Predicting Mental Conditions Based on "History of Present Illness" in Psychiatric Notes with Deep Neural Networks

Tung Tran[a], Ramakanth Kavuluru[a,b,*]

[a]*Department of Computer Science, University of Kentucky, 329 Rose Street, Lexington, KY 40506, USA*
[b]*Division of Biomedical Informatics, Department of Internal Medicine, University Kentucky, 725 Rose Street, Lexington, KY 40536, USA*

Abstract

Background: Applications of natural language processing to mental health notes are not common given the sensitive nature of the associated narratives. The CEGS N-GRID 2016 Shared Task in Clinical Natural Language Processing (NLP) changed this scenario by providing the first set of neuropsychiatric notes to participants. This study summarizes our efforts and results in proposing a novel data use case for this dataset as part of the third track in this shared task.

Objective: We explore the feasibility and effectiveness of predicting a set of common mental conditions a patient has based on the short textual description of patient's history of present illness typically occurring in the beginning of a psychiatric initial evaluation note.

Materials and Methods: We clean and process the 1000 records made available through the N-GRID clinical NLP task into a key-value dictionary and build a dataset of 986 examples for which there is a narrative for history of present illness as well as Yes/No responses with regards to presence of specific mental conditions. We propose two independent deep neural network models: one based on convolutional neural networks (CNN) and another based on recurrent neural networks with hierarchical attention (ReHAN), the latter of which allows for interpretation of model decisions. We conduct experiments to compare these methods to each other and to baselines based on linear models and named entity recognition (NER).

Results: Our CNN model with optimized thresholding of output probability estimates achieves best overall mean micro-F score of 63.144% for 11 common mental conditions with statistically significant gains ($p < 0.05$) over all other models. The ReHAN model with interpretable attention mechanism scored 61.904% mean micro-F1 score. Both models' improvements over baseline models (support vector machines and NER) are statistically significant. The ReHAN model additionally aids in interpretation of the results by surfacing important words and sentences that lead to a particular prediction for each instance.

*Corresponding author
Email addresses: tung.tran@uky.edu (Tung Tran), ramakanth.kavuluru@uky.edu (Ramakanth Kavuluru)

Conclusions: Although the history of present illness is a short text segment averaging 300 words, it is a good predictor for a few conditions such as anxiety, depression, panic disorder, and attention deficit hyperactivity disorder. Proposed CNN and RNN models outperform baseline approaches and complement each other when evaluating on a per-label basis.

Keywords: Psychiatric condition prediction, multi-label text classification, convolutional and recurrent neural networks, and hierarchical attention networks

---

1. Introduction

According to the 2014 National Survey on Drug Use and Health, the National Institute of Mental Health reports [28] that one in five adults suffer from a mental illness in a given year. A February 2011 Robert Wood Johnson Foundation research synthesis report [11] presents evidence that the subgroup of people with both mental and medical disorder comorbidities are at significant risk for poor quality of care and high costs. Given this, there has been major emphasis on identifying connections between mental disorders such as depression and anxiety disorders that have high prevalence and other chronic medical conditions including cancer, diabetes, and heart disease. Also, most of these analyses have generally focused on structured datasets even when natural language processing (NLP) techniques are being extensively used to derive insights from clinical notes for many chronic medical conditions. Overall, applying NLP techniques to assess mental health disorders has been a largely unexplored problem space. The 2016 Centers of Excellence in Genomic Science (CEGS) Neuropsychiatric Genome-Scale and RDoC Individualized Domains (N-GRID) NLP challenge proposed the first open competition to address this gap. As part of the challenge, a dataset of 1000 neuropsychiatric notes, which constitutes the first of its kind, was released to the participants.

Novel data use track. Track three of the N-GRID challenge explores research questions and novel use cases of the released dataset and is the primary focus of this paper while the first two tasks focus on de-identification and symptom severity score prediction. Specifically, we propose and demonstrate the application of deep neural networks in predicting individual patient mental conditions based on the short *history of present illness* text field of the corresponding note. For details about the organizational aspects of the shared task including data collection, annotation, and track objectives, please refer to the corresponding overview papers [12, 40].

Our novel data use-case. The notes provided for the challenge are rich in different types of information including demographic variables, histories of violent behavior, substance use, risk factors, and a treatment plan. Besides these, two additional fields that are consistently available and represented in a uniform manner are

1. "History of present illness and precipitating events": Averaging 300 words over all notes, this short text segment appears as one of the first few headings in a typical note describing the initial assessment and observations made by the psychiatrist.

2. "Psychiatric review of systems": This section of the note has a high level structure and is composed of a set of questions about the presence of 13 different mental conditions and the corresponding Boolean assessments of the psychiatrist.

Our novel data use case is to predict the presence of these mental conditions from the second field above solely based on the short text narrative from the first field on history of present illness. That is, the ground truth for the mental conditions' presence that we aim to predict is based on Yes/No answers to corresponding questions in the psychiatric review portion of the note. We believe a model capable of making such predictions with reasonable accuracy has several real-world applications. For one, it would make it possible for physicians and other healthcare professionals to make quick assessments, based on a relatively small narrative, that could lead to early hints of a mental disorder. It can also assist psychiatrists in filling out the corresponding structured fields when needed. Furthermore, such a model would make it possible to perform automated surveillance of a patient's ongoing mental condition, which can further be accomplished in a large-scale fashion over multiple databases for which patient notes are available. In practice, for this shared task, the psychiatric review fields are a good choice because unlike most other fields, as indicated earlier, they are consistently present in a large majority of notes made available. Given multiple conditions can be present for each case, we map the core prediction problem to a *multi-label text classification* instance and solve it using conditional models including deep neural networks. Next, we outline the organization of this manuscript.

In Section 2, we give further details about the dataset including specifics of different target labels predicted and some preliminary analysis of label correlation. We discuss our main methods involving deep neural networks in Section 3. In Section 4, we present the experimental setup including model configurations and evaluation measures. We then discuss our results and conduct extensive qualitative error analysis in Section 5.

2. Dataset: Labels and their Associations

Under the *psychiatric review of systems* heading of each note, there are questions pertaining to the presence of these thirteen conditions: depression, bipolar disorder, psychosis, general anxiety disorder (GAD), panic disorder, anxiety spectrum disorders, obsessive compulsive disorders (OCD), obsessive compulsive spectrum disorder (OCSD), attention deficit hyperactivity disorder (ADHD), post traumatic stress disorder (PTSD), eating disorders, dementia, and complicated grief. The answers are Boolean Yes/No responses and unambiguously indicate the presence/absence of a condition. For dementia, for example, we have the question: "Dementia: Has anyone told the patient they are concerned the patient has memory problems?" Some conditions are fine-grained in that they distinguish between variants of a particular disorder; e.g. for dementia, there exists a separate question (and an associated Boolean response label) concerning difficulty learning new information – Does the patient have trouble learning new information? To minimize label imbalance issues and for simplicity, we collapsed such couplings into a single label whose value is a *yes* if either of the sub-labels (i.e., responses) is positive. We also combined OCD with OCSD and GAD with anxiety spectrum disorders into single labels. So there are a total of 11 labels which we predict based on the text from the history of present illness field. The final list set of labels and their corresponding distribution in the dataset are displayed in Table 1. Given a note can have multiple labels assigned to it based on Yes responses to the corresponding condition related questions, we note that the proportions do not add up to 100% in the table.

Condition	Label Occurrence Proportion
ADHD	41%
Anxiety	68%
Bipolar	33%
Dementia	27%
Depression	77%
Eating Disorder	31%
Grief	27%
OCD/OCSD	34%
Panic	47%
Psychosis	25%
PTSD	38%

Table 1: Distribution of labels in the N-GRID dataset

	ADHD	Anxiety	Bipolar	Dementia	Depression	Eating Disorder	Grief	OCD/OCSD	Panic	Psychosis	PTSD
ADHD	-	3.2	4.6	5.7	2.2	5.7	5.3	4.9	3.7	6.0	4.9
Anxiety	3.2	-	2.2	3.0	2.6	3.9	3.3	6.1	7.0	2.7	3.1
Bipolar	4.6	2.2	-	3.5	5.7	3.9	3.4	3.6	3.3	11.6	4.9
Dementia	5.7	3.0	3.5	-	2.1	9.8	18.4	6.6	3.5	7.2	5.7
Depression	2.2	2.6	5.7	2.1	-	3.5	2.9	2.4	3.0	3.9	2.7
Eating Disorder	5.7	3.9	3.9	9.8	3.5	-	11.4	7.7	3.8	5.5	6.8
Grief	5.3	3.3	3.4	18.4	2.9	11.4	-	8.0	4.3	5.7	7.5
OCD/OCSD	4.9	6.1	3.6	6.6	2.4	7.7	8.0	-	4.9	6.3	5.9
Panic	3.7	7.0	3.3	3.5	3.0	3.8	4.3	4.9	-	4.3	4.8
Psychosis	6.0	2.7	11.6	7.2	3.9	5.5	5.7	6.3	4.3	-	6.2
PTSD	4.9	3.1	4.9	5.7	2.7	6.8	7.5	5.9	4.8	6.2	-

Table 2: Odds ratios of exposure between mental conditions

For this effort, we build a dataset composed of 986 of the total 1000 released notes for each of which we have the history of present illness section as well as Yes/No labels in the psychiatric review of systems. As a pre-processing step, we fixed a few formatting errors in the text such as in cases when line-breaks are missing in appropriate places or when present in inappropriate places. Next, we generated a key-value pair dictionary from each note by matching text segments with certain regular expressions, which were based on our manual observation of the note structure. In this process, we also accounted for other concerns including some frequent spelling mistakes and structural inconsistencies. Although we could have missed some fields, given this particular approach is based on few fields that are almost always present and written up in a consistent manner in the notes, we believe this regex based pre-processing strategy is effective for our purposes.

We computed the odds ratios (OR) of pair-wise labels and present them in Table 2; this can be interpreted as a measure of how strongly two labels are associated. An OR of 1 implies that there is no association, while OR < 1 implies that a condition is less likely to be present when the other is positive and OR > 1 signals that presence of a condition makes it more likely that the other occurs. From the table, it is clear that there is a positive correlation among all label pairs, with complicated grief and dementia having an exceptionally high correlation. This is an indication that multi-label classification methods that exploit label correlations might be more effective than those that treat each label independently.

3. Methods: Deep Neural Networks for Multi-Label Text Classification

Predicting the binary presence (Yes/No) of mental conditions based on the history of present illness field can be framed as a multi-label text classification problem where an input document needs to be assigned one or more categories from a fixed set [42]. Well known examples in biomedicine include assigning diagnosis codes to EMRs [21] and indexing biomedical articles with medical subject headings [20]. If there are m labels, traditionally this problem is solved by using the binary relevance approach – we form m datasets, one per label, of positive and negative examples from the original dataset. Here, an instance is considered a negative example if it is not assigned the label at hand even if it is assigned other labels. Next, m binary classifiers are built one per label and at test time, the labels corresponding to the classifiers that output a positive prediction are assigned to the test instance. Researchers typically use a linear conditional model such as support vector machines (SVMs) for text classification for each of the base models in the binary relevance approach. Such an approach has been successful but does not account for label correlations and might not be the best approach when associations exist. We nevertheless explore conventional approaches in our experiments for comparison purposes.

Beyond the traditional approach, there has been notable advancement in the realm of text classification by using a deep neural network architecture such as convolutional neural networks (CNNs) in conjunction with neural word embeddings [37]. CNNs were originally intended and motivated to replicate the visual perception of humans and animals and have experienced success in image recognition tasks [23]. A powerful aspect of CNNs is translational invariance, which allows them to detect unique contextual features regardless of where they appear in the field of vision. This along with the inclusion of the so called pooling operation (more later) makes it possible for CNNs to deal with variable-length inputs

such as text data. Using CNNs along with neural word embeddings has been shown to be effective in many NLP tasks (including text classification and relation extraction) since they additionally capture syntactic and semantic information [5, 9, 25]. Unlike CNNs, which are a feedforward type of network, recurrent neural networks (RNNs) have been successful in sequence labeling tasks such as part-of-speech tagging, named entity recognition (NER), and machine translation [4, 18] due to their ability to handle arbitrary-length sequential input via cyclical connections and some form of internal memory. In the main methods we propose in this section, predictions are made on all 11 psychiatric labels simultaneously using a single deep neural network model which has the advantage of accounting for label correlation to some extent – this is conceptually similar to multi-task learning [8] given the task of predicting each label is closely related. This setup is different from the binary relevance approach where correlation is ignored altogether.

Next, we introduce two different deep learning based methods that form our core methodology to address the problem at hand. In Section 3.1, we present a CNN-based model based on a prior approach for text classification as introduced by Yoon Kim [22] and later adapted by Rios and Kavuluru [37] for biomedical text classification. We adapt these prior efforts suitable for a multi-class (needing selection of exactly one class) scenario to the current multi-label situation. CNN models, while exceptional in performance, are not easily interpretable. To aid in interpretability, in Section 3.2, we introduce an alternative RNN-based approach that uses hierarchical *attention* mechanism [43]; the advantage being that such a network is able to learn word-level and sentence-level *softmax* weights which can be visualized and interpreted. We call this the recurrent hierarchical attention network (ReHAN) model.

Neural word embeddings. Both our approaches are based on using neural word embeddings, a setup that has been shown to be effective for learning tasks in NLP [10]. Word embeddings (e.g., those generated by Google Inc.'s Word2Vec program) are dense vector representations that have been shown to capture both semantic and syntactic information. A few recent approaches learn word vectors [5, 9, 25] (as elements of $\mathbb{R}^d$, where d is the dimension) in an unsupervised fashion from textual corpora. These dense word vectors obviate the sparsity issues inherent to the so called *one-hot* representations of words that lead to very large dimensionality (typically the size of the vocabulary) resulting in further issues in similarity computations. Before we proceed ahead, we note that the rest of this section focuses on main foundations of deep learning architectures. The detailed experimental setup including various hyperparameter settings and model configuration aspects are described in Section 4.

3.1. A CNN Model for Multi-Label Learning

We propose using a deep neural network architecture based on convolutional neural networks from our prior work [37] modified to suit this task. The full CNN architecture is shown in Figure 1. The input is a document with words $\mathbf{w} = (w_1, w_2, .., w_n)$ each represented by their corresponding index to the vocabulary V. The words are mapped to word vectors via an embedding matrix $E \in \mathbb{R}^{|V| \times d}$ to produce a document matrix $D \in \mathbb{R}^{n \times d}$ where d is the

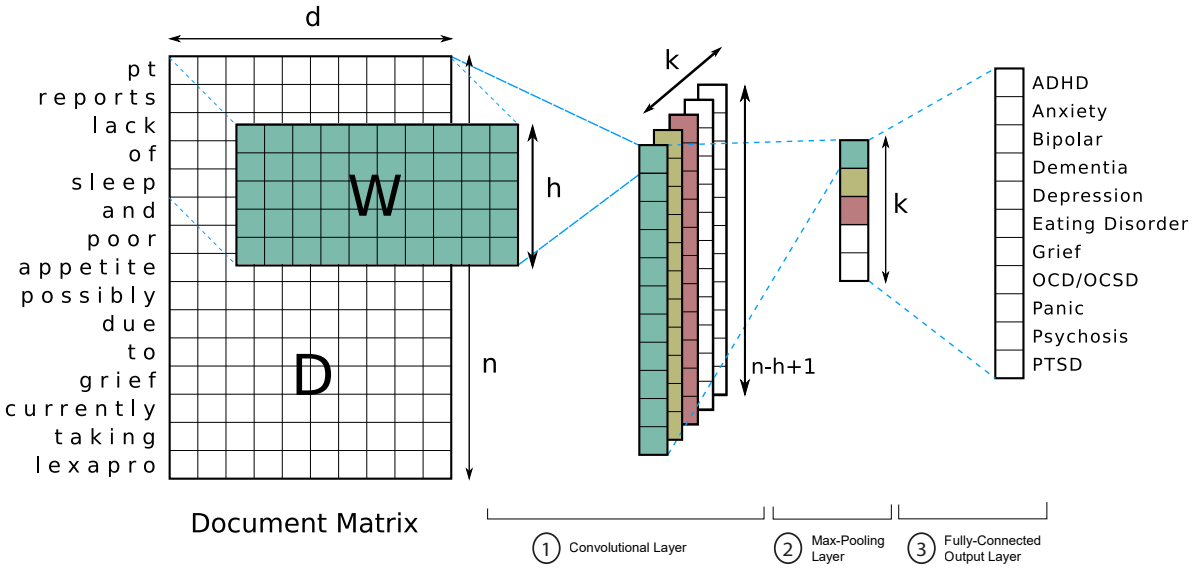


Figure 1: CNN model architecture for multi-label text classification

dimension of the word representation vectors. More concisely,

$$D = \begin{pmatrix} E_{w_1} \\ E_{w_2} \\ \vdots \\ E_{w_n} \end{pmatrix}$$

where E_i is the i^{th} row of E. The word embedding matrix can be initialized to random or pretrained values using methods identified in the introduction of this section; in either case, the word vectors are (further) modified via backward propagation. The central idea in CNNs is the so called *convolution* operation over the document matrix to produce a feature map representation using a *convolution filter* (CF). The convolution operation $*$ is formally defined as the sum of the element-wise products of two matrices. That is, for two matrices A and B of same dimensions, $A * B = \sum_j \sum_k A_{j,k} \cdot B_{j,k}$. With this, a CF is the matrix $W \in \mathbb{R}^{h \times d}$ that is applied as a convolution to a window of size h over D to produce a feature map $\mathbf{v} = [v_1, \ldots, v_{n-h+1}]$, such that

$$v_i = f(W * D_{i:i+h-1} + b)$$

where $D_{i:i+h-1}$ is a window of matrix D spanning from row i to row $i+h-1$, W and $b \in \mathbb{R}$ are learned parameters, and f is a non-linear activation function such as the sigmoid or hyperbolic tangent function. The goal is to learn multiple CF that can collectively capture diverse representations of the same document. Suppose there are k filters, then we produce k corresponding feature maps $\mathbf{v}^1, \ldots, \mathbf{v}^k$. We select the most distinctive feature of each feature map using a max-over-time pooling operation [10] to produce the final feature vector $\hat{\mathbf{p}} \in \mathbb{R}^k$, such that $\hat{\mathbf{p}} = [v^1_{max}, \ldots, v^k_{max}]$ where $v^j_{max} = \max(\mathbf{v}^j_1, \ldots, \mathbf{v}^j_{n-h+1})$.

We can also learn different sets of k CFs for different window sizes h as is typically the practice. Choosing a larger h provides more context and thus could be beneficial in improving predictive power but might adversely affect efficiency given the additional time needed. We can then take the corresponding feature vector for each window size and concatenate them to form the final feature vector. More formally, we can parameterize the window sizes as a sequence $h_1, \ldots, h_H$ of H unique sizes. Suppose $\hat{\mathbf{p}}^{h_i}$ denotes the feature vector produced on k filters with a window size of h_i, then the final $kH \times 1$ feature vector is

$$\hat{\mathbf{p}}^* = \hat{\mathbf{p}}^{h_1} \parallel \cdots \parallel \hat{\mathbf{p}}^{h_H}$$

where $\parallel$ is the vector concatenation operation. The details covered thus far correspond to components ① and ② of Figure 1.

The output layer consists of m sigmoid units (one per each of the m target labels) and is fully connected to the full feature vector $\hat{\mathbf{p}}^*$. The output vector $\mathbf{q} \in \mathbb{R}^m$ is thus defined as

$$\mathbf{q} = \sigma(W_q \hat{\mathbf{p}}^* + b_q) \tag{1}$$

where $W_q \in \mathbb{R}^{m \times kH}$ is a parameter matrix of the fully connected layer mapping feature vectors to output layer, $b_q \in \mathbb{R}^m$ is the vector of bias terms, and $\sigma(x)$ is the sigmoid function. This forms component ③ of Figure 1.

During training, we optimize the network parameters by minimizing the binary cross entropy loss function [27]

$$-\frac{1}{L} \sum_{i=1}^{L} \left(\sum_{j=1}^{m} y_j^i \log(q_j^i) + (1 - y_j^i) \log(1 - q_j^i) \right) \tag{2}$$

where y_j^i are the ground truth 0/1 values and q_j^i are model output values for the j-th label and i-th instance, and L is the number of training examples. Each sigmoid unit's $[0, 1]$ output is the probability estimate on which predictions are made for the corresponding label. That is, an output greater than 0.5 results in a positive prediction for the label. Thus, the final set of labels determined as such becomes the predicted set of conditions for the patient.

The network is trained using stochastic gradient descent (SGD) using mini-batches [30] approach, in which each training iteration uses only a small sample of the training data. Multiple epochs or "passes" over the training data are usually necessary to obtain a good fit. The model is prone to overfitting as the number of training epochs increases; in order to combat this, we apply the now popular dropout [39] regularization to the feature vector layer during the training phase. Given this has been a standard process in deep neural networks, we request readers to refer to our prior work [37, Section 3.1] for more specifics on the intuition and formal description of this regularization approach and the associated dropout parameter.

3.2. Interpretable Recurrent Hierarchical Attention Networks (ReHANs)

The CNN model from Section 3.1 is effective but not suitable if interpretability is a desired feature. Hence, we introduce an alternative model architecture using RNNs in combination with Hierarchical Attention Networks (HANs), henceforth called the ReHAN approach, that

performs competitively with interpretable predictions. The general model we present here is based on the architecture by Yang et al. [43], which allows for observation of the contributional weights of words and sentences in a document toward the eventual prediction using two levels of attention mechanisms [4]. We start out with a general introduction to RNNs.

3.2.1. RNNs, BiRNNs, and LSTMs

Unlike feedforward networks like CNNs and multi-layer perceptrons, RNNs have cyclical connections and are more suitable for language processing tasks where the meaning of a text segment is naturally dependent on what occurred in the narrative before it. This aligns closely with how we process language where the interpretation of a word is dependent on what occurred before it in the document. This recurrent composition of word vectors effectively lets information persist from the history of previously seen words. There is typically an input layer, a hidden layer that is connected to itself, and an output layer. The hidden layer's output is fed back to itself at consecutive time steps (generally as many times as there are words in the narrative) and the output at any time step is generally the recurrent composition of information until that point. Parameter optimization is implemented through the so called *back propagation through time* because of the "unfolding" of the cyclical connections in the hidden layer through different time steps. For a thorough treatment of RNNs, we encourage the reader to refer to a popular resource by Graves [15, Chapter 3].

In the context of RNNs for NLP, the input at each time step is the vector corresponding to the next word in the narrative. The output is the context vector that composes word vectors that include all previous words and itself using the RNN architecture. Additional details of RNNs for NLP applications are available in the detailed primer by Goldberg [14]. The final prediction for text classification can be made based on the output at the final time step or using some combination of all outputs generated at each step (more in Section 3.2.2).

Bidirectional RNNs. In addition to the default left to right processing of a document, it has been shown that running the RNN from right to left over the input text can yield additional contextual hints for eventual prediction tasks. This aids in exploiting signals that come from the future in interpreting the current word. This is not uncommon in NLP tasks where presence of passive voice and other language constructs have valuable information pertaining to the context of a word coming later in the text. This gave rise to bi-directional RNNs (BiRNNs) which essentially have two separate RNNs, each with its own parameters, capturing the context at each position from both directions. The output at each time step is a combination of output vectors from both RNNs typically produced via concatenation.

Long short-term memory. A significant issue with traditional RNNs is the problem of *vanishing gradients* [31] where the back propagated errors that are needed to update the parameters become extremely small for earlier layers (in the cyclical layer unfolding) due to the application of the familiar chain rule in computing derivatives of expressions involving functions of functions. Because of this, learning becomes extremely slow and may be ineffective overall. This effect increases the deeper the network is and hence is an issue for RNNs given the unfolded cyclical connections are as deep as the lengths of sentences. To counter this in RNNs, one popular idea is to use a more involved hidden layer with the so called *long short-term memory* (LSTM) units [13, 16]. Unlike in a traditional RNN, in LSTMs, the state representation includes an explicit *memory cell* access to and use of which is controlled through three

gates – first to control how much of the next input to incorporate in the memory (input gate), second to determine to what extent the current memory is to be forgotten (forget gate), and third to limit the extent of information from the current memory cell to propagate to the output state (output gate). These three gates control the flow of information based on the previous output and cell state via sigmoid outputs $\in [0, 1]$. We encourage readers to refer to Graves [15, Chapter 4] and Goldberg [14, Section 11] for thorough details of LSTMs and the corresponding derivations of gradients. In this paper, we used BiRNNs with LSTM units (simply termed BiLSTMs) in the hidden layer as the main neural architecture augmented by attention mechanism.

3.2.2. BiLSTM based ReHANs with Word and Sentence Level Attention

Interpretability is a major issue for nonlinear predictive models, especially for deep neural networks, where it is traded-off for better performance. As such, many recent efforts are focusing on deriving interpretable insights from neural models for NLP tasks. Although there exist methods that visualize and analyze the inner workings at different network layers and in different dimensions [24], high level insights can be derived from *attention* mechanisms. The intuition behind attention based classification in deep learning also arises from how we process language. Specifically, in classifying a document, human assessors also determine that certain segments are more informative/contributive toward the eventual decision than others. In fact, the N-GRID clinical NLP challenge's main task of predicting RDoC positive valence symptom severity scores is introduced in a document where the organizers highlight portions of a sample narrative that lead the experts to classify it as a SEVERE case. The attention mechanism essentially learns these informativeness weights as part of the overall prediction task when the BiLSTM network is augmented in a specific manner. Yang et al. [43] offer the first hierarchical attention framework for text classification by exploiting such inherent structure – words are composed to form sentences and sentences in turn form the document. We implemented their method (originally used for sentiment classification) to our current task of multi-label classification. The hierarchical attention architecture ReHAN is outlined in Figure 2.

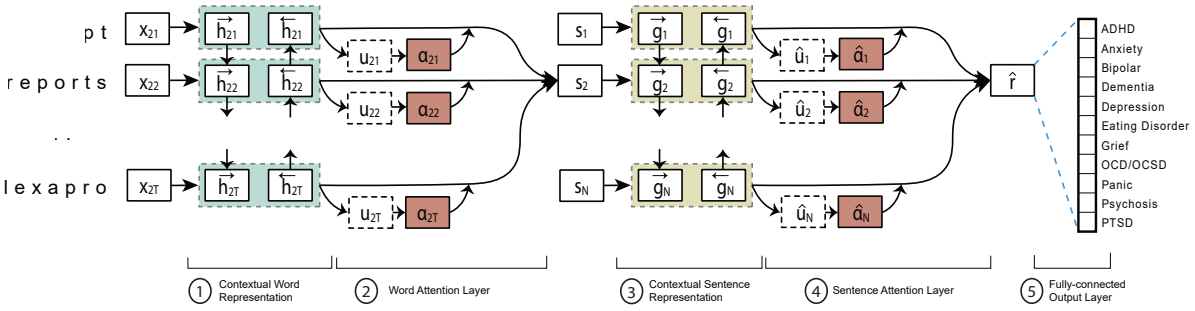


Figure 2: BiLSTM hierarchical attention network architecture for multi-label classification

Word-level attention. Let $w_{i,t}$ denote the t-th word of the i-th sentence. For simplicity, we assume the length of a sentence is T words. It is important to have fixed sentence size

given the attention mechanism learns custom weights per word position. Typically, this is accomplished by choosing T to be the length of the longest sentence in training data and padding a special *blank word* vector for small sentences and ignoring words after the T-th word for longer sentences encountered at test time. The blank word vector is treated like any other word vector and is updated during the training phase. Each word is mapped to a word vector via an embedding matrix E such that $x_{i,t} = E_{w_{i,t}}$ as in the case of CNNs. The input is passed through a recurrent layer composed of bi-directional LSTM units, i.e., iterating both in the forward and in the backward direction for the input sequence. In order to produce a feature vector for each word in the sequence that captures contextual information in both directions, we concatenate the outputs of the forward and backward word level LSTM (or WLSTM) units at each corresponding word position:

$$\overrightarrow{h}_{i,t} = WLSTM^{\rightarrow}(x_{i,t}), \overleftarrow{h}_{i,t} = WLSTM^{\leftarrow}(x_{i,t}), \text{ and } \mathbf{h}_{i,t} = \overrightarrow{h}_{i,t} \parallel \overleftarrow{h}_{i,t}, \text{ for } t = 1, \ldots, T,$$

where $\overrightarrow{h}_{i,t}, \overleftarrow{h}_{i,t} \in \mathbb{R}^{k_h}$ (the forward or backward LSTM is indicated based on the arrow direction for the corresponding symbols), $\mathbf{h}_{i,t} \in \mathbb{R}^{2k_h}$ is the concatenated output from both LSTMs, and vector length k_h is a hyperparameter specific to these LSTM units. Next, we outfit an attention mechanism layer on top of the contextualized word features as to produce a softmax weight $\alpha_{i,t}$ for each word in the sequence. This is achieved by first producing hidden feature vectors $u_{i,t}$ of length k_u (another hyperparameter) using the equation

$$\mathbf{u}_{i,t} = \tanh(W^{word} \cdot \mathbf{h}_{i,t} + b^{word}), \tag{3}$$

where $W^{word} \in \mathbb{R}^{k_u \times 2k_h}$ and $b^{word} \in \mathbb{R}^{k_u}$ are parameters. We then learn the per word attention weights $\alpha_{i,t}$ via a learnable context position vector $\mathbf{a}^w \in \mathbb{R}^{k_u}$ as

$$\alpha_{i,t} = \frac{\exp(\mathbf{u}_{i,t}^{\top} \cdot \mathbf{a}^w)}{\sum_t \exp(\mathbf{u}_{i,t}^{\top} \cdot \mathbf{a}^w)}. \tag{4}$$

The $\alpha_{i,t}$ weights are used as scalar factors to the original word-wise context vectors such that a sentence representation $\mathbf{s}_i \in \mathbb{R}^{2k_h}$ can be obtained as a weighted average:

$$\mathbf{s}_i = \sum_t \alpha_{i,t} \mathbf{h}_{i,t}. \tag{5}$$

The word level LSTMs and the corresponding attention structure correspond to components ① and ② of Figure 2.

Sentence-level attention. We now apply the same attention mechanism but at the sentence level using sentence vectors s_i, $i = 1, \ldots, N$, where N is the fixed number of sentences per document chosen to be the maximum such value over the training dataset with additional *blank vector* padding as outlined for word level attention. We can produce contextual sentence vectors by feeding the sentence embeddings through a bidirectional sentence level LSTM (SLSTM) layer as follows:

$$\overrightarrow{g}_i = SLSTM^{\rightarrow}(s_i), \overleftarrow{g}_i = SLSTM^{\leftarrow}(s_i), \text{ and } \mathbf{g}_i = \overrightarrow{g}_i \parallel \overleftarrow{g}_i, \text{ for } i = 1, \ldots, N,$$

where $\mathbf{g}_i \in \mathbb{R}^{2k_h}$ is the contextual sentence vector for the i-th sentence. We again fit an attention network at the sentence level, producing a final vector $\hat{\mathbf{r}} \in \mathbb{R}^{2k_h}$ that represents the full document as follows

$$\hat{\mathbf{u}}_i = \tanh(W^{sentence} \cdot \mathbf{g}_i + b^{sentence}),\ \hat{\alpha}_i = \frac{\exp(\hat{\mathbf{u}}_i^\top \cdot \mathbf{a}^s)}{\sum_i \exp(\hat{\mathbf{u}}_i^\top \cdot \mathbf{a}^s)},\ \text{and}\ \hat{\mathbf{r}} = \sum_i \hat{\alpha}_i \mathbf{g}_i,$$

where the formulation is similar to that for word level attention in eqs. (3) to (5), albeit with different parameters. The sentence level LSTMs and the corresponding attention structure correspond to components ③ and ④ of our full model in Figure 2.

Just as in eq. (1) in Section 3.1 for CNNs, the m sigmoid outputs are determined by

$$\hat{\mathbf{q}} = \sigma(W_{\hat{q}} \cdot \hat{\mathbf{r}} + b_{\hat{q}})$$

where $W_{\hat{q}} \in \mathbb{R}^{m \times 2k_h}$ and $b_{\hat{q}} \in \mathbb{R}^m$ are parameters. We optimize on the same binary cross entropy loss introduced in eq. (2). Dropout regularization is applied at the hidden feature layer for both word and sentence-level attention. This forms the final component of Figure 2.

4. Experimental Setup

In this section, we describe specific details of different experiments we conducted including baseline methods, model configurations, and evaluation measures. The CNN model was built using the Theano library [7] and the ReHAN model was implemented in the TensorFlow framework [1].

4.1. Baselines

Given the present illness text field may already contain direct mentions of various psychiatric conditions, running a named entity recognition (NER) and concept mapping tool on that field is an important baseline for our experiments. We first manually curated a set of related named entities (Concept Unique Identifiers (CUIs)) for each target label using the UMLS Metathesaurus [29] (2016AA dataset) as a reference by browsing through NLM's online interface. Let this set be K_c for label c where $|K_c|$ is 26 on average based on our curation. All such curated CUIs for each condition are presented in a supplementary file for this paper. For each instance i, we ran NLM's MetaMap [3] concept mapping tool and thus extracted UMLS concepts M_i from the corresponding text field. We configured the tool to run on strict mode using the 2016AA dataset with word sense disambiguation enabled. Next, we predicted label c for instance i if and only if $|K_c \cap M_i| > 0$.

We also ran our experiments with a straightforward linear support vector machine (SVM) based binary model, training one model per label, based on uni/bi-gram features extracted from the narrative.

4.2. CNN Model Configuration

As a reminder, both of our neural network models rely on word vectors to form the input document matrix. Given our prior experiences with obtaining superior results with using pre-trained word vectors [37] as opposed to randomly initialized vectors, we used those published by Pyysalo et al. [34] with a dimensionality of 200 induced from PubMed abstracts with

the *word2vec* [25] program. In neural network learning, it is common practice to initialize parameters with relatively small non-zero values [15, Chapter 3.3] to break symmetry and facilitate the learning process [19]. Hence non-word vector parameters were heuristically initialized to a random value in $[-0.15, 0.15]$. We use $k = 250$ filters for each window size of three, four, and five adjacent tokens. For the non-linear activation function, we use the recommended rectified linear unit [26]. We set the dropout regularization hyperparameter $p = 0.5$ for the training process. We trained 25 epochs with a mini-batch size of five instances. Parameter states are check-pointed on each epoch and the parameter state with the best micro-F1 on the validation set is used for evaluation on the test set. The models are trained using the RMSProp [41] optimizer, an extension of SGD, with a learning rate of 0.001. We train ten such CNN models with different random parameter initializations as part of an ensemble and predictions are made by averaging the probability estimates output over all models. Next, we outline two additional CNN configurations that involve post-processing the basic CNN model's per-class outputs.

CNN with meta-labeler. In this variant, we extend the CNN model with a *meta-labeler* component. We rank labels based on CNN output scores and select the top $\bar{k}$ labels (regardless of whether the corresponding sigmoid output is > 0.5) as the final predictions. $\bar{k}$ is a hyper parameter predicted for each instance based on a linear regression model built using uni/bi-gram features. The intuition is that the narrative might also be informative of the number of labels to be chosen. This could be important to pick up infrequent labels whose sigmoid units may not fire often.

CNN with optimized threshold. This is similar to the meta-labeler approach but instead of selecting the top few labels, we choose customized sigmoid unit output thresholds for each label separately based on the validation fold. That is, we choose a hyperparameter threshold $\bar{t}_j \in [0, 1]$ for each label $j = 1, \ldots, m$, such that we predict the label as true if and only if $q_j > \bar{t}_j$ where q_j is the sigmoid unit output for the j-th label. The thresholds are learned on a validation fold in each cross validation iteration. Both this approach and the meta-labeler approach are popular in literature and have resulted in performance gains in our earlier efforts in multi-label classification [21, 35, 36].

4.3. ReHAN Model Configuration

We used the same pre-trained word vectors as in Section 4.2 and similarly initialize other network parameters to a random value in $[-0.15, 0.15]$ for the ReHAN architecture in Section 3.2.2. Words that occur less than five times in the dataset were are not only ignored in order to reduce vocabulary space (as in Yang et al. [43]) but discarded altogether to additionally reduce maximum sequence length and hence overall training time. We introduce another form of noise similar to dropout by randomly removing words from the sequence and replacing it with the *blank word* token during training. In doing so, we force the model to cope with a distinctively noisy version of the training set on each epoch. We find that introducing this noise by nullifying words at a probability of 0.25 along with a dropout hyperparameter of $p = 0.7$ worked favorably as a form of regularization. Likewise, we found that setting the LSTM unit length k_h and hidden feature vector length k_u to relatively small values ($k_h = k_u = 50$) worked well for this task. We train at most 25 epochs with a mini-batch

size of three instances using RMSProp [41], again picking the parameter state checkpoint with the best F1 score on the validation to be used for testing and evaluation. Here also we ensemble ten ReHAN models and average their outputs to make final predictions.

4.4. Evaluation Measure and Experimental Design

Since the distribution of labels is not balanced, we evaluated the effectiveness of our methods using the F1 score metric instead of accuracy. We use label specific precision, recall, and F1-score to measure per-label effectiveness of our methods. In order to evaluate the overall model performance over all 11 conditions, we used the well known micro-averaged F1 score [42] which is the harmonic mean of

$$\text{micro precision} = \frac{\sum_c \text{TP}_c}{\sum_c \text{TP}_c + \sum_c \text{FP}_c} \quad \text{and} \quad \text{micro recall} = \frac{\sum_c \text{TP}_c}{\sum_c \text{TP}_c + \sum_c \text{FN}_c},$$

where TP_c, FP_c, and FN_c are the true positive, false positive, and false negative counts, respectively, for class c. We evaluated each method using the 10-fold cross-validation technique. Given we also need a validation dataset for hyperparameter tuning, we used eight folds for training, one fold for validation, and the remaining one for testing.

5. Main Results and Discussion

Our per-label scores are shown in Table 3 and micro measures are displayed in Table 4. In the rest of this section we analyze these results and discuss interpretability aspects of the ReHAN model.

5.1. Result Analysis

From Table 3, we find that neural models outperform the baselines for nine out of 11 labels in terms of F1 scores. The CNN model is the best performer for five labels, ReHAN model for four labels, and SVM for the other two labels. Although relatively short, we find that the history of present illness field tends to be a good predictor of some conditions, such as depression and anxiety, with F1 scores of 87 and 80 respectively. However, these are also the top two most frequent labels in our list (depression and anxiety occur in 77% and 68% of all records, respectively). The ReHAN approach exhibits high recall across almost all labels as observed from Table 3. We see that the CNN/Thresholding and ReHAN approaches complement each other very well with CNN/Thresholding being the best model for predicting bipolar disorders, depression, eating disorders, and OCD/OCSD while ReHAN is the decisive choice for predicting ADHD, dementia, complicated grief, and panic disorder.

From the per-label results we conjecture that it is possible to improve on these results by either (1) combining the RNN and CNN models at the architectural level or (2) keeping the two models separate and delegating label prediction to the model that exhibits the best results for that particular label. In any case, the ReHAN approach would be the ideal choice under circumstances that require fewer false negatives (better recall). Such situations are not uncommon especially in medicine when it is best to "err on the side of caution". Conversely, the NER approach exhibits high precision overall but very low recall (nearly half that of the ReHAN model), which can be attributed to the fact that it is looking for presence of very

Label	SVM			NER			CNN			CNN/ML			CNN/Threshold			ReHAN		
	P%	R%	F%	P%	R%	F%	P%	R%	F%	P%	R%	F%	P%	R%	F%	P%	R%	F%
ADHD	58.7	45.5	50.5	**87.2**	23.0	35.7	63.2	54.8	58.0	63.4	50.2	54.6	62.2	58.8	59.9	56.4	**65.3**	**60.1**
Anxiety	69.4	95.9	**80.4**	**75.7**	66.4	70.7	68.5	95.8	79.9	68.2	94.7	79.2	68.0	**96.4**	79.7	69.2	94.5	79.9
Bipolar	58.5	37.6	45.3	**79.9**	24.2	36.9	72.6	44.8	54.9	78.3	36.7	49.6	71.6	47.5	**56.7**	58.5	**54.0**	54.1
Dementia	41.5	20.0	26.2	31.8	3.6	6.3	55.0	17.3	25.6	**63.5**	5.7	10.1	59.7	16.3	24.9	48.0	**23.8**	**30.4**
Depression	78.5	98.1	87.1	**84.7**	64.9	73.4	78.7	98.0	87.2	78.3	97.8	86.9	78.7	98.1	**87.3**	76.9	**99.3**	86.6
Eating Disorder	50.2	31.6	38.4	62.7	17.8	27.4	61.4	35.0	43.0	**72.1**	22.2	32.4	62.0	35.5	**44.0**	48.5	**38.8**	42.4
Grief	47.5	20.8	28.1	37.7	3.3	6.0	54.7	17.3	25.6	**58.2**	7.2	12.4	56.2	16.6	24.7	49.3	**28.6**	**35.0**
OCD/OCSD	60.2	38.6	46.2	46.7	21.0	28.7	63.0	42.8	49.4	**65.0**	28.4	38.2	61.9	44.8	**51.2**	50.3	**47.3**	47.8
Panic	**60.6**	57.0	58.2	60.4	31.6	41.1	56.3	70.9	61.9	56.1	64.8	59.4	57.0	72.8	63.2	55.4	**77.5**	**64.2**
Psychosis	57.7	**25.5**	**34.9**	24.3	17.5	20.1	57.7	22.3	30.9	**69.3**	10.3	17.0	54.9	24.3	31.4	46.0	23.2	28.7
PTSD	52.0	42.7	45.7	**66.7**	20.2	30.8	57.8	49.2	52.0	54.0	34.5	40.8	56.8	**53.6**	**54.3**	48.9	47.1	47.2

Table 3: Results comparing methods on each target label

specific terms in the text with little inference. Nevertheless, it does very well in predicting depression since depression is more likely to be explicitly discussed in text. After depression and anxiety, we notice F1 scores over 60 for ADHD and panic disorder.

	Micro-P (%)	Micro-R (%)	Micro-F (%)
NER	**69.500**	34.400	46.000
SVM	63.863	56.423	59.787 ± 0.583
CNN	65.386	60.789	62.843 ± 0.704
CNN+Meta-Labeler	67.029	53.314	59.276 ± 0.657
ReHAN	59.478	**65.184**	61.904 ± 0.946
CNN+Thresholding	65.629	61.115	**63.144** ± 0.709
CNN+Thresholding+RandInit	64.857	59.641	62.000 ± 0.941

Table 4: Results comparing overall effectiveness of our methods

According to the results in Table 4, the CNN model with output score thresholding works best overall with a mean micro-averaged F1 score of 63.144, offering good balance of precision and recall. The base CNN model comes in at second in terms of mean micro F1 score. The last column shows 95% confidence intervals around the mean F1 score computed using 40 repeated experiments using different shuffles of our dataset. We performed pair-wise comparisons between different models using the F1 scores from these forty train-test splits with the paired t-test approach. All our deep net models except for the CNN meta-labeler model had statistically significant ($p < 0.05$) improvements over the linear SVM model. We also found that the CNN model with thresholding showed statistically significant ($p < 0.05$) improvements over all other models in Table 4.

When evaluated based on the F2 measure (which gives more importance to recall), we found that the ReHAN outperforms all other models. This is not surprising given we noticed that its recall gains are statistically significant ($p < 0.05$) in comparison with all other models. The CNN with meta-labeler performs poorly even when compared to the base CNN model and this is likely because the former is underestimating the label count and making

very precise predictions at the expense of recall. To evaluate the importance of using pre-trained word embeddings instead of randomly initialized embeddings, we took our model with the best mean performance, the CNN with thresholding, and experimented with it using randomly initialized word vectors. The mean F1 score went down by more than 1% (last two rows of Table 4) and this dip was found to be statistically significant ($p < 0.05$). Hence, it is clear that pre-trained embeddings would be helpful for this task based on this dataset.

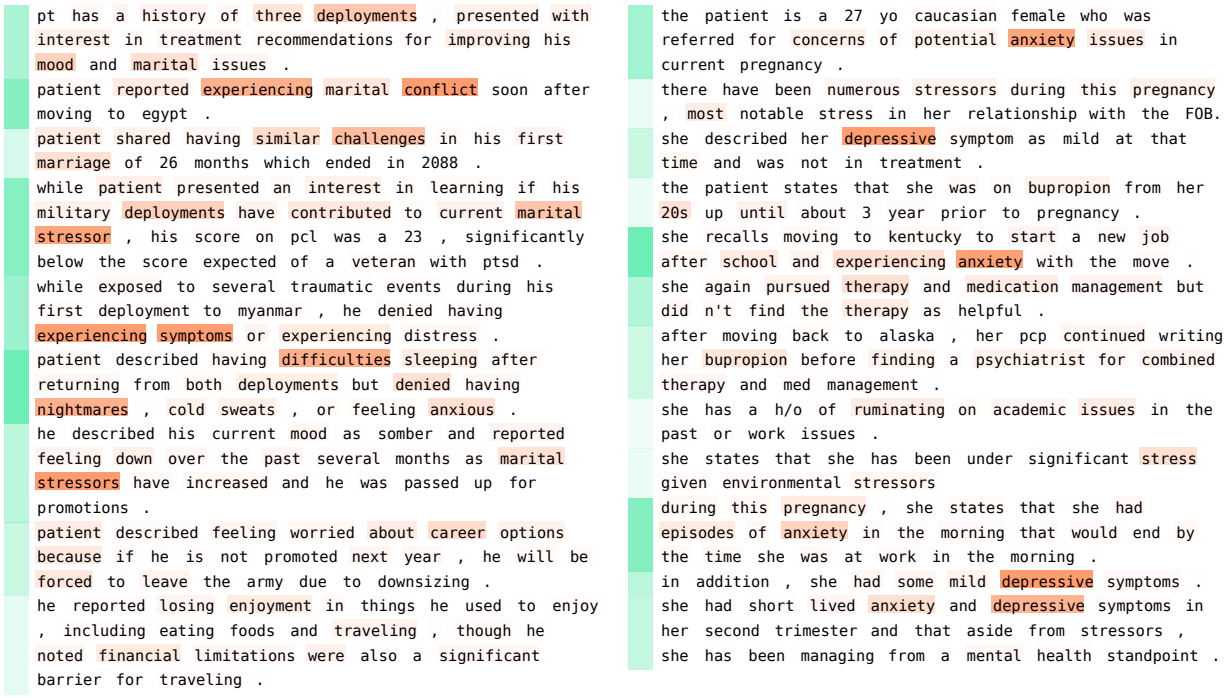
These are our preliminary results with a 986 instance dataset and we believe the performances will improve with larger datasets; as we do not need any hand labeling for this task, adding more records that are curated as part of routine patient care should be feasible in general with appropriate IRB protocols.

5.2. Interpretability and Error Analysis

In Section 3.2 we introduced the ReHAN model that employs hierarchical attention mechanisms. Such a model is able to learn to recognize the importance of words and sentences based on context as it pertains to the task. Specifically, once a test instance is passed through the model, the weights that are generated for different words and sentences at runtime can be visualized and interpreted in order to assess how and why the model made a particular prediction for that instance. This in turn can help the physician make informed final decisions and can be a complementary tool that can help expedite tasks and maintain quality control in a clinical setting. Following the trend established in Yang et al. [43], we scale the weight of each word by the square root of its parent sentence weight and use $\alpha_{i,t} \cdot \sqrt{\hat{\alpha}_i}$ as the word weight for visualization so as to emphasize words in very important sentences while at the same time granting visibility to important words in less important sentences.

In Figure 3a, we show a nuanced true positive instance for depression and anxiety. Only a few segments of the full note are shown with all private health information altered. Although no direct usage of the word depression or its derivatives occur in this narrative, ReHAN is still able to correctly classify based on several highlighted phrases including "feeling down over the past several months" which is directly related to depression based on the questions in the note. Similarly, although the word anxiety is not directly mentioned, highlighted phrases such as "feeling worried about career options", and "losing enjoyment" are directly indicative of anxiety based on the questions for anxiety in the psychiatric review of systems. Even though the word PTSD is present in the note, ReHAN correctly classified it as a negative case for it given the semantics of the note around the word clearly indicate the psychiatrist not leaning toward such an assessment based on the PCL score. In Figure 3b, we see a false positive for anxiety. However, we see several strong indications via highlighted phrases such as "potential anxiety issues", "experiencing anxiety with the move", and "short lived anxiety", all of which seem to indicate that this could be a potential error in the ground truth annotation of this note. That is, we believe this surfaces a potential quality control issue and a possible missed diagnosis. These examples demonstrate the power of hierarchical attention models in producing instance specific insights into the prediction process.

In order to identify ambiguous words that could be sources of difficulty for the model, we looked at the top ranking words for both correct (FPs and TNs) and incorrect (FPs and FNs) predictions for each class. The word-level attention weights are scaled by sentence-level attention weights to better represent its ranking for a document instance given the



(a) A TP for depression and anxiety

(b) An FP for anxiety

Figure 3: ReHAN based visualization of word and sentence weights for interpretability

hierarchical nature of the model. If a word ranks among top five for an instance that is a true positive for some class c, we add it to the set S_c^{TP}. We do the same for FPs, FNs, and TNs by adding top weighted terms in corresponding instances to S_c^{FP}, S_c^{FN}, and S_c^{TN} respectively. We examine the overlap of high ranking terms for TP and FP examples for each label by computing $S_c^{TP} \cap S_c^{FP}$. The terms in these intersections are ambiguous in that their presence is deemed important by the model but reality informs that their presence alone may not be enough to arrive at a positive decision. Similarly we also determine $S_c^{TN} \cap S_c^{FN}$ for each label c. We present some of the more interesting overlapping terms resulting from this experiment in Table 5. We note that there are no overlaps between top five terms for TN and FN instances for anxiety, depression, and panic disorder. Unsurprisingly, these are also the top three labels for which our model performs relatively well (especially with high recall), which may be an indication that there is less semantic ambiguity when making predictions on these labels. We believe these terms may need special handling potentially in a post-processing setup to refine model decisions.

6. Related Work and Limitations

Although the dataset used in our study represents the first of its kind released to all participants in a shared task setting, a few earlier efforts already used psychiatric notes in interesting ways.

NLP applications in psychiatry. Almost all papers in our literature review that applied NLP methods to psychiatric notes are from the past decade. Rumshisky et al. [38] predict early

Label	Intersection of top ranking terms for TPs and FPs	Intersection of top ranking terms for TNs and FNs
ADHD	medication, mental, weight, obsessive, attacks, emotional	emotional, down, feelings
Anxiety	harm, sexual, mood, angry, nightmares, concerns, stress	∅
Bipolar	eating, weight, disorder, mania, generalized, anger	harm, mental, sexual, mood, dose, violence
Dementia	disordered, weight, anger	mood, agoraphobia, driving, crying, abusing, outbursts
Depression	crying, emotionally, insomnia, nightmares, dose, violence	∅
Eating Disorder	violence, years, medication, anger	past, insomnia, heroin, years, stress, attacks
Grief	disordered, weight, spring, problems, anger	issues, mood, crying, experience, feelings, heroin, stress
OCD/OCSD	harm, medication, disordered, years, anger	driving, mood, emotionally, feelings, generalized, experienced
Panic	medication, weight, anger, children, nightmares, obsessive	∅
Psychosis	eating, disordered, weight, years, disorder	house, anger, children, issues, mood, angry, crying, pain
PTSD	program, harm, spring, medication, disorder, nightmares	down, anger, dose, conflict, years, wife, stress, past, behavior

Table 5: Ambiguous but heavily weighted terms leading to FPs and FNs. Other commons terms in these lists that were omitted to avoid redundancy are: depressive, depressed, anxiety, manic, hypomanic, bipolar, panic, psychotic, and psychotherapy.

readmits (within 30 days) to inpatient psychiatric units through topic models built with discharge summaries. Jackson et al. [17] identify over 40 key symptoms (e.g., aggression, apathy, irritability, and stupor) of severe mental illness based on discharge summaries from nearly 8000 patients visiting a UK based mental healthcare provider using SVM models. Perlis et al. [32] provide results of one of the first text mining applications of psychiatric notes where they apply logistic regression models (with LASSO regularization) and show that combining information from unstructured notes with coded information results in major gains in predicting patient mood state when compared with using coded information alone. For additional examples of NLP applications in psychiatry, we refer the reader to this detailed literature review by Abbe et al. [2]. There is also a quickly growing body of literature detailing machine learned models to predict mental health status based on social media data. For a detailed analysis of the current state-of-the-art in this emerging domain, readers are encouraged to refer to the deep learning architecture by Benton et al. [6]. An important related effort by Pestian et al. [33] involves identifying emotions discussed in suicide notes.

Limitations of our study. An important caveat of our work is that concept mapping based approach through MetaMap is a weak baseline that relies on catching only explicit direct mentions of conditions in the notes and does not go for any prediction/inference. As such, it is expected to perform poorly as a baseline. In fact, its recall in our task is almost half that of the recall achieved by our best model. This method may simply be capturing those conditions that are the primary reasons for the current visit (and hence directly mentioned) but are nevertheless assessed as part of the psychiatric review of systems, thus showing up in our 11 labels. The SVM model is a stronger baseline and we demonstrated that except the CNN+Meta-labeler model, all our deep net models outperform it.

We would like to clarify that our prediction of the 11 conditions in this study is based solely on the training ground truth labels obtained from the Yes/No responses to the condition-specific questions as explained earlier in Section 2. In this sense, these may not be directly used in medical practice to arrive at a clinical diagnosis given such diagnoses are typically made using more exhaustive resources such as the Diagnostic and Statistical

Manual of Mental Disorders (DSM-5) published by the American Psychiatric Association. Nevertheless, our predictions can be treated as signals that warrant further examination of the patient's case. In this preliminary effort, we have not exploited label correlations. Accounting for such correlations and fine-tuning individual per-label classifiers may lead to further improvements overall.

7. Conclusion

In this paper, we demonstrated that the short *history of present illness* segment in a psychiatric evaluation note can be used as a good predictor for a few psychiatric conditions. We introduced models based on CNNs and RNNs and compared them to baseline models. We showed that CNNs had superior performance on average while RNNs with attention networks are more suitable when interpretability is desired. We found that the CNN model with output score thresholding results in statistically significant improvements over all other models. However, our efforts in employing RNNs to address the problem are preliminary. We believe there is unexplored potential in using attention mechanisms for this particular problem and dataset based on how well the RNN and CNN models complement each other on a per-label evaluation. The next focus of our research will be to improve on performance while retaining interpretability, possibly using CNNs in conjunction with attention mechanisms.

Acknowledgements

We thank anonymous reviewers for their constructive criticism, suggestions for improving readability, and recommendations for better evaluations. Thanks to Richard Charnigo for advising us on the paired t-test needed to assess statistical significance of our results. We are grateful to the U.S. National Library of Medicine for providing the primary support for this work through grant R21LM012274. We are thankful for additional support by the National Center for Advancing Translational Sciences through grant UL1TR001998 and the Kentucky Lung Cancer Research Program through grant PO2 41514000040001. Finally, we are grateful to the organizers of the N-GRID clinical NLP shared task and the support through NIH grants MH106933 and R13LM011411 that made the task and the associated workshop possible. The content is solely the responsibility of the authors and does not necessarily represent the official views of the NIH.

References

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL `http://tensorflow.org/`. Software available from tensorflow.org.

[2] A Abbe, C Grouin, P Zweigenbaum, and B Falissard. Text mining applications in psychiatry: a systematic literature review. *International journal of methods in psychiatric research*, 25(2):86–100, 2016.

[3] Alan R. Aronson and François-Michel Lang. An overview of MetaMap: historical perspective and recent advances. *Journal of the American Medical Informatics Association*, 17(3):229–236, 2010.

[4] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[5] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. A neural probabilistic language model. *The Journal of Machine Learning Research*, 3:1137–1155, 2003.

[6] Adrian Benton, Margaret Mitchell, and Dirk Hovy. Multitask learning for mental health conditions with limited social media data. In *Proceedings of the 15th Conference of the EACL: Volume 1, Long Papers*, pages 152–162, 2017.

[7] James Bergstra, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, and Yoshua Bengio. Theano: a CPU and GPU math expression compiler. In *Proceedings of the Python for Scientific Computing Conference (SciPy)*, June 2010. Oral Presentation.

[8] Rich Caruana. Multitask learning. In *Learning to learn*, pages 95–133. Springer, 1998.

[9] Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM, 2008.

[10] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537, 2011.

[11] Benjamin Druss and Elizabeth Walker. Mental disorders and medical comorbidity. `http://www.rwjf.org/content/dam/farm/reports/issue_briefs/2011/rwjf69438`.

[12] Michele Filannino, Amber Stubbs, and Özlem Uzuner. Symptom severity prediction from neuropsychiatric clinical records: Overview of 2016 CEGS N-GRID Shared Tasks Track 2. *Journal of Biomedical Informatics*, 2017.

[13] Felix A Gers, Jürgen Schmidhuber, and Fred Cummins. Learning to forget: Continual prediction with LSTM. *Neural computation*, 12(10):2451–2471, 2000.

[14] Yoav Goldberg. A primer on neural network models for natural language processing. *Journal of Artificial Intelligence Research*, 57:345–420, 2016.

[15] Alex Graves. *Supervised Sequence Labelling with Recurrent Neural Networks*, volume 385 of *Studies in Computational Intelligence*. Springer, 2012. ISBN 978-3-642-24796-5. doi: 10.1007/978-3-642-24797-2.

[16] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[17] Richard G Jackson, Rashmi Patel, Nishamali Jayatilleke, Anna Kolliakou, Michael Ball, Genevieve Gorrell, Angus Roberts, Richard J Dobson, and Robert Stewart. Natural language processing to extract symptoms of severe mental illness from clinical text: the clinical record interactive search comprehensive data extraction (cris-code) project. *BMJ open*, 7(1):e012012, 2017.

[18] Nal Kalchbrenner and Phil Blunsom. Recurrent continuous translation models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1700–1709, 2013.

[19] Andrej Karpathy. Neural Networks Part 2: Setting up the data and the loss. `http://cs231n.github.io/neural-networks-2/`, 2016.

[20] Ramakanth Kavuluru and Yuan Lu. Leveraging output term co-occurrence frequencies and latent associations in predicting medical subject headings. *Data & Knowledge Engineering*, 94(Part B):189–201, 2014.

[21] Ramakanth Kavuluru, Anthony Rios, and Yuan Lu. An empirical evaluation of supervised learning approaches in assigning diagnosis codes to electronic medical records. *Artificial intelligence in medicine*, 65(2):155–166, 2015.

[22] Yoon Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, October 2014.

[23] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[24] Jiwei Li, Xinlei Chen, Eduard Hovy, and Dan Jurafsky. Visualizing and understanding neural models in nlp. In *Proceedings of NAACL-HLT*, pages 681–691, 2016.

[25] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119, 2013.

[26] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted Boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 807–814, 2010.

[27] Jinseok Nam, Jungi Kim, Eneldo Loza Mencía, Iryna Gurevych, and Johannes Fürnkranz. Large-scale multi-label text classification – revisiting neural networks. In *Machine Learning and Knowledge Discovery in Databases*, pages 437–452. Springer, 2014.

[28] National Institute of Mental Health. Any mental illness (AMI) among U.S. adults. `https://www.nimh.nih.gov/health/statistics/prevalence/any-mental-illness-ami-among-us-adults.shtml`.

[29] National Library of Medicine. Unified Medical Language System Reference Manual. `http://www.ncbi.nlm.nih.gov/books/NBK9676/`, 2009.

[30] Jiquan Ngiam, Adam Coates, Ahbik Lahiri, Bobby Prochnow, Quoc V Le, and Andrew Y Ng. On optimization methods for deep learning. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 265–272, 2011.

[31] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. *Proceedings of the 30th International Conference on Machine Learning*, 28:1310–1318, 2013.

[32] RH Perlis, DV Iosifescu, VM Castro, SN Murphy, VS Gainer, Jessica Minnier, T Cai, S Goryachev, Q Zeng, PJ Gallagher, et al. Using electronic medical records to enable large-scale studies in psychiatry: treatment resistant depression as a model. *Psychological medicine*, 42(01):41–50, 2012.

[33] John P Pestian, Pawel Matykiewicz, Michelle Linn-Gust, Brett South, Ozlem Uzuner, Jan Wiebe, K Bretonnel Cohen, John Hurdle, and Christopher Brew. Sentiment analysis of suicide notes: A shared task. *Biomedical informatics insights*, 5(Suppl 1):3, 2012.

[34] Sampo Pyysalo, Filip Ginter, Hans Moen, Tapio Salakoski, and Sophia Ananiadou. Distributional semantics resources for biomedical text processing. *Proceedings of 5th International Symposium on Languages in Biology and Medicine*, pages 39–44, 2013.

[35] Anthony Rios and Ramakanth Kavuluru. Supervised extraction of diagnosis codes from EMRs: Role of feature selection, data selection, and probabilistic thresholding. In *Healthcare Informatics (ICHI), 2013 IEEE International Conference on*, pages 66–73. IEEE, 2013.

[36] Anthony Rios and Ramakanth Kavuluru. Analyzing the moving parts of a large-scale multi-label text classification pipeline: Experiences in indexing biomedical articles. In *Healthcare Informatics (ICHI), 2015 International Conference on*, pages 1–7. IEEE, 2015.

[37] Anthony Rios and Ramakanth Kavuluru. Convolutional neural networks for biomedical text classification: application in indexing biomedical articles. In *Proceedings of the 6th ACM Conference on Bioinformatics, Computational Biology and Health Informatics*, pages 258–267. ACM, 2015.

[38] A Rumshisky, M Ghassemi, T Naumann, P Szolovits, VM Castro, TH McCoy, and RH Perlis. Predicting early psychiatric readmission with natural language processing of narrative discharge summaries. *Translational Psychiatry*, 6(10):e921, 2016.

[39] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[40] Amber Stubbs, Michele Filannino, and Özlem Uzuner. De-identification of psychiatric intake records: Overview of 2016 CEGS N-GRID Shared Tasks Track 1. *Journal of Biomedical Informatics*, 2017.

[41] Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2), 2012.

[42] Grigorios Tsoumakas, Ioannis Katakis, and Ioannis P. Vlahavas. Mining multi-label data. In *Data Mining and Knowledge Discovery Handbook*, pages 667–685. 2010.

[43] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489. Association for Computational Linguistics, 2016.