsymbol{b}_c \in \theta$ are the learned weights and bias vector for the sigmoid layer, and σ is the standard logistic sigmoid function, $\sigma(x) = {e^x}/{e^{x+1}}$. Equation 19 allows us to approximate the likelihood of the over-all impression $c \in C$ being assigned to the report associated with $\boldsymbol{e}$ as:

$$P(c \mid \boldsymbol{e}; \theta) = \begin{cases} 1 - \tilde{c}, & \text{if } c = \text{NORMAL} \\ \tilde{c}, & \text{if } c = \text{ABNORMAL} \end{cases} \tag{14}$$

Training the Model with EEG Reports

We trained our model by learning the parameters θ which minimize the loss when computing the over-all impression c for each report $\boldsymbol{R}$ in the training set $\mathcal{X}$. In our experiments, we used the *cross-entropy loss* between the predicted over-all impression c and the gold-standard value $\hat{c}$ indicated by the neurologist (in the removed *impression* section). Formally:

$$\mathcal{L}(\theta) \propto \sum_{(\boldsymbol{R}, \hat{c}) \in \mathcal{X}} [P(c \mid \boldsymbol{e}; \theta) \cdot P(\boldsymbol{e} \mid \boldsymbol{W}) \cdot P(\boldsymbol{W} \mid \boldsymbol{R})] \cdot \log P(\hat{c}) \tag{15}$$

where $P(\hat{c}) = 1$ if $c = \text{ABNORMAL}$, and zero otherwise. We trained our model using adaptive moment estimation (ADAM).

Inferring Underspecified Information from EEG Reports

The optimal over-all impression c for a new EEG report $\boldsymbol{R}$ can be determined in three steps: (1) transform $\boldsymbol{R}$ into a word-level feature matrix, $\boldsymbol{W} = \boldsymbol{U}\boldsymbol{R}$, using the projection matrix $\boldsymbol{U}$ learned from the training data; (2) transform the word-level feature matrix $\boldsymbol{W}$ into a single report-level feature vector $\boldsymbol{e}$ using either the DAN or the RNN; and (3) determine the over-all impression c from the report-level feature vector $\boldsymbol{e}$.

Implementation Details

In our experiments, we implemented our model using Tensorflow (version 0.8). Because ADAM tunes the learning rate as it trains, we initialized ADAM using the default parameters in Tensorflow (learning rate $= 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = e^{-8}$). For the purposes of our experiments, gradient clipping was not applied, and no regularization terms were added. Model parameters were determined using a grid-search as follows: skip-gram, ReLU. and LSTM dimensionality were chosen from $\{100,\ 200,\ 500,\ 1000\}$. When performing grid search, we constrained all ReLUs to share the same dimensionality. We found the optimal dimensionality for the skip-gram embeddings, ReLU layers, and LSTM to each be 200 dimensions/units.

For Aim 2 of the project, the UTD team has also developed a novel Deep Section Recovery Model (DSRM) which applies deep neural learning on a large body of EEG reports in order to infer the

expected clinical correlations for a patient based solely on the natural language content in his or her EEG report. The paper that reported on the DSRM approach received the **Homer Warner award** at the 2017 AMIA Symposium. The DSRM was trained and evaluated using the Temple University Hospital (TUH) EEG Corpus[5] by (a) identifying and removing the clinical correlation section written by the neurologist and (b) training the DSRM to infer the entire clinical correlation section from the remainder of the report. At a high level, the DSRM can be viewed as operating through two general steps:

Step 1: word- and report- level features are automatically extracted from each EEG report to capture contextual, semantic, and background knowledge; and

Step 2: the most likely clinical correlation section is jointly (a) inferred and (b) expressed through automatically generated natural language.

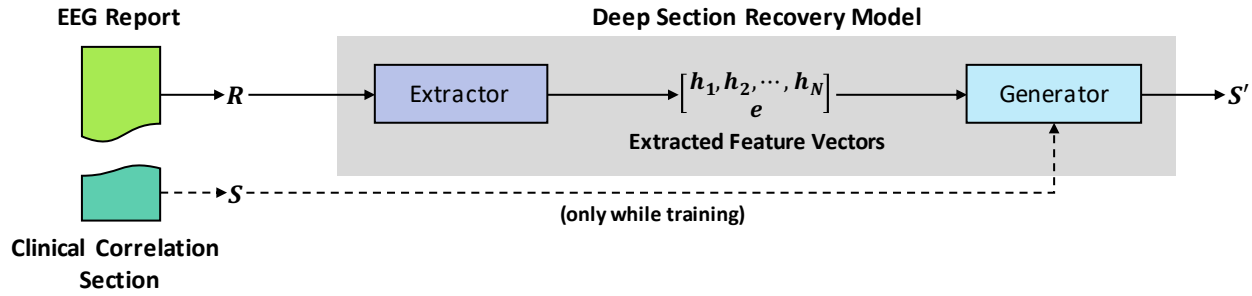

Figure 5: Simplified Architecture of the Deep Section Recovery Model (DSRM).

Inferring the Clinical Correlation Section

When writing the clinical correlation section of an EEG report, the neurologist considers the information described in the previous sections, such as relevant clinical history or notable epileptiform activities, as well as their accumulated medical knowledge and experience with interpreting EEGs. This type of background knowledge is difficult to capture with hand-crafted features because it is rarely explicitly stated; rather, it is implied through the subtlety, context, and nuance afforded the neurologist by natural language. Consequently, to approach this problem, we present a deep neural network architecture which we refer to as the Deep Section Recovery Model (DSRM). Illustrated in Figure 5, the DSRM consists of two major components:

- the *Extractor* which learns how to automatically extract (a) feature vectors representing contextual and background knowledge associated with each word in a given EEG report as well as (b) a feature vector encoding semantic, background, and domain knowledge about the entire report; and
- the *Generator* which learns how to use the feature vectors extracted by the Extractor to produce the most likely clinical correlation section for the given report while also considering the semantics of the natural language it is generating.

In order to train and evaluate the DSRM, we identified all EEG reports in the TUH EEG Corpus which contained a *CLINICAL CORRELATION* section and *removed* that section from the report. The model was trained to recover the missing clinical correlation section in the training set and evaluated based on the clinical correlation sections it inferred for reports in the test set. In the remainder of this section, we describe (1) the natural language pre-processing steps applied to the data, (2) the mathematical problem formulation, (3) the Extractor, (4) the Generator, (5) how the parameters of the model are learned from the training set, and (6) how the learned parameters are used to infer the most likely clinical correlation section for a (new) EEG report.

Natural Language Pre-processing

Before applying the Deep Section Recovery Model, we pre-processed each EEG report with three basic natural language processing steps: (1) sentence boundaries were identified using the OpenNLP sentence splitter; (2) word boundaries were detected using the GENIA tokenizer, and (3) section boundaries were identified using a simple regular expression search for capitalized characters ending in a colon. These three pre-processing steps allowed us to represent each section of an EEG report as a sequence of words in which the symbols $\langle s \rangle$ and $\langle /s \rangle$ were used to indicate the start and end of each sentence, $\langle p \rangle$ and $\langle /p \rangle$ were used to indicate the start and end of each section, and $\langle d \rangle$ and $\langle /d \rangle$ were used to indicate the start and end of each report.

Problem Formulation

In order to formally define the problem, it is necessary to first define the vocabulary as the set of all words observed at least once in any section (including the clinical correlation section) of any EEG report in the training set. Let V indicate the size or number of words in the vocabulary. This allows us to represent an EEG report as sequence of V -length one-hot vectors corresponding to each word in the report, i.e., $R \in \{0,1\}^{N \times V}$ where N is the length or number of words in the report. Likewise, we also represent a clinical correlation section as a sequence of V -length one-hot vectors; in this case, $S \in \{0,1\}^{M \times V}$ where M is the number of words in the clinical correlation section. The goal of the Deep Section Recovery Model is to infer the most likely clinical correlation section for a given EEG report. Let θ be the learn-able parameters of the model. Training the model equates to finding the values of θ which assign the highest probabilities to the gold-standard (neurologist-written) clinical correlation sections for each EEG report in the training set; formally:

$$\boldsymbol{\theta} = \underset{\boldsymbol{\theta'}}{argmax} \Pr(\boldsymbol{S}|\boldsymbol{R};\boldsymbol{\theta'}) \tag{16}$$

We decomposed the probability of a particular clinical correlation section being produced for a given EEG report (i.e., correctly identifying and describing the clinical correlations in the report) into two factors:

$$Pr(\boldsymbol{S}|\boldsymbol{R};\boldsymbol{\theta}) \approx \overbrace{Pr(\boldsymbol{e},\boldsymbol{h_1},\cdots,\boldsymbol{h_N}|\boldsymbol{R};\boldsymbol{\theta})}^{Extractor} \cdot \overbrace{Pr(\boldsymbol{S}|\boldsymbol{e},\boldsymbol{h_1},\cdots,\boldsymbol{h_N};\boldsymbol{\theta})}^{Generator} \tag{17}$$

where the first factor is implemented by the Extractor and the second factor is implemented by the Generator.

The Extractor

The language in the clinical correlation section is intended to relate findings and observations described in the previous sections of the record to the over-all clinical picture of the patient. Consequently, in order to automatically produce the clinical correlation section, the goal of the Extractor is to automatically (1) identify important neurological findings and observations (e.g., "background slowing"); (2) identify descriptions of the patient's clinical picture (e.g., "previous seizure"); and (3) determine the inferred relationship(s) between each finding and the clinical picture as described by the EEG report or implied by medical knowledge and experience (e.g., "observed epileptiform activity is consistent with head trauma"). It should be noted that the length and content of each EEG report varies significantly throughout the collection, both in terms of the sections included in each report as well as the content in each section. Moreover, when producing an EEG report, each neurologist writes in a different style, ranging between terse 12-word sections to 600-word sections organized into multiple paragraphs. Consequently, the role of the Extractor is to overcome these barriers and extract meaningful feature vectors which characterize semantic, contextual, and domain knowledge. To address these requirements, we implemented the Extractor using the deep neural architecture illustrated in Figure 6. The Encoder relies on five neural layers to produce feature vectors for each word in the report ($h_1,\cdots,h_N$) as well as a feature vector characterizing the entire report (e):

Layer 1: Embedding. The role of the embedding layer is to embed each word in the EEG report $\boldsymbol{R}_i$ (represented as a V-length 1-hot vector) into a K-length continuous vector $\boldsymbol{r}_i^{(1)}$(where $K \ll V$). This is accomplished by using a fully connected linear projection layer, $\boldsymbol{r}_i^{(1)} = \boldsymbol{R}_i\boldsymbol{W}_e + \boldsymbol{b}_e$, where ($\boldsymbol{W}_e \in \mathbb{R}^{\mathrm{V}\mathrm{x}K}, \boldsymbol{b}_e \in \mathbb{R}^{Vx1}$) $\in \boldsymbol{\theta}$ correspond to the vocabulary projection matrix and bias vector learned by the Extractor.

Layer 2: Bidirectional Recurrent Neural Network. Layer 2 implements a bidirectional recurrent neural network (RNN) using two parallel RNNs trained on the same inputs: (1) a forward RNN which processes words in the EEG report in left-to-right order and (2) a backward RNN which processes words in the EEG report in right-to-left order. This allows the forward RNN to extract features capturing any short- or long-range contextual information about each word in R provided by any preceding words in the EEG report (e.g. that "slowing" is negated in "no background slowing"). Likewise, the backward RNN extracts features capturing any short- or long-

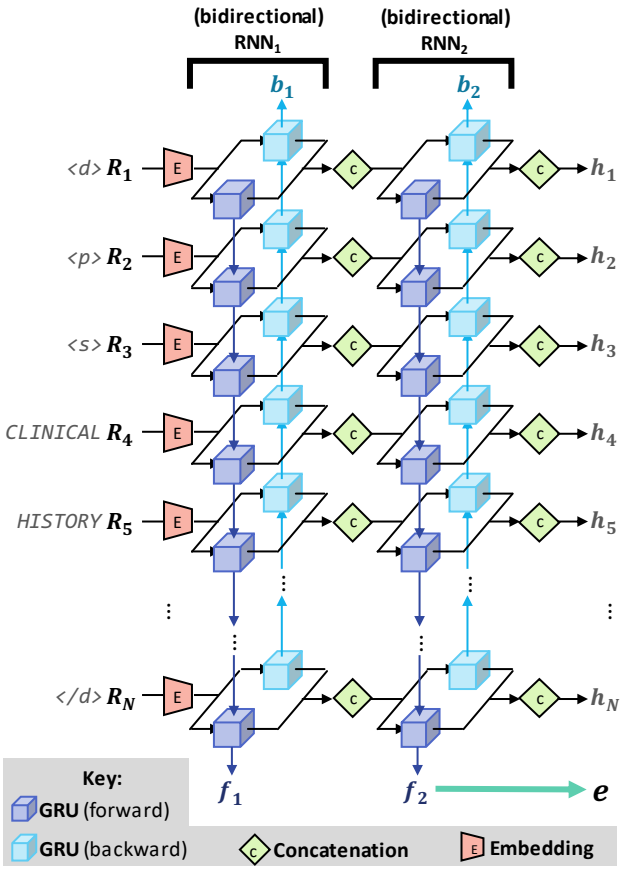

Figure 6: Detailed Architecture of the Extractor

range contextual information provided by successive words in the EEG report (e.g. that "hyperventilation" described in the introduction section may influence the inclusion of "spike and wave discharges" in the EEG impression or description sections). Formally, the forward RNN maps the series word embeddings $\boldsymbol{r}_1^{(1)}, \ldots, \boldsymbol{r}_N^{(1)}$ to a series of "forward" word-level feature vectors $\boldsymbol{r}_1^{(2f)}, \ldots, \boldsymbol{r}_N^{(2f)}$, while the backward RNN maps $\boldsymbol{r}_1^{(1)}, \ldots, \boldsymbol{r}_N^{(1)}$ to a series of "backward" word-level feature vectors $\boldsymbol{r}_1^{(2b)}, \ldots, \boldsymbol{r}_N^{(2b)}$. In our model, the forward and backward RNNs were implemented as a series of shared Gated Recurrence Units (GRUs).

Layer 3: Concatenation. The concatenation layer combines the forward and backward word-level feature vectors to produce a single feature vector for each word, namely, $\boldsymbol{r}_N^{(3)} = \left[\boldsymbol{r}_N^{(2f)}; \boldsymbol{r}_N^{(2b)}\right]$ where $[\boldsymbol{x};\boldsymbol{y}]$ indicates the concatenation of vectors x and y

Layer 4: 2nd Bidirectional Recurrent Neural Network. In order to allow the model to extract more expressive features, we use a second bidirectional RNN layer. This layer operates identically to the bidirectional RNN in Layer 2, except that the word-level feature vectors produced in Layer 3, i.e., $\boldsymbol{r}_1^{(3)}, \ldots, \boldsymbol{r}_N^{(3)}$, are used as the input to the bidirectional RNN (instead of $\boldsymbol{r}_1^{(1)}, \ldots, \boldsymbol{r}_N^{(1)}$ used in Layer 2). Likewise, the memory states produced in Layer 4 are denoted as $\boldsymbol{f_2}$ and $\boldsymbol{b_2}$, corresponding to the forward RNN and the backward RNN, respectively. Unlike the bidirectional RNN used in Layer 2, we use the final memory of the forward RNN (i.e. $\boldsymbol{f_2}$) as the report-level feature vector e which will be used by the Generator.

Layer 5: 2nd Concatenation. As in Layer 3, the second concatenation layer combines the forward and backward word-level features vectors produced in the previous layer. In the case of Layer 5, however, we used the resulting feature vectors $\boldsymbol{h_1}, \ldots, \boldsymbol{h_N}$ as the word-level feature vectors which will be provided to the Generator.

The Generator

The role of the Generator is to generate the most likely clinical correlation section for a given EEG report using the feature vectors extracted by the Extractor. It is important to note that because the clinical correlation sections vary both in terms of their length and content, the number of possible clinical correlations sections that could be produced is intractably high (V^{MMAX} where M_{MAX} is the maximum length of a clinical correlation section). Consequently, we substantially reduce the complexity of the problem by modeling the assumption that each word in the clinical correlation section can be determined based solely on (1) the word-level feature vectors $h_1, \cdots, h_N$ extracted by the Extractor, (2) the report-level feature vector e extracted by the Extractor, and (3) any preceding words produced by the Generator. This assumption allows us to define the probability of any clinical correlation section, S^0, having been produced by a neurologist for a given EEG report (i.e., the second factor in Equation 2) as:

$$\Pr(\boldsymbol{S'}|\boldsymbol{R}) = \prod_{j=1}^{M} \Pr\left(\boldsymbol{S'_j}|\boldsymbol{S'_{j-1}}, \ldots, \boldsymbol{S'_1}, \boldsymbol{e}, \boldsymbol{h_1}, \ldots, \boldsymbol{h_N}; \boldsymbol{\theta}\right) \quad (18)$$

To compute Equation 18, we designed the Generator to act as a type of Recurrent Neural Language Model (RNLM) which incorporates a Recurrent Neural Network (RNN) to produce one word in

the clinical correlation section at-a-time while maintaining and updating an internal memory of which words have already been produced.

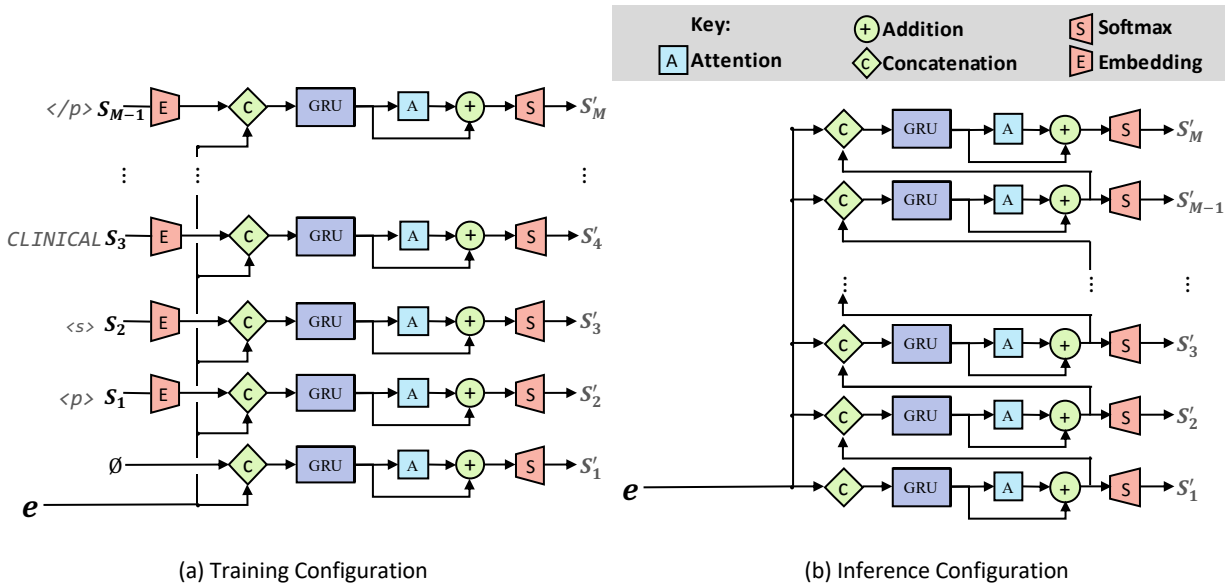

Figure 7: Detailed Architecture of the Generator under (a) Training and (b) Inference Configurations.

To improve training efficiency, the Generator has two similar but distinct configurations: one for training, and one for inference (e.g. testing). Figure 7 illustrates the architecture of the Generator under both configurations. The primary difference between each configuration is the input to the RNN: when training, the model embeds the previous word from the *gold-standard* clinical correlation section (e.g. S'_{j-1}) to predict S'_j while during inference the RNN operates on the embedding of the previously *generated* word (e.g. S'_{j-1}) to predict S'_j. The Generator produces the natural language content of a clinical correlation section for a given EEG report using four layers (with the preliminary embedding layer in the training configuration acting as an extra "zero"-th layer):

- **Layer 0: Embedding.** The embedding layer, which is only used when the Generator is in training configuration, embeds each word in the gold-standard clinical correlation section $\boldsymbol{s}_j$ (represented by V-length 1-hot vectors) into an L-length continuous vector space, $\boldsymbol{s}_j^{(0)}$, where $L \ll V$. This is accomplished by using a fully connected linear projection layer, $\boldsymbol{s}_j^{(0)} = \boldsymbol{S}_j \boldsymbol{W}_G + \boldsymbol{b}_G$ where $(\boldsymbol{W}_G \in \mathbb{R}^{VxL}, \boldsymbol{b}_G \in \mathbb{R}^{Vx1}) \in \boldsymbol{\theta}$ correspond to the vocabulary projection matrix and vocabulary bias vector learned by the Generator.

- **Layer 1: Concatenation.** The first layer used in both configurations of the Generator is a concatenation layer which combines the embedded representation of the previous word with e, the report-level feature vector extracted by the Extractor, $\boldsymbol{s}_j^{(1)} = \left[\boldsymbol{s}_{j-1}^{(0)}; \boldsymbol{e}\right]$ where $[x, y]$ indicates the concatenation of vectors x and y and $\boldsymbol{s}_0^{(0)}$ is defined as a zero vector.

- **Layer 2: Gated Recurrent Unit.** The second layer used by both configurations is a Gated Recurrent Unit (GRU). The GRU allows the model to accumulate memories encoding long-distance relationships between each produced word of the clinical correlation section, $\boldsymbol{S'}$, and any words previously produced by the model. This is performed by updating and maintaining an internal memory within the GRU which is shared across all words in the clinical correlation section. We denote the output of the GRU as $\boldsymbol{s}_j^{(2)}$.

- **Layer 3: Attention.** In order to improve the quality and coherence of natural language produced by the Generator, an attention mechanism was introduced. The attention mechanism allows the Generator to consider all of the world-level feature vectors $\boldsymbol{h}_1, \ldots, \boldsymbol{h}_N$ produced by the Extractor for the given report, and learns the degree that each word in the EEG report influences the selection of (or *aligns* with) $\boldsymbol{S}'_j$; formally:

$$s_j^{(3)} = \sum_{i=1}^{N} \alpha_{ij} h_i \qquad \alpha_{ij} = \frac{\exp(\beta_{ij})}{\prod_{l=1}^{N} \exp(\beta_{lk})} \qquad \beta_{ij} = \sigma(W_\beta \boldsymbol{s}_j^{(2)} + U_\beta \boldsymbol{h}_i + \boldsymbol{b}_\beta)$$

  such that $\boldsymbol{\alpha}_{i,j}$ is an alignment vector used in the alignment model β_{ij} which determines the degree that the i^{th} word in the EEG report $\boldsymbol{R}$ (represented by $\boldsymbol{h}_i$) influences the j^{th} word of the clinical correlation section $\boldsymbol{S}'_j$ (represented by $\boldsymbol{s}_j^{(2)}$).

- **Layer 4: Addition.** The role of the fourth layer is to combine the result of the previous attention layer with the result of the GRU in Layer 2, i.e., $\boldsymbol{s}_i^{(4)} = \boldsymbol{s}_i^{(3)} + \boldsymbol{s}_i^{(2)}$

- **Layer 5: Softmax Projection.** In order to measure the probability of each word $\boldsymbol{S}'_j$ being produced for the given EEG report, we use a final softmax projection layer to produce a vocabulary-length vector $s_j^{(5)}$ in which the v^{th} element indicates the probability that $\boldsymbol{S}'_j$ should be generated as the v^{th} word in the vocabulary, $\boldsymbol{s}_i^{(5)} = softmax(\boldsymbol{s}_i^{(4)} \boldsymbol{W}_p + \boldsymbol{b}_p)$ where $softmax(x) = \frac{\exp(x)}{\sum_{v=1}^{V} \exp(x_v)}$, and $v \in [1, V]$. This allows us to complete the definition of Equation 3:

$$\Pr\left(\boldsymbol{S}'_j = v \middle| \boldsymbol{S}'_j, \ldots, \boldsymbol{S}'_1, \boldsymbol{e}, \boldsymbol{h}_1, \ldots, \boldsymbol{h}_N; \theta\right) = \boldsymbol{s}_{jv}^{(5)} \tag{19}$$

Training the Deep Section Recovery Model

Training the Deep Section Recovery Model (DSRM) is achieved by finding the parameters $\boldsymbol{\theta}$ which are most likely to produce the gold-standard clinical correlation sections for each EEG report in the training set T. Formally, we model this by minimizing the cross-entropy loss between the vocabulary-length probability vectors produced by the model ($\boldsymbol{s}_j^{(5)}$) and the one-hot vectors corresponding to each word in the gold-standard clinical correlation section ($\boldsymbol{S}_j$).

$$\mathrm{L}(\theta) \propto \sum_{(\boldsymbol{R},\boldsymbol{S}) \in T} \left[\sum_{j=1}^{M} [\boldsymbol{s}_j^{(5)} log \boldsymbol{S}_j + (1 - \boldsymbol{s}_j^{(5)}) \log (1 - \boldsymbol{s}_j^{(5)})] \right] \tag{20}$$

The model was trained using Adaptive Moment Estimation (ADAM) (with an initial learning rate η=0.001).

Inferring Clinical Correlations

Given $\boldsymbol{\theta}$ learned from the training set, the clinical correlation section $\boldsymbol{S}$ can be generated for a new EEG report $\boldsymbol{R}$ using the inference configuration illustrated in Figure 3b. In contrast to the training configuration in which $\boldsymbol{S}'_j$ is selected using the previous word from the gold-standard clinical correlation section ($\boldsymbol{S}_{j-1}$), during inference, the model predicts $\boldsymbol{S}'_j$ using the word previously produced by the model ($\boldsymbol{S}'_{j-1}$). It is important to note that, unlike training, we do not know the length of the clinical correlation section we will generate. Consequently, the model continually generates output until it produces the END-OF-SECTION symbol $\langle/p\rangle$. Thus, the length of the inferred clinical correlation section *M* is determined dynamically by the model. When inferring the most likely clinical correlation section, it is necessary to the convert the vocabulary probability vectors $\boldsymbol{s}_1^{(5)}, \ldots, \boldsymbol{s}_M^{(5)}$ to one-hot vocabulary vectors $\boldsymbol{S}'_j$ that can be directly mapped to natural language.[1]

3. Retrieve Patient Cohorts From the EMRs That Document Their Hospital Visits

In the third year of the project, the team from the University of Texas at Dallas has enhanced the Multi-Modal EEG Patient Cohort Retrieval system called *MERCuRY* (and acronym for Multi-modal EncephalogRam patient Cohort discoverY), by incorporating a learning-to-rank methodology.

In the third year of the project, the team from the University of Texas at Dallas has enhanced the MERCuRY system with learning-to-rank capabilities. Ranking of the patients in the cohort was essential in the usability studies performed with the MERCuRY system, as it enabled neurologist researchers to rapidly identify effective interventions for epilepsy accompanied by mental health comorbidities. However, not all the patients from the cohorts discovered by MERCuRY were relevant to the cohort criteria. Relevance judgements produced by neurologists indicated limitations of the system, but also provided important lessons that can be used for learning how to rank patients. Inspired by this observation, we designed a *learning patient cohort retrieval* (**L-PCR**) system using the publicly-available collection of electroencephalography (EEG) reports from the Temple University Hospital (TUH) EEG Corpus. Patient cohorts were recognized from the TUH EEG Corpus based on descriptions provided by practicing neurologists. Specifically, we trained and evaluated the L-PCR system using 30 cohort descriptions generated by four practicing neurologists.

Unlike traditional patient cohort retrieval systems, such as MERCuRY, the L-PCR system uses a *learning-to-rank* approach for identifying patient cohorts that takes advantage of physician feedback. The learning-to-rank paradigm allows the L-PCR system to consider *relevance judgments* performed by clinicians to *learn* an improved patient relevance model used for retrieving and ranking patients for any given cohort descriptions.

The L-PCR system illustrated in Figure 8 includes five main components:

[1] Let $\hat{\boldsymbol{s}}_{\boldsymbol{j}} = argmax(\boldsymbol{s}_j^{(5)})$; $\boldsymbol{S}'_j$ is defined as the one-hot vector in which the $\hat{\boldsymbol{s}}_j^{th}$ value is 1 and all other values are zero.

- a **query processing** component processes a given cohort description d_i to produce a machine-readable query, $q_i^{c,e}$;
- an **EHR processing** component produces an index of the narratives from the EHR collection;
- a **visit retrieval** component retrieves a sub-set of "candidate" visits from the EHR collection, $[v_1, \cdots, v_M]$, to be ranked by the learning relevance model;
- a **feature extraction** component extracts features vectors $[\boldsymbol{x}_1^i, \cdots, \boldsymbol{x}_M^i]$ corresponding to each candidate visit the relationship between the visit and the cohort description; and
- the **learning relevance model** uses a Random Forest (RF) classifier to infer the *relevance scores* $[s_1^i, \cdots, s_M^i]$ for each candidate visit $[v_1^i, \cdots, v_M^i]$ based on their associated feature vectors $[\boldsymbol{x}_1^i, \cdots, \boldsymbol{x}_N^i]$; the RF is trained using the relevance judgments $[y_1^i, \cdots, y_M^i]$ provided by physicians.

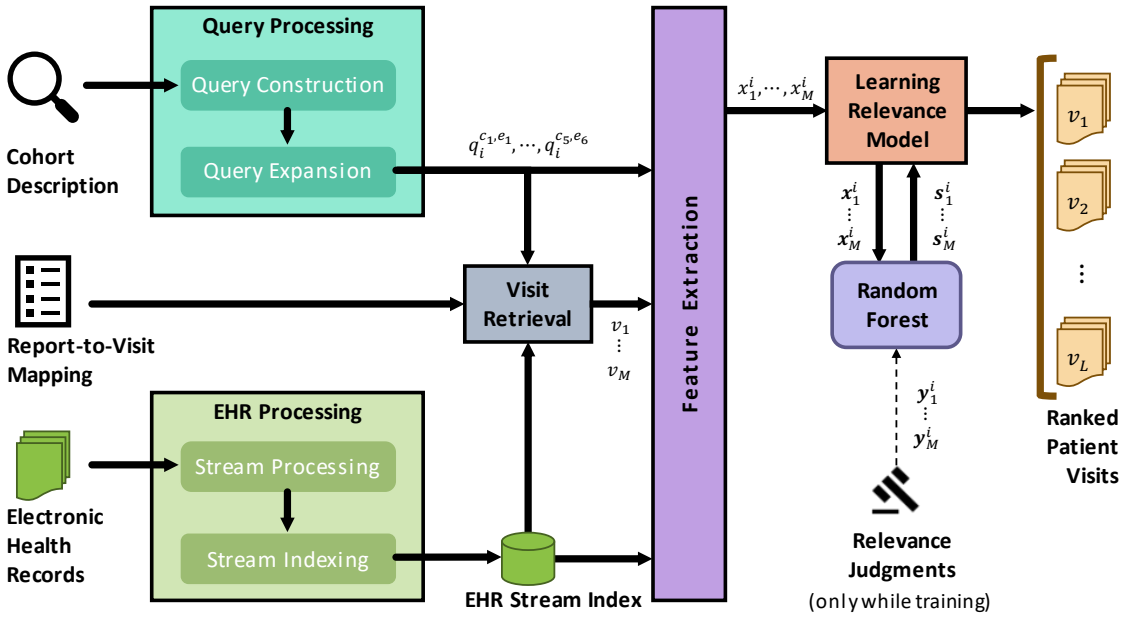

Figure 8: Architecture of the learning patient cohort retrieval (L-PCR) system.

While the query processing was already developed for the MERCuRY in the previous years of the project, novel methods had tp be developed for the Electronic Health Record (HER) processing used by the L-PCR system.

Electronic Health Record Processing

Stream Processing. We unified indexing, searching, and feature extraction across the TUH EHR collection, by representing the EHR as a set of multiple, abstract *streams* of unstructured information. Each stream corresponds to one or more sections in the EHR collection. Conceptually, each stream acts as a "lens" that determines which sections of the EHR are considered during feature extraction and retrieval. The stream representation allows the L-PCR system to automatically account for the semantics of each stream, without the semantics being

explicitly encoded. Figure 9 illustrates the streams used for each EHR collection available from the Temple University Hospital in the form of a big data EEG data. Processing streams of narratives from the EEG reports is important, because identifying patient relevant to a cohort description needs to take into account that a patient may have multiple *hospital visits*, which need to be assembled to determine the relevance of the patient to the cohort description.

Stream Indexing. To expedite feature extraction from the EHRs associated with each hospital visit, we separately indexed the content of each EHR collection using Apache Lucene. We used a *tiered* indexing approach in which each stream was indexed independently, allowing individual streams of each EHR to be retrieved during feature extraction and retrieval. No pre-processing was applied beyond tokenization with Lucene's English Analyzer.

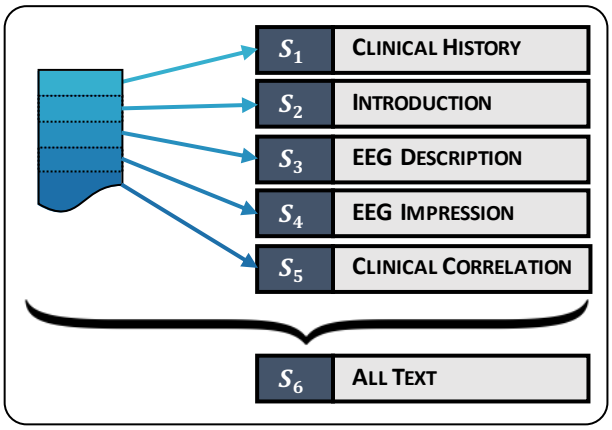

Figure 9. Indexed Streams from EEG Reports (left) and Hospital Records (right).

Visit Retrieval

To reduce complexity and improve scalability of the L-PCR system, rather than extracting features from every EHR in the collection, we rely on a basic retrieval step to identify a high-recall set of "candidate'' visits likely to be relevant to the cohort description. These candidate visits are obtained by constructing a query with Bag-of-Words (C_1), expanding by All Expansions (E_5), and identifying the top M ranked EHRs by the All Text stream (S_4/S_5) with the BM25 ranking function (in our experiments we used $M = 2{,}000$). This allowed the set of "candidate" visits to be obtained by mapping the retrieved EHRs to their corresponding patient visits.

Feature Extraction

Determining whether a "candidate" patient visit v_j is relevant to (i.e., satisfies the criteria from) a given cohort description d_i requires access to a rich set of features derived from (a) the cohort description d_i, (b) the patient visit v_j and (c) the interactions between d_i and v_j. To account for the variation between cohort descriptions, we considered multiple strategies for transforming d_i into queries. Let $q_i^{c,e}$ represent the query obtained when using query construction method c and query expansion method e. Likewise, we considered multiple strategies for representing the information encoded in each visit v_j. Hence, we considered r_k^s the textual content provided by

stream s of the electronic health record r_k, and define $v_j^s = \left\{r_1^s, r_2^s, \cdots, r_{N_j}^s\right\}$ as the content of stream s from each report associated with visit v_j. We produced a single feature vector $\boldsymbol{x}_j^i$ encoding information about d_i and v_j by extracting the 14 high-level multi-valued features listed in Table 1.

Table 1: Features extracted for a cohort description $\boldsymbol{d_i}$ and hospital visit $\boldsymbol{v_j}$. Additional details for each feature are provided in Appendix E. $\mathbb{N}$ represents the natural numbers, $\mathbb{R}$ represents the real numbers, and the exponent (if provided) indicates the *dimensionality*, or number of values produced by that feature in the resultant feature vector).

	Feature Description	Domain of Values
$\boldsymbol{F_1}$	number of **criteria** detected in cohort description d_i with each construction method c	$\mathbb{N}^{\lvert C\rvert}$
$\boldsymbol{F_2}$	number of **terms** in $q_i^{c,e}$ for each $c \in C$, and each expansion method $e \in E$	$\mathbb{N}^{(\lvert C\rvert\times\lvert E\rvert)}$
$\boldsymbol{F_3}$	*statistics* of the **normalized inverse document frequency (IDF)** of $q_i^{c,e}$ in each stream $s \in S$ for each c, e.	$\mathbb{R}^{(\lvert A\rvert\times\lvert C\rvert\times\lvert E\rvert\times\lvert S\rvert)}$
	(*features encoding information about the cohort description* $\boldsymbol{d_i}$)	
$\boldsymbol{F_4}$	number of **reports** associated with v_j	$\mathbb{N}$
$\boldsymbol{F_5}$	distribution of **report types** associated with v_j	$\mathbb{R}^{\lvert T\rvert}$
$\boldsymbol{F_6}$	*statistics* of the **number of words** in each $r^s \in v_j^s$ for every s	$\mathbb{N}^{(\lvert A\rvert\times\lvert S\rvert)}$
	$\left($*features encoding information about the candidate visit* $\boldsymbol{v_j}\right)$	
$\boldsymbol{F_7}$	whether the **age** (if any) specified in cohort description i matches the age in any stream of any report $r^s \in v_j$	$\{0,1\}$
$\boldsymbol{F_8}$	whether the **gender** (if any) specified in cohort description i matches the most frequently-mentioned gender in any stream of any report $r^s \in v_j$	$\{0,1\}$
$\boldsymbol{F_9}$	whether the **hospital status** in cohort description i matches the hospital status in any stream of any report $r^s \in v_j$	$\{0,1\}$
$\boldsymbol{F_{10}}$	*statistics* of the **Dirichlet**-smoothed **language model similarity**[37] **(LM:Dir)** between $q_i^{c,e}$ and each $r^s \in v_j$ for every c, e, s	$\mathbb{R}^{(\lvert A\rvert\times\lvert C\rvert\times\lvert E\rvert\times\lvert S\rvert)}$
$\boldsymbol{F_{11}}$	*statistics* of the **Jelinek-Mercer**-smoothed **language model similarity**[37] **(LM:JM)** between $q_i^{c,e}$ and each $r^s \in v_j$ for every c, e, s	$\mathbb{R}^{(\lvert A\rvert\times\lvert C\rvert\times\lvert E\rvert\times\lvert S\rvert)}$
$\boldsymbol{F_{12}}$	*statistics* of the **BM25 similarity**[38] between $q_i^{c,e}$ and each $r^s \in v_j$ for every c, e, s	$\mathbb{R}^{(\lvert A\rvert\times\lvert C\rvert\times\lvert E\rvert\times\lvert S\rvert)}$
$\boldsymbol{F_{13}}$	*statistics* of the **TF-IDF similarity**[7] between $q_i^{c,e}$ and each $r^s \in v_j$ for every c, e, s	$\mathbb{R}^{(\lvert A\rvert\times\lvert C\rvert\times\lvert E\rvert\times\lvert S\rvert)}$
$\boldsymbol{F_{14}}$	*statistics* of the **term frequency (TF)** between $q_i^{c,e}$ and each $r^s \in v_j$ for every c, e, s	$\mathbb{R}^{(\lvert A\rvert\times\lvert C\rvert\times\lvert E\rvert\times\lvert S\rvert)}$
	$\left($*features encoding the relationship between the cohort description* $\boldsymbol{d_i}$ *and candidate visit* $\boldsymbol{v_j}\right)$	

As shown, 10 of the 14 features illustrated in Table 1 are multivalued, i.e., consist of distinct values for each possible query representation $q_i^{c,e}$ of d_i and each stream s of v_j^s (where applicable). Each

of these values corresponds to a single entry in the resultant feature vector – i.e., F_1 corresponds to five entries in the generated feature vector. Moreover, Features F_3, F_6 and F_{10}-F_{14} capture the distribution of feature values extracted for each component of the query (F_1) or for each report associated with the hospital visit (F_6, F_{10}-F_{14}) using five *aggregation methods* (described below). Of note are features F_{10}-F_{14} which incorporate standard relevance models from information retrieval to measure the relevance between the criteria in $q_i^{c,e}$ and each stream of visit v_j^s.

Aggregation Methods. To capture the distribution of feature values obtained using different streams or for each report associated with a candidate visit, we considered five aggregating statics $A = \{\text{mean, minimum, maximum, variance, sum}\}$.

The Learning Relevance Model

The role of the learning relevance model (LRM) is to infer a *relevance score* s_j^i between every candidate visit v_j and the cohort description d_i using the feature vector $\boldsymbol{x}_j^i$ extracted above. This is accomplished by using the *pairwise* strategy of learning-to-rank. Given (1) feature vectors $[\boldsymbol{x}_1^i, \cdots, \boldsymbol{x}_N^i]$ associated with candidate visits $[v_1, \cdots, v_N]$ and (2) "gold-standard" relevance judgments $[y_1^i, \cdots, y_N^i]$ indicating the relevance of each candidate visit to d_i, the Random Forest is trained to infer the scores $[s_1^i, \cdots, s_N^i]$ which result in the optimal ordering of hospital visits as indicated by $[y_1^i, \cdots, y_N^i]$. We investigated multiple learning-to-rank approaches when designing the Learning Relevance Model, including pointwise, pairwise, and listwise strategies. Specifically, we analyzed the performance of RankNet, RankBoost, AdaRank, Coordinate Ascent, LambdaMART, Multiple Additive Regression Trees (MART), ListNet, and Random Forests. We found Random Forests to obtain the best performance on a small held-out set of cohort descriptions. Although we investigated multiple sets of model parameters, we found no statistically significant change in performance when changing the ranking criterion (entropy vs Gini impurity), number of sampled features ($\sqrt{x}$, $\log(x)$, or $0.1x$), or maximum forest size (200, 500, or ∞).

Goals Specific to Aim 4 in the main project: *Validate the usefulness of the patient cohort identification system by collecting feedback from clinicians and medical students.* For each query, medical experts shall examine the top ranked cohorts for common precision errors (false positives), and the bottom five ranked common recall errors (false negatives). In a very fruitful collaboration, both the Temple University team and the UTD team have participated in the evaluation and validation of the patient cohort identification system implemented in the MERCuRY system. We have assembled *250 clinically relevant queries* that are used by neurologists to evaluate the quality of the EEG reports/records considered relevant by the patient cohort retrieval system in its current form. In addition, we have collected judgements of the patient cohorts through a secure-interface generated at UTD. We primarily evaluated the MERCuRY system according to its ability to retrieve patient cohorts with and without learning to rank. To this end, we generated a set of 100 evaluation queries. For each query, we retrieved the ten most relevant patients as well as a random sample of ten additional patients retrieved between ranks eleven and one hundred. We asked six relevance assessors to judge whether each of these patients belonged or did not belong to the given cohort. Moreover, the order of the documents (and queries) were randomized and judges were not told the ranked position of each patient. Each query and patient pair was judged by at least two

relevance assessors, obtaining an inter-annotator agreement of 80.1% (measured by Cohen's kappa).

This experimental design allowed us to evaluate not only the set of patients retrieved for each cohort, but also the individual rank assigned to them. Specifically, we adopted standard measures for information retrieval effectiveness, where patients labeled as belonging to the cohort were considered *relevant* to the cohort query, and patients labelled as not belonging to the cohort were considered as *non-relevant* the cohort query. Note that because our relevance assessments consider only a sample of the patients retrieved for each query, we adopted two measures of ranked retrieval quality: the Mean Average Precision (MAP) and the Normalized Discounted Cumulative Gain (NDCG). The MAP provides a single measurement of the quality of patients retrieved at each rank for a particular topic. Likewise, the NDCG measures the *gain* in overall cohort quality obtained by including the patients retrieved at each rank. This gain is accumulated from the top-retrieved patient to the bottom-retrieved patient, with the gain of each patient discounted at lower ranks. Lastly, we computed the "Precision at 10" metric (P@10), which measures the ratio of patients retrieved in the first ranks which belong to the patient cohort.

4. Defining Hierarchical epileptiform Activity Descriptors (HAD) for EEGs

After defining and designing a fine-grained hierarchy of activity descriptors last year, the team from the University of Texas at Dallas has proceeded to design an automatic methodology of discovering long-distance relations between concepts from the HAD identified in the same EEG report, e.g:

> CLINICAL HISTORY: 55 year old man admitted for [change in mental status]$_{MEDICAL\ PROBLEM}$, with a past medical history of [GI bleed]$_{MEDICAL\ PROBLEM}$, [anemia]$_{MEDICAL\ PROBLEM}$ [encephalopathy]$_{MEDICAL\ PROBLEM}$, and others.
>
> MEDICATIONS: [Pantoprazole]$_{TREATMENT}$, [Folic Acid]$_{TREATMENT}$, [Carvedilol]$_{TREATMENT}$
>
> INTRODUCTION: Digital video EEG was performed at the bedside using standard 10-20 system of electrode placement with 1 channel EKG.
>
> DESCRIPTION OF THE RECORD: The background EEG is characterized by [slowing]$_{EEG\ ACTIVITY}$ and [disorganization]$_{EEG\ ACTIVITY}$. There is prominent shifting arrhythmic [delta activity]$_{1EEG\ ACTIVITY}$ more prominent in the left mid to anterior temporal region. [Photic stimulation]$_{EEG\ EVENT}$ generates scant [driving]$_{EEG\ ACTIVITY}$.
>
> IMPRESSION: Abnormal EEG due to:
>
> 1. Marked background [slowing]$_{EEG\ ACTIVITY}$ and [disorganization]*EEG ACTIVITY*
> 2. Some arrhythmic [delta activity]$^{1}{}_{EEG\ ACTIVITY}$
>
> CLINICAL CORRELATION: These findings are supportive of a [bihemispheric disturbance of cerebral function]$_{MEDICAL\ PROBLEM}$. These are nonspecific findings which can be seen in a toxic and metabolic [encephalopathy]$_{MEDICAL\ PROBLEM}$ and/or underlying [cerebrovascular disease]$_{2MEDICAL\ PROBLEM}$.

In the exemplified EEG report seven relations between HAD concepts need to be identified:

(R1) [delta activity]$_{\text{EEG ACTIVITY}}$ →*Evidences*→[cerebrovascular disease]$_{\text{MEDICAL PROBLEM}}$;
(R2) [slowing]$_{\text{EEG ACTIVITY}}$ →*Evidences*→[bihemispheric disturbance of cerebral function]$_{\text{MEDICAL PROBLEM}}$;
(R3) [disorganization]$_{\text{EEG ACTIVITY}}$ →*Evidences*→[bihemispheric disturbance of cerebral function]$_{\text{MEDICAL PROBLEM}}$;
(R4) [bihemispheric disturbance of cerebral function]$_{\text{MEDICAL PROBLEM}}$→Evidences![encephalopathy]$_{\text{MEDICAL PROBLEM}}$;
(R5) [Pantoprazole]$_{\text{TREATMENT}}$→*TREATMENT-FOR*→ [GI bleed]$_{\text{MEDICAL PROBLEM}}$;
(R6) [Folic acid]$_{\text{TREATMENT}}$→*TREATMENT-FOR*→[anemia]$_{\text{MEDICAL PROBLEM}}$ and
(R7) [photic stimulation]$_{\text{EEG EVENT}}$→*Evokes*→[driving]$_{\text{EEG ACTIVITY}}$.

Thus, in addition to the hierarchical relations incorporated in the HAD, we found necessary to discover three forms of relations between the concepts of the hierarchy, namely: *EVIDENCES*, *EVOKES*, and *TREATMENT-FOR.*

The *EVIDENCES* relation considers (a) EEG events, EEG activities, treatments, and medical problems as providing evidence for (b) medical problems mentioned in the EEG report. The *EVOKES* relation represents the relationship where a medical concept evokes an EEG activity. EEG events, other EEG activities, medical problems and treatments can all evoke EEG activities. The *TREATMENT-FOR* relation links treatments to the medical problems for which they are prescribed. In addition, we made the decision to annotate relations between medical concepts, and not between their mentions in the EEG report. Because the same concept can be mentioned multiple times in the same EEG report, the representation of concepts achieved while pre-processing the EEG reports by (i) their normalized mention and (ii) their attributes made it possible to recognize co-referring mentions of the same concept by simply grouping concepts with the same normalized mention name and attribute values. Therefore, all co-referring mentions were considered a unique concept, and relations were annotated between unique concept pairs.

5. Automated Tagging of HADs in Medical Texts

We focused on the automatic identification of relations between pairs of HAD concepts automatically annotated in EEG reports, regardless of their presence in the same sentence, section or across sentences and sections of the report, has been made possible by a novel deep learning system that we designed and implemented in the previous year, namely the Multi-task Active Deep Learning (MTADL) paradigm. This year we developed the **Memory-Augmented Active Deep Learning** (MAADL) system, with the goal of identifying binary relations between HAD concepts, as designed in the Aim 3 of the supplement project. MAADL combines the strength of the Active Learning framework with the advantages of deep learning. While deep learning methods provide unprecedented performance in many tasks, active learning allows a deep learner to achieve this performance with less manually annotated training data, as it exposes the system to new examples on which its performance is still suffering. The paper that reported on the MTADL paradigm, authored by Ramon Maldonado, Travis Goodwin and Sanda Harabagiu, from the University of Texas at Dallas, received the AMIA Clinical Research Informatics Award at the 2018 AMIA Informatics Summit.

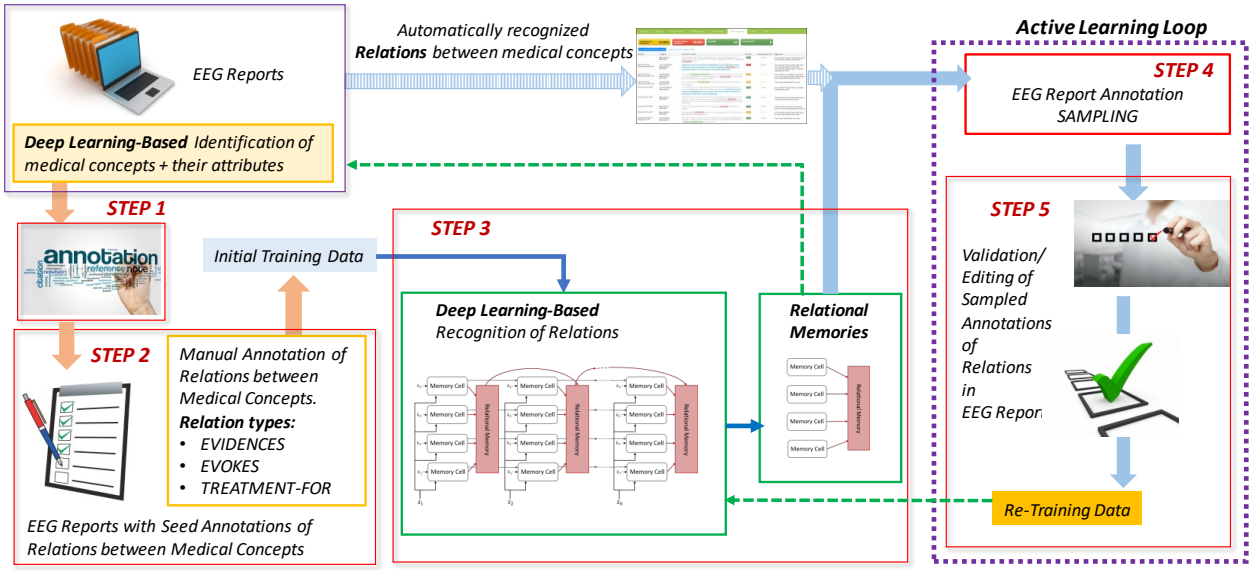

Figure 10: The Memory-Augmented Active Deep Learning (MAADL) system for automatically identifying relations between pairs of medical concepts in EEG reports

The identification of relations between medical concepts in MAADL, illustrated in Figure 10, uses the following five steps:

STEP 1: The development of an annotation schema for relations between medical concepts in EEG reports;
STEP 2: Annotation of relations between medical concepts in the initial training data;
STEP 3: Design of a deep learning method for detecting relations between medical concepts in the EEG reports;
STEP 4: Development of sampling methods for the MAADL;
STEP 5: Usage of the Active Learning system which involves:
    STEP 5.a: Accepting/Editing annotations of sampled examples of relations between medical concepts in EEG reports;
    STEP 5.b: Re-training the deep learning method and evaluating the re-trained system.

STEP 1: Annotation Schema for relations between pairs of medical concepts in EEG reports: The annotation schema has been developed in the Aim 3 of the supplement project.

STEP 2: Initial Relation Annotations: A set of 40 EEG reports with 198 *EVIDENCES* relations, 146 *EVOKES* relations, and 72 *TREATMENT-FOR* relations were manually annotated and used as the initial training data for the relation detection system. This set of EEG reports had previously been manually annotated with medical concepts and their attributes to ensure errors in concept/attribute detection did not affect relation detection.

STEP 3: Design of Deep Learning Architecture for the Memory-Augmented Active Deep Learning System: We designed a deep learning architecture, called **EEG-RelNet**, which provides an end-to-end detection of relations between medical concepts in each EEG report by using a neural network augmented with two types of memories: (i) a memory for each medical concept;

and (ii) a memory for each relation between each pair of medical concepts. Moreover, the relational memory is dynamic as it changes to model the specific relations observed in each EEG report.

STEP 4: Active Learning Sampling Method: To improve the quality of the identified relations between medical concepts in EEG reports, as illustrated in Figure 10, an active learning loop is designed. In an active learning framework, the sampling method is used to automatically select examples of relations for human validation. Since this work is focused on relation detection between pairs of medical concepts, we chose a sampling method that only prioritizes relation detection performance. Therefore, we selected standard uncertainty sampling whereby EEG reports containing relations for which the model is most uncertain are selected for manual validation. The uncertainty of a report is measured at the report level by averaging the uncertainty of each relation classification decision in the report. The uncertainty of a relation classification decision is calculated using Shannon Entropy, $H(R) = \sum_t R_t \times logR_t$, where R is a vector representing the probability distribution over possible relation types. These probability distributions are derived by EEG-RelNet from the learned dynamic relation memory, as shown in Figure 10.

STEP 5: Usage of the Memory-Augmented Active Deep Learning System: As shown in Figure 4, each iteration of active learning involves using the EEG-RelNet to make automatic relation annotations on the unlabeled EEG reports, selecting the most informative examples for manual validation, and re-training the EEG-RelNet using the new set of validated training examples.

EEG-RelNet: a Deep Learning Architecture for long-distance Relation Detection in EEG Reports While medical concepts (EEG activities, EEG events, medical problems, treatments and tests) are available in each EEG report, due to the preprocessing that was applied to the entire TUH EEG corpus, inference of the *EVIDENCES*, *EVOKES*, and *TREATMENT-FOR* relations between pairs of such concepts was produced through dynamic memories based on neural networks, capable to capture the implicit participation of each medical concept in a relation of interest. This was made possible because we developed the **EEG-RelNet**, a deep neural network architecture that operates on the full text of an EEG report considering all medical concepts identified in the report to detect relations of the type *EVIDENCES*, *EVOKES*, and *TREATMENT-FOR* between any pair of concepts. More specifically, given the full text of an EEG report and the set of medical concepts identified in that report, EEG-RelNet can predict whether there is relation of type t, R_{ij}^t, between any pair of medical concepts c_i and c_j recognized in the report. To do so, EEG-RelNet processes the EEG report, one sentence at a time, reading its words, encoding the information from the sentence, processing the sentence information in the dynamic relational memory, and predicting each type of relation based on the dynamic memories after they have processed each sentence in the EEG report. The three modules of EEG-RelNet are:

- the **Input Encoding Module** which encodes information from the report at concept- and sentence-level embedding vectors, which are used throughout the deep learning architecture;
- the **Dynamic Relational Memory Module** which maintains and updates a set of hidden states called memories to capture accumulated information about each medical concept and potential relation in the EEG report;

- the **Output Module** which uses the updated memories to determine the most likely relations (and their types) between medical concepts in the EEG report.

In the remainder of this section, we provide a detailed description of each module of EEG-RelNet.

The Input Encoding Module. The role of this module is to learn (1) an embedding encoding each medical concept as well as each of its attributes and (2) an embedding encoding the information from each sentence in the EEG report. Formally, we represent an EEG report as a set of medical concepts, $C = \{c_1, \ldots, c_d\}$ and a sequence of sentences, $[s_1, \ldots, s_n]$. Each medical concept, c_i is associated with several *N*-dimensional vectors called embeddings: (a) an embedding for the normalized concept name, $\widetilde{c_i} \in \mathbb{R}^N$ and (b) separate embeddings for each of its *A* attributes values $\{a_1^{c_i}, \ldots, a_A^{c_i}\}$. Thus, the embedding $\overrightarrow{c_i}$ for a medical concept is created by (1) concatenating the embedding for the name of the medical concept with the embedding for each of its attributes and (2) projecting this concatenated vector using a learned weight matrix $W_c \in \mathbb{R}^{N \times N \times (|A|+1)}$, i.e. $\overrightarrow{c_i} = W_c \times [\widetilde{c_i}, a_1^{c_i}, \ldots, a_A^{c_i}]$. In this way, each medical concept is represented by an embedding, $\overrightarrow{c_i}$ which is a vector in $\mathbb{R}^N$.

Participation of medical concepts in relations is informed by the context of each concept in the text of the EEG report. Contextual information is provided by the words of the sentence where the concept is mentioned, hence a representation of words from each sentence as is also desirable. Therefore, we learn an embedding e_i for each word w_i in a sentence, enabling us to represent each sentence as a sequence of embeddings $E = [e_1, \ldots, e_m]$ such that the elements of E occur in the same order as the words from the sentence. While the traditional choice for combining and composing the embeddings in E into a single sentence embedding would be a Recurrent Neural Network (RNN), we instead adopt a more recent and significantly more efficient strategy, namely a *positional mask*, such that the *k*-th sentence from the EEG report is represented as: $\overrightarrow{s_k} = \sum_{i=1}^{m} f_i \odot e_i$, given that the sentence had m words, and the vectors $[f_1, \ldots, f_m]$ represent the learned positional mask while $\odot$ is the element-wise product. It is important to note that the same vectors $[f_1, \ldots, f_m]$ are used when each new sentence is encoded and they are learned jointly with the other parameters of the deep learning model.

The Dynamic Relational Memory Module. Because EEG reports often contain long-distance relations between concepts we relied on a *Dynamic Relational Memory* (DRM) Module to keep track of the interactions between medical concepts in each report. The DRM accumulates information about medical concepts and the relations between them by processing each sentence encoded by the Input Module and updating a set of hidden states, called memories. Specifically, given a sentence embedding, $\overrightarrow{s_k}$, and the corresponding set of concept embeddings $[\overrightarrow{c_1}, \ldots, \overrightarrow{c_d}]$, there are two scenarios for each $\overrightarrow{c_i}$: (scenario 1): the medical concept c_i has not been mentioned in any previous sentence, thus its Concept Memory needs to be accounted for using a single, shared Concept Memory Cell; and (scenario 2): the concept c_i has been previously mentioned, and thus its corresponding Concept Memory needs to be updated. Moreover, since each medical concept c_i may participate in a relation, in (scenario 1), a unique Relation Memory needs to account for each relation in which the concept participates, whereas in (scenario 2) the corresponding Relation Memory needs to be updated. If an EEG report refers to *d* medical concepts, there will be *d* Concept Memory cells and $d \times (d-1)$ Relation Memory cells. The Dynamic Relational Memory (DRM) consists of the entire set of Concept and Relation Memories in an EEG report.

The Concept Memories are organized as a Key-Value Memory Network. Key-value paired memories are a generalization of the way context of concepts is stored in memory. In a Key-Value Memory Network, the lookup (addressing) stage is based on the key vector while the reading stage (giving the returned result) returns the value memory. Consequently, in EEG-RelNet, memory vectors are tied to so-called key vectors enabling the model to only update a memory vector when the input sentence has context that is relevant to the memory's associated key vector. The dynamic relations memory model of EEG RelNet is illustrated in Figure 11.

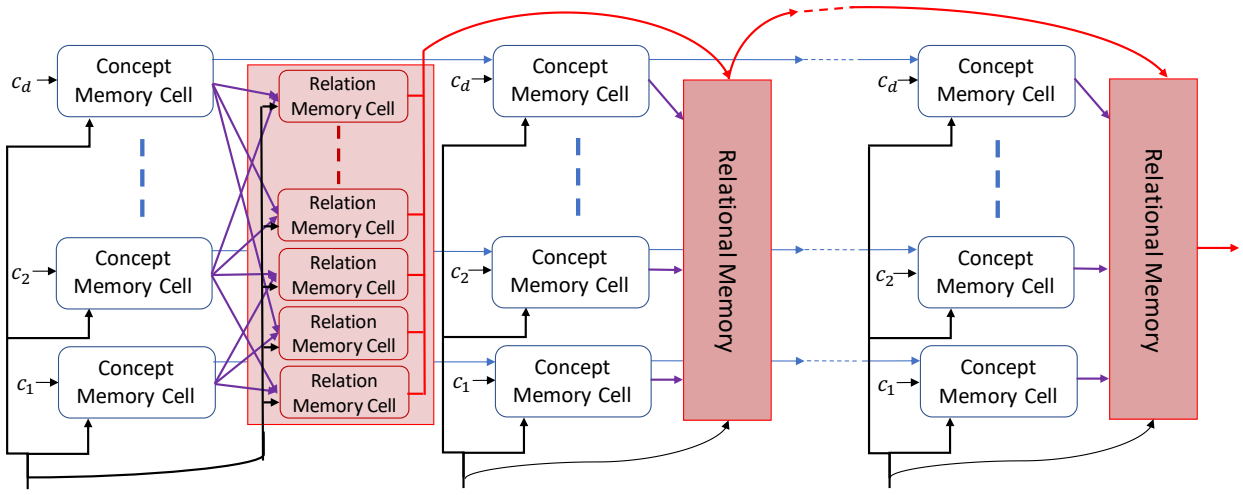

Figure 11: The Dynamic Relational Memory Module of EEG-RelNet. The Dynamic Relational Memory Module processes n sentences, updating a set of d Concept Memories and $d \times (d-1)$ Relation Memories for each sentence.

It is well known that when concept embeddings are used as key vectors, the associated memory vectors will accumulate information about those concepts. Consequently, in EEG-RelNet, concept embeddings are used as key vectors allowing the network to update each Concept Memory, h_i, if an input sentence is relevant to the concept, c_i.

The Concept Memory Cell, illustrated in Figure 12, is used to update a Concept Memory, h_i, given a medical concept embedding, c_i, and a sentence encoding, $\overrightarrow{s_k}$, via the following equations:

$$g_i^c = \sigma(< \overrightarrow{s_k}, h_i + c_i >) \quad (21)$$

$$\widetilde{h_\iota} = \phi(W_u h_i + W_v c_i + W_s \overrightarrow{s_k}) \quad (22)$$

$$h_i \leftarrow h_i + g_i^c \odot \widetilde{h_\iota} \quad (23)$$

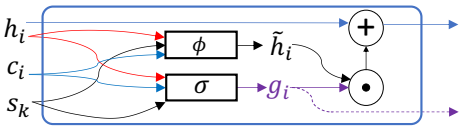

Figure 12: Concept Memory Cell

where W_u; W_v and W_s are trainable weight matrices in $\mathbb{R}^{N \times N}$, $< \cdot, \cdot >$is the inner product, σ is the sigmoid function and ϕ is a Parametric Rectified Linear Unit (PReLU). Equation 21 is a gating function that determines how much the k-th input sentence affects the i-th Concept Memory such that $g_i^c \in [0,1]$ values close to 1 indicate sentence s_k is relevant to medical concept c_i and values

close to 0 indicate the opposite. Equation 22 defines the candidate Concept Memory that will be used to update the existing Concept Memory, h_i, after it is scaled by g_i^c as shown in equation 23.

As illustrated in Figure 11, when each sentence s_i is processed, the DRM uses and updates not only concept memories, but also a much larger set of relation memories. This is explained by the fact that, unfortunately, maintaining a single memory vector for each concept is not sufficient for modeling concepts that participate in multiple relations, especially when those relations involve concepts that are mentioned at significant distance in the EEG report. Thus, to model the interactions each concept has with each other concept in the same EEG report, we maintain a set of Relation Memories corresponding to each pair of concepts from the EEG report, $\{r_{ij}: i,j \in C, i \neq j\}$, where C is the set of medical concepts in the EEG report. Each Relation Memory is updated using the Relation Memory Cell illustrated in Figure 13 via the following equations:

$$g_{ij}^r = g_i^c \times g_j^c \times \sigma(< \overrightarrow{s_k}, r_{ij} >) \quad (24)$$

$$\widetilde{r_{ij}} = \phi(W_A r_{ij} + W_B \overrightarrow{s_k}) \quad (25)$$

$$r_{ij} \leftarrow r_{ij} + g_{ij}^r \odot \widetilde{r_{ij}} \quad (26)$$

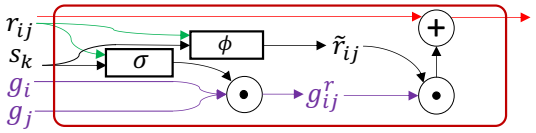

Figure 13: Relation Memory Cell

where W_A and W_B are trainable weight matrices in $\mathbb{R}^{N \times N}$. As in the Concept Memory Cell, the Relation Memory Cell uses a gating function (equation 24) and a candidate memory (equation 25) to update the Relation Memory in a way that reflects how relevant the input sentence, s_k, is to the concept pair, (c_i, c_j). To compute the gate value g_{ij}^r, the Relation Memory Cell uses the two concept gate values, g_i^c; and g_j^c from the Concept Memory Cells for concepts c_i and c_j, ensuring that input sentences that are relevant to either concept can be used to update the Relation Memory. By maintaining a memory vector for each pair of concepts and updating that memory vector as the model accumulates information across each sentence in an EEG report, thus the EEG-RelNet can be interpreted as constructing a *local latent knowledge graph* for each EEG report, where each Relation Memory represents a possible relation in the graph.

The Output Module. The output module makes use of the Dynamic Relational Memory updated after processing the last sentence in the EEG report to identify relations (and their types) between any pair of medical concepts from the report. The relation prediction, $\widehat{R_{ij}}$ between medical concepts c_i and c_j is produced by passing the Concept Memories associated with concepts c_i and c_j along with the Relational Memory r_{ij} to two fully connected PReLU layers followed by a softmax layer:

$$\boldsymbol{q_{ij}} = \phi(W_q[h_i, h_j, r_{ij}]) \quad (27)$$

$$\boldsymbol{R^{ij}} = softmax(\phi(W_z \boldsymbol{q_{ij}}) \quad (28)$$

where $W_q \in \mathbb{R}^{N \times 3N}$ and $W_z \in \mathbb{R}^{4 \times N}$ are learned weight matrices, and ϕ is a Parametric Rectified Linear Unit. $\boldsymbol{R^{ij}}$ is a probability distribution over 4 possible relations: the 3 relation types described

in the annotation schema and a 4th type indicating no relation. Consequently, the relation (if any) detected between concepts c_i and c_j is given by $\widehat{R_{ij}} = argmax_t\ R_t^{ij}$.

Temple University:

The Temple University team employed a large number of students and one postdoc on this project. There were four major labor categories:

(1) Postdoc: Scott Yang was the sole postdoc recruited for this project. Since he joined the project with limited programming experience (primarily MATLAB), he received intensive training on a variety of software skills: Unix/Linux command line programming, C/C++ programming, Python programming and html/web programming. He was also introduced to a rigorous software engineering process including revision control software and GitHub/GitLab. Further, he received training on state-of-the-art machine learning packages such as Theano, Kera, TensorFlow and HTK.

With respect to professional development, in addition to learning how to a set of advanced features in Microsoft Office and Mendeley to develop and manage publications, he received significant training on proposal writing, technical presentations and technical writing.

He was encouraged to publish and attend professional conferences. These are described in the extensive publication list presented later in this report.

(2) Graduate Students: We have employed six graduate students on this project. Four were pursuing PhDs and two pursued MS degrees. Therefore, an important part of their training included normal graduate student training such as didactic graduate-level courses, thesis and dissertation proposals and PhD preliminary exams.

As with (1), they also received significant amounts of programming training and professional development training through publications and conference presentations. Further, they led the development of several real-time demonstration systems and were trained how to develop and present technical demonstrations.

Training was so successful that we lost two of these students to industry jobs before they graduated. Their marketability was a testament to their training, as students trained in machine learning are received very attractive job offers from industry these days. Both students will complete their PhDs remotely, but working full-time certainly slows the process down.

(3) Undergraduate Students: We employ a large number of undergraduates on this project in four capacities: (1) data annotation, (2) software engineering, (3) web development and (4) IT support. The data annotators learn how to annotate EEGs in a manner similar to the way neurologists annotate the data. We have a well-developed process for training them and in a publication cited later we show that their annotation accuracy is excellent compared to clinically-trained neurologists.

The software engineering training is similar to what was previously described. Our undergraduates become expert programmers in C/C++ and Python, which makes them very valuable. We assist them in obtaining summer internships and full-time employment once they are trained. It is very clear that they qualify for jobs they would not normally be able to get after this training.

Several of the students work in positions we described as IT support, which involves maintaining our Linux cluster and server environment, and web support, which is the primary means by which we disseminate information and manage customer relationships. Needless to say, these skills, which typical ECE students at Temple don't develop, make them extremely marketable in sectors ranging from defense to financial engineering.

Finally, all students involved in this project received extensive training on how to read and interpret EEGs. They worked closely with the Department of Neurology at Temple Hospital and attended many EEG readings sessions conducted by the hospital as part of their medical student training.

University of Texas at Dallas:

Six PhD students have been advised for their research conducted for this project at University of Texas at Dallas.

Travis Goodwin has defended his PhD thesis in March 2018, after 6 years of PhD studies in Computer Science at UTD, advised by Prof. Sanda Harabagiu. He has accepted a fellowship at NIH starting in June 2018. He participated in our project for the past three years, developing novel research in the area of multi-modal indexing, inference of underspecified information in the EEG reports and interaction of various factors in the EEG reports. In November 2017, he received the **Homer Warner award** at the 2017 AMIA Symposium for the paper co-authored with his adviser, Dr. Sanda Harabagiu, entitled "*Inferring Clinical Correlations from EEG Reports with Deep Neural Learning*". Travis has also been working on defining the HAD tags under the supplement project. He has also worked on using deep learning methods for the automatic annotation of HAD tags as well for generating data-driven neural knowledge representations of the knowledge discerned from the EEG reports. He is also the co-author of a paper that received the **AMIA Clinical Research Informatics Award** at the 2018 AMIA Informatics Summit, in March 2018. The paper is entitled "*Memory-Augmented Active Deep Learning for Identifying Relations Between Distant Medical Concepts in Electroencephalography Reports"* with the authors: Ramon Maldonado, Travis Goodwin and Sanda Harabagiu. Travis has successfully submitted 23 conference papers and has received in 2016 the **Best Student Paper Award** at the ACM International Conference in Information and Knowledge Management (CIKM-2016), a major conference on knowledge managements and information retrieval. In addition, he has published four journal papers. These accomplishments exceeded Travis's Individual Development Plans (IDPs).

Ramon Maldonado is a 3rd year PhD student in Computer Science at UTD, advised by Prof. Sanda Harabagiu, who has performed research on automatically identifying all medical concepts and recognizing the relations between them in the Temple University Hospital EEG data, in the form of EEG reports documenting 25,000 sessions and 15,000 patients collected over 12 years at Temple University Hospital. Ramon is a qualified PhD student, by passing a set of qualifying exams, while developing new techniques for his research, which is remarkable. He is the lead author on the paper entitled "*Memory-Augmented Active Deep Learning for Identifying Relations Between Distant Medical Concepts in Electroencephalography Reports",* which has received the **AMIA Clinical Research Informatics Award** at the 2018 AMIA Informatics Summit, in March 2018. Ramon is also the lead authors of the paper entitled *"Active Deep Learning-Based Annotation of Electroencephalography Reports for Cohort Identification",* which was **nominated for Distinguished Paper Award** at the 2017 American Medical Informatics Association Joint Summits on Clinical Research Informatics (AMIA-CRI*).* Ramon has also been lead author on a paper published in *the American Medical Informatics Association Annual Symposium (AMIA-2017)* which introduced a novel form of medical knowledge, called *medical knowledge embeddings*, developed from the TUH corpus, in the form of EEG-MKE. The EEG-MKE have been described in the paper entitled "*Deep Learning Meets Biomedical Ontologies: Knowledge Embeddings for Epilepsy*". Moreover, Ramon was a co-author on a journal paper published this year in the Journal of Biomedical Informatics. These accomplishments meet Ramon's Individual Development Plans (IDPs).

Stuart Taylor is a 2nd year PhD student in Computer Science at UTD, advised by Prof. Sanda Harabagiu, who has worked on the initial development of the active deep learning for annotating EEG reports. Stuart

is a qualified PhD student, by passing a set of qualifying exams, while developing new techniques for his research, which is remarkable. He worked on the generation of queries for the evaluation of patient cohorts. Stuart has published a poster at the *American Medical Informatics Association Annual Symposium (AMIA)* in 2017 and submitted a full paper at the *American Medical Informatics Association Annual Symposium (AMIA)* in 2018. has plan on working on the recognition of HAD tags in biomedical texts. These accomplishments meet Stuart's Individual Development Plans (IDPs).

Pracheta Sahoo, Shasha Jin and SaraRouhani, are qualified PhD students in Computer Science at UTD, advised by Prof. Vibhav Gogate, who have participated in the project a few months this year to help with machine learning frameworks for deep learning and for evaluation of the results as well as to produce relevance judgements. They are part of the Statistical Relational Artificial Intelligence and Machine Learning Lab, led by Prof. Gogate at UTD. They have published papers in the AAAI conference.

C. PRODUCTS

C.1 PUBLICATIONS

Are there publications or manuscripts accepted for publication in a journal or other publication (e.g., book, one-time publication, monograph) during the reporting period resulting directly from this award?

Yes

Publications Reported for this Reporting Period

Public Access Compliance	Citation
Complete	Obeid I, Picone J. The Temple University Hospital EEG Data Corpus. Frontiers in neuroscience. 2016;10:196. PubMed PMID: 27242402; PubMed Central PMCID: PMC4865520; DOI: 10.3389/fnins.2016.00196.
Complete	Obeid I, Picone J. The Temple University Hospital EEG Data Corpus. Frontiers in neuroscience. 2016;10:196. PubMed PMID: 27242402; PubMed Central PMCID: PMC4865520; DOI: 10.3389/fnins.2016.00196.
Complete	Obeid I, Picone J. The Temple University Hospital EEG Data Corpus. Frontiers in neuroscience. 2016;10:196. PubMed PMID: 27242402; PubMed Central PMCID: PMC4865520; DOI: 10.3389/fnins.2016.00196.
N/A: Not Journal	Obeid I, Picone J. Augmentation of Brain Function: Facts, Fiction and Controversy. 1 ed. Lebedev M, editor. Lausanne, Switzerland. Frontiers Media S.A.; 2016. Chapter 8, The Temple University Hospital EEG Data Corpus; 394-398p. 394-398p.
Non-Compliant	Eva V, Ahsan T, Shah V, Jamshed D, Golmohammadi M, Obeid I, Picone J. Electroencephalographic Slowing: A Primary Source of Error in Automatic Seizure Detection. Proceedings of the IEEE Signal Processing in Medicine and Biology Symposium. 2017 December 01:1-5. DOI: 10.1109/SPMB.2017.8257018.
Non-Compliant	Golmohammadi M, Ziyabari S, Shah V, Obeid I, Picone J. Gated Recurrent Networks for Seizure Detection. Proceedings of the IEEE Signal Processing in Medicine and Biology Symposium. 2017 December 01:1-5. DOI: 10.1109/SPMB.2017.8257020.
Non-Compliant	Shah V, Golmohammadi M, Ziyabari S, von Weltin E, Obeid I, Picone J. Optimizing Channel Selection for Seizure Detection. Proceedings of the IEEE Signal Processing in Medicine and Biology Symposium. 2017 December 06:1-5. DOI: 10.1109/SPMB.2017.8257019.
Complete	Goodwin TR, Maldonado R, Harabagiu SM. Automatic recognition of symptom severity from psychiatric evaluation records. Journal of biomedical informatics. 2017 November;75S:S71-S84. PubMed PMID: 28576748; PubMed Central PMCID: PMC5705296; DOI: 10.1016/j.jbi.2017.05.020.
Complete	Goodwin TR, Maldonado R, Harabagiu SM. Automatic recognition of symptom severity from psychiatric evaluation records. Journal of biomedical informatics. 2017 November;75S:S71-S84. PubMed PMID: 28576748; PubMed Central PMCID: PMC5705296; DOI: 10.1016/j.jbi.2017.05.020.
Complete	Goodwin TR, Maldonado R, Harabagiu SM. Automatic recognition of symptom severity from psychiatric evaluation records. Journal of biomedical informatics. 2017 November;75S:S71-S84. PubMed PMID: 28576748; PubMed Central PMCID: PMC5705296; DOI: 10.1016/j.jbi.2017.05.020.
Non-Compliant	Golmohammadi M, Ziyabari S, Shah V, Obeid I, Picone J. Deep Architectures for Spatio-Temporal Modeling: Automated Seizure Detection in Scalp EEGs. Proceedings of the IEEE International Conference on Machine Learning and Applications (ICMLA). 2018 July 01:1-6. DOI: 10.1109/ICMLA.2018.00118.
Complete	Shah V, von Weltin E, Lopez S, McHugh JR, Veloso L, Golmohammadi M, Obeid I, Picone J. The Temple University Hospital Seizure Detection Corpus. Frontiers in neuroinformatics. 2018;12:83. PubMed PMID: 30487743; PubMed Central PMCID: PMC6246677; DOI: 10.3389/fninf.2018.00083.
Complete	Shah V, von Weltin E, Lopez S, McHugh JR, Veloso L, Golmohammadi M, Obeid I, Picone J. The Temple University Hospital Seizure Detection Corpus. Frontiers in

	neuroinformatics. 2018;12:83. PubMed PMID: 30487743; PubMed Central PMCID: PMC6246677; DOI: 10.3389/fninf.2018.00083.
Complete	Shah V, von Weltin E, Lopez S, McHugh JR, Veloso L, Golmohammadi M, Obeid I, Picone J. The Temple University Hospital Seizure Detection Corpus. Frontiers in neuroinformatics. 2018;12:83. PubMed PMID: 30487743; PubMed Central PMCID: PMC6246677; DOI: 10.3389/fninf.2018.00083.
Complete	Golmohammadi M, Harati Nejad Torbati AH, Lopez de Diego S, Obeid I, Picone J. Automatic Analysis of EEGs Using Big Data and Hybrid Deep Learning Architectures. Frontiers in human neuroscience. 2019;13:76. PubMed PMID: 30914936; PubMed Central PMCID: PMC6423064; DOI: 10.3389/fnhum.2019.00076.
Complete	Golmohammadi M, Harati Nejad Torbati AH, Lopez de Diego S, Obeid I, Picone J. Automatic Analysis of EEGs Using Big Data and Hybrid Deep Learning Architectures. Frontiers in human neuroscience. 2019;13:76. PubMed PMID: 30914936; PubMed Central PMCID: PMC6423064; DOI: 10.3389/fnhum.2019.00076.
Complete	Golmohammadi M, Harati Nejad Torbati AH, Lopez de Diego S, Obeid I, Picone J. Automatic Analysis of EEGs Using Big Data and Hybrid Deep Learning Architectures. Frontiers in human neuroscience. 2019;13:76. PubMed PMID: 30914936; PubMed Central PMCID: PMC6423064; DOI: 10.3389/fnhum.2019.00076.
N/A: Not Journal	Golmohammadi M, Shah V, Obeid I, Picone J. Machine Learning Applications in Medicine and Biology. 1 ed. Obeid I, Picone J, editors. New York, New York, USA. Springer-Verlag; 2020. Chapter 1, Deep Learning Approaches for Automatic Analysis of EEGs; TBDp. TBDp.
Non-Compliant	Shah V, von Weltin E, Ahsan T, Ziyabari S, Golmohammadi M, Obeid I, Picone J. On the Use of Non-Experts for Generation of High-Quality Annotations of Seizure Events. Journal of Clinical Neurophysiology. Forthcoming. DOI: Not Available.
Non-Compliant	Shah V, Golmohammadi M, Obeid I, Picone J. Objective evaluation metrics for automatic classification of EEG events. IEEE Sensors. Forthcoming. DOI: Not Available.
In Process at NIHMS	Active deep learning for the identification of concepts and relations in electroencephalography reports. Journal of biomedical informatics. DOI: 10.1016/j.jbi.2019.103265.

Non-compliant Publications Previously Reported for this Project

Public Access Compliance	Citation
Non-Compliant	Goodwin T, Harabgiu S. Inferring Clinical Correlations from EEG Reports with Deep Neural Learning. American Medical Informatics Association Annual Symposium (AMIA). 2017 March.
Non-Compliant	Taylor S, Goodwin T, Harabagiu S. An Evaluation of Syntactic Dependency Parsers on Clinical Data. American Medical Informatics Association Annual Symposium (AMIA). 2017 March.
Non-Compliant	Goodwin T, Harabagiu S. Read, Decide and Explain! Recollective (Explanatory) Question Answering. Annual Conference of the Association of Computational Linguistics (ACL). 2017 February.
Non-Compliant	Goodwin T, Harabagiu S. Knowledge Representations and Inference Techniques for Medical Question Answering. ACM Transactions on Intelligent Systems and Technology. 2017 March.
Non-Compliant	Goodwin T, Harabagiu S. Deep Learning from EEG Reports for Inferring Underspecified Information. Proceedings of the American Medical Informatics Association Joint Summits on Clinical Research Informatics (AMIA-CRI). 2017 March.
Non-Compliant	Maldano R, Goodwin T, Harabagiu S. Active Deep Learning-Based Annotation of Electroencephalography Reports for Cohort Identification. Proceedings of the American Medical Informatics Association Joint Summits on Clinical Research Informatics (AMIA-CRI). 2017 March 01.
Non-Compliant	Maldonado R, Goodwin T, Skinner M, Harabagiu S. Deep Learning Meets Biomedical

	Ontologies: Knowledge Embeddings for Epilepsy. American Medical Informatics Association Annual Symposium (AMIA). 2017 March.
Non-Compliant	Goodwin T, Harabagiu S. Multi-Modal Patient Cohort Identification from EEG Report and Signal Data. Proceedings of the American Medical Informatics Association Annual Symposium (AMIA). 2016 November.

C.2 WEBSITE(S) OR OTHER INTERNET SITE(S)

Category	Explanation
Data or Databases	https://www.isip.piconepress.com/projects/tuh_eeg/: A web site devoted to dissemination of data and resources for this project.
Research Material	https://www.isip.piconepress.com/projects/nih_cohort/: A web site devoted to dissemination of information about the project.
Software	https://www.isip.piconepress.com/projects/tuh_eeg/: At this same web site, we also disseminate software related to the project. For example, at this URL: https://www.isip.piconepress.com/projects/tuh_eeg/downloads/nedc_eval_eeg/v1.3.1/ which is a link under the main site, we disseminate a standardized scoring package we developed so that researchers can reproduce our scoring metrics and performance results.

C.3 TECHNOLOGIES OR TECHNIQUES

NOTHING TO REPORT

C.4 INVENTIONS, PATENT APPLICATIONS, AND/OR LICENSES

Have inventions, patent applications and/or licenses resulted from the award during the reporting period? No

If yes, has this information been previously provided to the PHS or to the official responsible for patent matters at the grantee organization? No

C.5 OTHER PRODUCTS AND RESOURCE SHARING

Nothing to report

D. PARTICIPANTS

D.1 WHAT INDIVIDUALS HAVE WORKED ON THE PROJECT?

Commons ID	S/K	Name	Degree(s)	Role	Cal	Aca	Sum	Foreign Org	Country	SS
JOSCONE	Y	Picone, Joseph	MS,PHD	PD/PI	0.0	0.0	1.0			NA
SHARABAGIU	Y	Harabagiu, Sanda Maria	PHD	PD/PI	0.0	0.0	1.0			NA
OBEID07	Y	Obeid, Iyad	BS,MS,PHD	PD/PI	0.0	0.0	1.0			NA
AILOPEZ	N	Lopez, Silvia	BS	Graduate Student (research assistant)	12.0	0.0	0.0			NA
VINSHAH	N	Shah, Vinit	MS	Graduate Student (research assistant)	12.0	0.0	0.0			NA
TRAVISGOODWIN	N	Goodwin, Travis	MS	Graduate Student (research assistant)	12.0	0.0	0.0			NA
STUARTTAYLOR	N	Taylor, Stuart	BS	Graduate Student (research assistant)	12.0	0.0	0.0			NA
RAMONMALDONADO	N	Maldonado, Ramon	BS	Graduate Student (research assistant)	12.0	0.0	0.0			NA
MATIESS	N	Thiess, Matthew	HS	Undergraduate Student	2.0	0.0	0.0			NA
ELLROME	N	Krome, Elliott	HS	Undergraduate Student	2.0	0.0	0.0			NA
DTREJO	N	Trejo, Devin	HS	Undergraduate Student	2.0	0.0	0.0			NA
CHRBELL	N	Campbell, Chris	HS	Undergraduate Student	2.0	0.0	0.0			NA
JAMHUGH	N	McHugh, James	HS	Undergraduate Student	2.0	0.0	0.0			NA
TAMHSAN	N	Ahsan, Tameem	HS	Undergraduate Student	2.0	0.0	0.0			NA
STEWONG	N	Wong, Steven	HS	Undergraduate Student	12.0	0.0	0.0			NA
JASRGEY	N	Bergey, Jason	HS	Undergraduate Student	2.0	0.0	0.0			NA
PATSOMARU	N	Somaru, Pat	HS	Undergraduate Student	2.0	0.0	0.0			NA
NIJHITE	N	White, Nija	BS	Undergraduate Student	1.0	0.0	0.0			NA

NICECCA	N	Mecca, Nicholas	HS	Undergraduate Student	2.0	0.0	0.0			NA
DAWSHED	N	Jamshed, Dawer	PhD	Graduate Student (research assistant)	12.0	0.0	0.0			NA
GOLMOHAMMADI	N	Golmohammadi, Meysam	MS	Graduate Student (research assistant)	12.0	0.0	0.0			NA
SCOTTYANG	N	Yang, Su	PHD	Postdoctoral Scholar, Fellow, or Other Postdoctoral Position	12.0	0.0	0.0			NA
EVALTIN	N	von Weltin, Eva	HS	Undergraduate Student	12.0	0.0	0.0			NA

Glossary of acronyms:
S/K - Senior/Key
DOB - Date of Birth
Cal - Person Months (Calendar)
Aca - Person Months (Academic)
Sum - Person Months (Summer)
Foreign Org - Foreign Organization Affiliation
SS - Supplement Support
RE - Reentry Supplement
DI - Diversity Supplement
OT - Other
NA - Not Applicable

D.2 PERSONNEL UPDATES

D.2.a Level of Effort

Not Applicable

D.2.b New Senior/Key Personnel

Not Applicable

D.2.c Changes in Other Support

Not Applicable

D.2.d New Other Significant Contributors

Not Applicable

D.2.e Multi-PI (MPI) Leadership Plan

Not Applicable

E. IMPACT

E.1 WHAT IS THE IMPACT ON THE DEVELOPMENT OF HUMAN RESOURCES?

Not Applicable

E.2 WHAT IS THE IMPACT ON PHYSICAL, INSTITUTIONAL, OR INFORMATION RESOURCES THAT FORM INFRASTRUCTURE?

We are now collecting and organizing EEG data for Temple Hospital. For the first time they can easily access archives of EEG data. This is proving to be enormously valuable to their clinical operations.

E.3 WHAT IS THE IMPACT ON TECHNOLOGY TRANSFER?

Not Applicable

E.4 WHAT DOLLAR AMOUNT OF THE AWARD'S BUDGET IS BEING SPENT IN FOREIGN COUNTRY(IES)?

NOTHING TO REPORT

G. SPECIAL REPORTING REQUIREMENTS

G.1 SPECIAL NOTICE OF AWARD TERMS AND FUNDING OPPORTUNITIES ANNOUNCEMENT REPORTING REQUIREMENTS

NOTHING TO REPORT

G.2 RESPONSIBLE CONDUCT OF RESEARCH

Not Applicable

G.3 MENTOR'S REPORT OR SPONSOR COMMENTS

Not Applicable

G.4 HUMAN SUBJECTS

Not Applicable

G.5 HUMAN SUBJECTS EDUCATION REQUIREMENT

Not Applicable

G.6 HUMAN EMBRYONIC STEM CELLS (HESCS)

Does this project involve human embryonic stem cells (only hESC lines listed as approved in the NIH Registry may be used in NIH funded research)?

No

G.7 VERTEBRATE ANIMALS

Not Applicable

G.8 PROJECT/PERFORMANCE SITES

Not Applicable

G.9 FOREIGN COMPONENT

No foreign component

G.10 ESTIMATED UNOBLIGATED BALANCE

Not Applicable

G.11 PROGRAM INCOME

Not Applicable

G.12 F&A COSTS

Not Applicable

I. OUTCOMES

I.1 What were the outcomes of the award?

There are three major impacts from this project: (1) The data and resources generated from this project are being accessed by over 2,500 subscribers on a regular basis. The EEG Corpus released is being used to advance technology development using deep learning techniques. (2) The EEG event technology has demonstrated the ability to detect clinically important life-altering events such as seizures with state-of-the-art accuracy and as little as 7 seconds of latency. The entire TUH EEG Corpus has been released with automically generated annotations using this technology. (3) The Cohort Retrieval system integrates signal and text information to support natural language queries. A demonstration system was developed that allows user to submit queries on the TUH EEG Corpus. This technology was publicly demonstrated at several conferences in 2018 and 2019.

The natural language processing component of the system, known as Mercury, automatically recognizes critical concepts in an EMR: (1) EEG activities and their attributes, (2) EEG events, (3) medical problems, (4) medical treatments and (5) medical tests mentioned in the narratives of the EEG reports, along with their inferred forms of modality and polarity. This allows unstructured EEG reports to be searched using natural language queries. This capability, combined with the ability to index events in the actual EEG signal, can serve as a clinical decision support tool. For example, we used these technologies to allow Temple Hospital neurologists to identify candidate patients for a drug trial.