Learning from the Experience of Doctors: Automated Diagnosis of Appendicitis Based on Clinical Notes

Steven Kester Yuwono **Hwee Tou Ng**
Department of Computer Science
National University of Singapore
sky@u.nus.edu
nght@comp.nus.edu.sg

Kee Yuan Ngiam
Department of Surgery
National University Hospital
kee_yuan_ngiam@nuhs.edu.sg

Abstract

The objective of this work is to develop an automated diagnosis system that is able to predict the probability of appendicitis given a free-text emergency department (ED) note and additional structured information (e.g., lab test results). Our clinical corpus consists of about 180,000 ED notes based on ten years of patient visits to the Accident and Emergency (A&E) Department of the National University Hospital (NUH), Singapore. We propose a novel neural network approach that learns to diagnose acute appendicitis based on doctors' free-text ED notes without any feature engineering. On a test set of 2,000 ED notes with equal number of appendicitis (positive) and non-appendicitis (negative) diagnosis and in which all the negative ED notes only consist of abdominal-related diagnosis, our model is able to achieve a promising $F_{0.5}$-score of 0.895 while ED doctors achieve $F_{0.5}$-score of 0.900. Visualization shows that our model is able to learn important features, signs, and symptoms of patients from unstructured free-text ED notes, which will help doctors to make better diagnosis.

1 Introduction

Medical diagnosis is an important task which requires high accuracy and efficiency, especially for patients admitted to the accident and emergency (A&E) department of a hospital. These patients have a wide range of medical conditions. However, it is highly improbable for a medical doctor to gain expertise in all medical fields. Therefore, it is very challenging for the attending doctors to perform quick and accurate diagnosis in order to prevent further complications.

Most of the relevant and useful information (e.g., signs and symptoms) is in the form of free text notes entered by medical doctors. The text does not consist of well-formed and well-structured sentences, but rather sentence fragments containing medical abbreviations and frequent misspelling (due to the time constraints imposed on doctors).

The task addressed in this paper is to diagnose *acute appendicitis*, a binary classification task. Appendicitis was chosen because of the fact that the lifetime risk of having appendicitis is high (8.6% for males and 6.7% for females (Addiss et al., 1990)). Furthermore, there would be high clinical impact if our system is successful. Besides reducing the number of misdiagnoses, our system is expected to help reduce cost by minimizing the number of patients requiring Computed Tomography (CT) scans. CT scans are performed by doctors when they are unsure whether a patient suffers from appendicitis. Although CT scans were found to be 98% accurate in diagnosing acute appendicitis (Rao et al., 1998), they are harmful to our body – one CT scan emits approximately 400[1] times the radiation of a regular chest X-ray. Moreover, there is an exponential increase (from 2.9% to 82.4% in 22 years) in CT scan utilization without any improvement in outcomes (Repplinger et al., 2016; Markar et al., 2014).

We propose a neural network model, which is a combination of a convolutional neural network (CNN) (LeCun et al., 1989), a recurrent neural network (RNN) (Elman, 1990), and a residual network (He et al., 2016) inspired by their recent successes in multiple tasks. RNN has proven to be successful in natural language processing (NLP) tasks such as machine translation (Bahdanau et al., 2015), automated essay scoring (Taghipour and Ng, 2016), and question answering (Kundu and

[1] https://www.fda.gov/radiation-emittingproducts/radiationemittingproductsandprocedures/medicalimaging/medicalx-rays/ucm115329.htm (Accessed on 7 June, 2019)

Proceedings of the BioNLP 2019 workshop, pages 11–19
Florence, Italy, August 1, 2019.

Ng, 2018). CNN has also been successfully used in NLP (Collobert et al., 2011; Chollampatt and Ng, 2018). The main strength of neural networks is that we can train the model without any feature engineering. Therefore, the model is scalable and generalizable to learn other diseases.

2 Automated Diagnosis

In this section, we define the diagnosis task and the evaluation metric used for measuring the performance of the automated diagnosis system.

2.1 Task Description

We formulate the task as a binary classification problem. Given a free-text ED note, and optional real-valued features (from the structured fields), the model is required to classify the ED note as positive appendicitis (represented by a `1`), or negative appendicitis (represented by a `0`). This is accomplished by producing a probability score, and comparing the score against a threshold, such that the class is positive if the probability score exceeds the threshold.

The corpus of hospital ED notes used in this paper is obtained from the National University Hospital (NUH), Singapore, spanning a period of ten years. However, the diagnosis stored in each ED note is not the true diagnosis. The ground truth is stored in the discharge summary (DS) of a patient after the patient is discharged from the hospital. Our corpus consists of about 180,000 ED notes and DS pairs. Each ED note contains 440 words on average.

The ED notes are written in sentence fragments and point forms, and very often contain abbreviations, symbols, and misspelled words. This adds to the difficulty in diagnosing appendicitis. Moreover, the free-text ED notes contain patients' personal health information (PHI) such as name, identification number, and contact number. The ED notes need to be anonymized (by removing the PHI) before they are used for research purposes. We have developed a simple and efficient algorithm to anonymize the ED notes (Yuwono et al., 2016) and it is used in this work.

2.2 Evaluation Metric

The standard evaluation metrics of binary classification are recall, precision, specificity, F_1-score, and $F_{0.5}$-score. The last two are shown in Equation 1.

$$F_1 = 2 \times \frac{precision \times recall}{precision + recall}$$
$$F_{0.5} = (1 + 0.5^2) \times \frac{precision \times recall}{(0.5^2 \times precision) + recall} \quad (1)$$

Let TP, FP, FN, and TN denote true positive, false positive, false negative, and true negative respectively. The positive class refers to class `1` (appendicitis), while the negative class refers to class `0` (not appendicitis). As clinicians favor precision and specificity over recall, we have adopted $F_{0.5}$-score as our main evaluation metric. We aim to have FP as low as possible to prevent patients from being operated on when they do not have appendicitis. Clinicians view FN as more tolerable (as compared to FP), because doctors are still required to investigate the condition of patients not diagnosed as appendicitis until they recover.

3 Neural Networks

3.1 Model Architecture

We have created a novel neural network architecture named convolutional residual recurrent neural network (CR2). Our architecture is illustrated in Figure 1.

Lookup Table Layer: The first layer of our neural network projects each word into a d_{LT} dimensional space. Given a sequence of words $\mathbf{W}$ represented by their *one-hot* representations $(\mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_M)$, the output of the lookup table layer (LT) is given by Equation 2.

$$\begin{aligned} LT(\mathbf{W}) &= (\mathbf{E}\mathbf{w}_1, \mathbf{E}\mathbf{w}_2, ..., \mathbf{E}\mathbf{w}_M) \\ &= (\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_M) \end{aligned} \quad (2)$$

where $\mathbf{E}$ is the word embedding matrix which is learnt during training and M is the number of words in an ED note.

Convolution Layer: After the dense representation of the input sequence is computed from the lookup table layer, it is fed as the input to a convolution layer to extract *local features*. Given a window of word representations of length l, (i.e., $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_l$), they are first concatenated to form vector $\bar{\mathbf{x}}$, and then an output convolution vector $\mathbf{c}$ of length d_c is computed as shown in Equation 3.

$$\mathbf{c} = \mathbf{W}_v \bar{\mathbf{x}} + \mathbf{b} \quad (3)$$

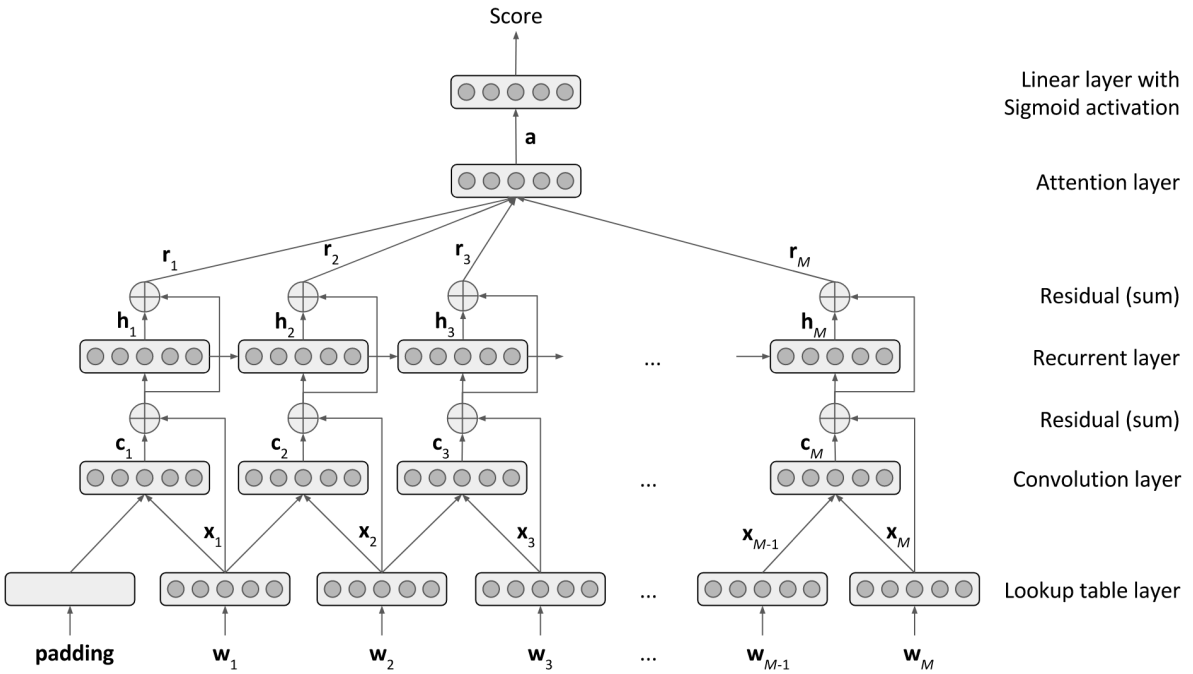

Figure 1: Our neural network architecture (CR2).

$\mathbf{W}_v$ and $\mathbf{b}$ are the trainable weight and bias parameters respectively, and they are shared across all windows in a sequence.

Residual Layer: We perform the sum operation on the sequence of the word embeddings ($\mathbf{X} = \mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_M$) and the output of the convolutional layer ($\mathbf{C} = \mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_M$) as shown in Equation 4.

$$\text{Sum}(\mathbf{X}, \mathbf{C}) = \mathbf{X} + \mathbf{C} \tag{4}$$

To be able to perform the sum operation as shown above, the dimension of the word embeddings (d_{LT}) and the dimension of the output vectors of the convolution layer (or the number of filters) (d_c) have to be equal.

Recurrent Layer: After combining *local features* extracted by the convolution layer with the original dense word representations, the resulting vectors are fed as input to a recurrent layer. The recurrent layer processes the input to generate a representation of a given ED note. There are three well-known RNN units: basic recurrent units (Elman, 1990), gated recurrent units (GRU) (Cho et al., 2014), and long short-term memory units (LSTM) (Hochreiter and Schmidhuber, 1997). Based on our experimental results, LSTM outperforms the other two units and hence we only use LSTM as our RNN unit.

LSTM is able to learn to preserve or forget information. To control the flow of information, LSTM uses three gates to forget or pass information to the next time step. The formal definition of LSTM is described in Equation 5.

$$\begin{aligned}
\mathbf{i}_t &= \sigma(\mathbf{W}_i \mathbf{x}_t + \mathbf{U}_i \mathbf{h}_{t-1} + \mathbf{b}_i) \\
\mathbf{f}_t &= \sigma(\mathbf{W}_f \mathbf{x}_t + \mathbf{U}_f \mathbf{h}_{t-1} + \mathbf{b}_f) \\
\tilde{\mathbf{c}}_t &= \tanh(\mathbf{W}_c \mathbf{x}_t + \mathbf{U}_c \mathbf{h}_{t-1} + \mathbf{b}_c) \\
\mathbf{c}_t &= \mathbf{i}_t \circ \tilde{\mathbf{c}}_t + \mathbf{f}_t \circ \mathbf{c}_{t-1} \\
\mathbf{o}_t &= \sigma(\mathbf{W}_o \mathbf{x}_t + \mathbf{U}_o \mathbf{h}_{t-1} + \mathbf{b}_o) \\
\mathbf{h}_t &= \mathbf{o}_t \circ \tanh(\mathbf{c}_t)
\end{aligned} \tag{5}$$

$\mathbf{x}_t$ is the input vector at time t. LSTM produces one vector $\mathbf{h}_t$ at each time step t ($\mathbf{h}_0$ is the zero vector). $\mathbf{W}_i, \mathbf{W}_f, \mathbf{W}_c, \mathbf{W}_o, \mathbf{U}_i, \mathbf{U}_f, \mathbf{U}_c, \mathbf{U}_o$ are weight matrices and $\mathbf{b}_i, \mathbf{b}_f, \mathbf{b}_c, \mathbf{b}_o$ are the bias vectors. The circle symbol $\circ$ denotes element-wise multiplication and σ denotes the sigmoid function. The output of the recurrent layer is $\mathbf{H} = (\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_M)$. Following (Taghipour and Ng, 2016), we use every output of the intermediate states of the RNN and perform summing (residual) and then pooling in the next layer to have a better representation of the entire ED note.

Residual Layer: We perform the sum operation on the sequence of the output vectors from the recurrent layer ($\mathbf{H} = \mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_M$) and the output vectors of the previous residual layer

($\text{Sum}(\mathbf{X}, \mathbf{C})$) as shown in Equation 6.

$$\mathbf{R} = \text{Sum}(\mathbf{H}, \mathbf{X} + \mathbf{C}) = \mathbf{H} + \mathbf{X} + \mathbf{C} \quad (6)$$

To be able to perform the sum operation as shown above, the dimension of the word embedding vectors (d_{LT}), output vectors of the convolution layer (d_c), and output vectors of the hidden RNN layer (d_r) have to be equal.

Attention layer: Visualizing the learned model is of high importance in the medical domain. By using an attention mechanism, we can show the degree of importance of words and phrases. Attention mechanism has been successful in many recent studies (Bahdanau et al., 2015; Hermann et al., 2015; Rush et al., 2015). The outputs of the previous residual layer $\mathbf{R} = (\mathbf{r}_1, \mathbf{r}_2, ..., \mathbf{r}_M)$ are used as inputs of the attention layer. In other words, this layer receives M vectors of size d_r, where d_r is the output dimension of the recurrent layer. $\mathbf{R}$ is a rich representation of the words in the ED note using a combination of word embeddings, CNN outputs, and RNN outputs. Each vector $\mathbf{r}_t$ is multiplied by a learnable real-valued weight s'_t between 0 and 1 before adding the elements of all M vectors into a single vector $\mathbf{a}$ as a form of *weighted average*. The functions of the attention layer are defined in Equation 7.

$$\begin{aligned} s_t &= \mathbf{v} \cdot \tanh(\mathbf{W}_r \mathbf{r}_t) \\ s'_t &= \text{softmax}(\mathbf{s})_t \\ \mathbf{a} &= \sum_{t=1}^{M} s'_t \mathbf{r}_t \end{aligned} \quad (7)$$

$\mathbf{W}_r$ is a trainable matrix of size $d_r \times d_r$ and $\mathbf{v}$ is a trainable vector of size d_r. To learn more complex functions, $\mathbf{W}_r$ is introduced to increase the number of parameters and $\tanh$ is introduced to add non-linearity. $\mathbf{W}_r$ and $\mathbf{v}$ are *shared* across all time steps t. To make sure that the weights for all time steps sum to 1, the softmax function is performed on all the weights $\mathbf{s} = (s_1, s_2, ..., s_M)$. The attention layer is able to learn to assign varying weights to different time steps t depending on the input $\mathbf{r}_t$. The main advantage of having an attention layer is that we can retrieve the weight s'_t for each time step, and hence we are able to visualize and measure the importance of each word in the ED note.

Linear Layer with Sigmoid Activation: If there are no additional real-valued features, the input of this layer is the vector $\mathbf{a}$. Otherwise, it will be $[\mathbf{a}, \mathbf{l}]$, the concatenation of $\mathbf{a}$ and $\mathbf{l}$, where $\mathbf{l}$ contains the additional real-valued features which will be described in the next subsection. The linear layer maps the input vector into a single scalar value. This mapping is a simple linear transformation, therefore the computed scalar value is unbounded. Since we are expected to predict either class `0` or `1`, we will use a sigmoid function to ensure the scalar value is in the range $(0, 1)$. The mapping of the linear layer after applying the sigmoid function is shown in Equation 8.

$$s(\mathbf{x}) = \sigma(\mathbf{w} \cdot \mathbf{x} + b) \quad (8)$$

where $\mathbf{x}$ is the input vector $\mathbf{a}$ or $[\mathbf{a}, \mathbf{l}]$, $\mathbf{w}$ is the weight vector, and b is the bias value.

3.2 Additional Real-valued Features

Before using additional real-valued features such as lab results in the neural network, the values need to be normalized. We have adopted `normal sigmoid` to normalize the real-valued features which is shown in Equation 9. $\bar{x}$ and σ represent the mean and standard deviation for a particular feature (e.g., white blood cell count).

$$\begin{aligned} \text{normal}(x) &= \frac{(x - \bar{x})}{\sigma} \\ \text{normal\_sigmoid}(x) &= \frac{1}{1 + e^{-\text{normal}(x)}} \end{aligned} \quad (9)$$

There are also entries where ED notes are not accompanied by any lab results. To deal with missing values, we calculate the mean ($\bar{x}$) of all existing entries in the training set of that particular feature (e.g., white blood cell count) and then use the average value to fill in the gap.

In order to include the L real-valued normalized features $\mathbf{l} = (l_1, l_2, ..., l_L)$ in the model, we concatenate L real numbers (after normalizing them) to the output of the attention layer, before going into the next layer. The input of the final layer will be $[\mathbf{a}, \mathbf{l}]$, a vector of size $d_r + L$. Figure 2 illustrates the process above.

3.3 Training

We use the RMSProp optimization algorithm (Dauphin et al., 2015) to minimize a loss function over the training data. Given N training ED notes and their corresponding true class s_i^* (either `0` or `1`), the model computes the predicted score s_i in the range of $(0, 1)$ for all training ED notes and then updates the network weights such that the loss

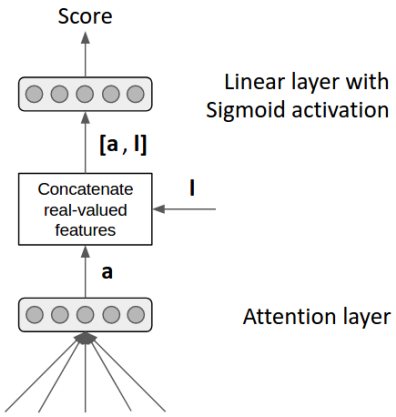

Figure 2: Concatenation of real-valued features before the final layer.

function is minimized. The loss function we have adopted in our system is the binary cross-entropy loss function as shown in Equation 10.

$$L(\mathbf{s}, \mathbf{s}^*) = -\sum_{i=1}^{N} s_i^* \log(s_i) + (1 - s_i^*) \log(1 - s_i) \tag{10}$$

In our data set, the distribution of the classes is highly imbalanced – the proportion of ED notes in class `0` can be as high as 98.4%, with the remaining 1.6% ED notes in class `1`. To tackle this problem, we have adopted a *weighted* binary cross-entropy loss function, where each class is weighted *inversely proportional* to the class frequency in the training data to allocate more weight to the less frequent class, similar to the technique used by (Chollampatt et al., 2016) for rescaling.

To prevent overfitting, we have adopted dropout (Srivastava et al., 2014) regularization. We also clip the gradient if the gradient norm is larger than a certain threshold. We train the neural network for a specified number of epochs and evaluate the model on a validation set in every epoch. The epoch with the highest $F_{0.5}$-score on the validation set is then selected as the final model.

3.4 Threshold Adjustment

The output or score of the neural network is a real number between 0 and 1. However, we need to transform the score to either `1` (positive) or `0` (negative) to solve our binary classification problem. Therefore, there is a need to set a threshold as the decision boundary. The default threshold used to split the two classes is 0.5. For example, if the prediction score is greater than 0.5, then the predicted class is positive (appendicitis); otherwise negative (not appendicitis).

The aforementioned threshold can be used to tune the model to have lower FP but higher FN, and vice versa. In this paper, we would like to achieve the lowest possible FP, trading for a higher FN. To achieve this, we use the validation set to search for a threshold with the best $F_{0.5}$-score. First, we use the model in the current epoch to predict the score of each instance in the validation set. Second, we sort the validation instances in ascending order of the predicted scores. Third, we perform a linear search to find the cut-off threshold to achieve the best $F_{0.5}$-score on the validation set. This is repeated in every epoch, resulting in a unique threshold for each epoch. The epoch with the best $F_{0.5}$-score (using its own unique threshold) on the validation set is used as the final model to evaluate the test set, using the same threshold used in the validation set.

4 Experiments

4.1 Setup

Our network has several hyper-parameters which need to be set. We use the RMSProp optimizer with decay rate of 0.9 and learning rate of 0.001. Mini-batch[2] size is 32 and the model is trained for 25 epochs. The vocabulary is created using all words in the training set. Out-of-vocabulary words are replaced by a special `<unknown>` token. Words that contain any digits are replaced by a special `<num>` token. The network is regularized by using dropout (Srivastava et al., 2014) with probability 0.5. During training, if the norm of the gradient exceeds 10, it will be clipped to a maximum value of 10. Word embedding dimension (d_{LT}), output dimension of the hidden layer for the RNN (d_r), and the number of filters for the CNN (d_c) are set to 300. The convolution window size (l) is set to 3. We initialize the lookup table layer with our custom pre-trained word embeddings which are trained using our entire corpus of 180,000 ED notes excluding the notes used as validation and test set. We use the word2vec skip-gram model (Mikolov et al., 2013) to train our word embeddings. Although the lookup table

[2]To create mini-batches for training, all the ED notes in a mini-batch are padded using a dummy token to have the same length. To remove the effect of padding tokens during training, they are masked to prevent the network from miscalculating the gradients.

layer is initialized with pre-trained word embeddings, the lookup table layer is trainable and not fixed. We utilize 4 additional features from the structured patient data, namely age, gender, and two lab test results (white blood cell count and neutrophils), and incorporate them into the network as described in Section 3.2.

4.2 Dataset

We have about 180,000 ED notes and DS pairs in total. The class distribution of the ED notes in the entire corpus is shown in Table 1 (second and third column). The first class listed in the first column is the class predicted by ED doctors in the ED notes, while the second class listed in the first column is the true diagnosis class obtained from the DS.

4.2.1 Dataset 1: Natural Distribution (Original Dataset)

Using the corpus shown in Table 1, we randomly sample 10% for training, 10% for validation, and 10% for test. The number of ED notes is 18,111, 18,108, and 18,107 respectively following its natural class distribution (about 1.6% positive ED notes). To speed up training, we only use ED notes with 750 words or less in the training set, resulting in 16,854 instead of 18,111 ED notes for training. We do not impose any length limit for both the validation and test set.

Class	Number of ED notes	Percentage
++ (TP)	2,194	1.2 %
+− (FP)	1,071	0.6%
−+ (FN)	796	0.4 %
−− (TN)	177,210	97.8 %
Total	181,271	100 %

Table 1: Class distribution of ED notes.

4.2.2 Dataset 2: Equal Class Distribution with Random Negative ED Notes

In our second dataset, we obtain a subset of the 181,271 ED notes (from Table 1) to create a dataset with 50% positive and 50% negative ED notes. There are 2,980, 1,000, and 2,000 ED notes for training, validation, and test respectively with equal distribution of positive and negative classes in each set. The negative ED notes consist of randomly sampled ED notes of all diagnosis classes that are not appendicitis.

4.2.3 Dataset 3: Equal Class Distribution with Abdominal-related Negative ED Notes

Our third dataset is very similar to our second dataset (in Section 4.2.2) with the same class distribution. The only difference is that the negative ED notes in this dataset only consist of abdominal-related diagnosis instead of any random diagnosis that is not appendicitis. The number of ED notes in the training, validation, and test set are the same as those in dataset 2. The 1,000 positive ED notes in this test set are identical to the 1,000 positive ED notes in the test set in dataset 2. Dataset 3 is more challenging than dataset 2 because the signs and symptoms of appendicitis are very similar to those of other abdominal conditions. The class distribution of all three test sets is shown in Table 2.

4.3 Results and Discussions

The experimental results of the best model (CR2, described in Sections 3 and 4.1) on the three datasets are summarized in Table 3.

We train the neural network model (end to end) on a single GPU (Nvidia TITAN X Pascal), and the training time is 3.2 hours for dataset 1, and 35 minutes for each of the datasets 2 and 3. After the model is trained, it is able to perform acute appendicitis diagnosis rapidly, at 400 ED notes per second. The *best* single CR2 model is chosen based on the highest $F_{0.5}$-score on the validation set over 50 runs with different seeds. The *average* score for the CR2 model in each column is calculated over 50 runs with different seeds. The $\pm$ sign represents the standard deviation over the 50 runs.

We have two baseline methods, namely a maxent (maximum entropy, also known as logistic regression) classifier and an Alvarado rule-based scoring system. This is inspired by prior work (Deleger et al., 2013) which performs appendicitis risk stratification using an Alvarado rule-based scoring system with features obtained from free text. Before using the aforementioned two methods, the texts are first tokenized, and negation are detected through Negex (Chapman et al., 2001), a simple regular expression rule-based algorithm which has been modified to suit our needs. For maxent, a list of words is built from the training ED notes and we obtain the bag-of-words representation for each ED note, add the lab results and other structured fields, and then use them as features to train a maxent classifier.

Class	Dataset 1		Dataset 2		Dataset 3	
	# ED notes	**%**	**# ED notes**	**%**	**# ED notes**	**%**
++ (TP)	216	1.2 %	734	36.7 %	734	36.7 %
+− (FP)	104	0.6 %	6	0.3 %	36	1.8 %
−+ (FN)	78	0.4 %	266	13.3 %	266	13.3 %
−− (TN)	17,709	97.8 %	994	49.7 %	964	48.2 %
Total	18,107	100 %	2,000	100 %	2,000	100 %

Table 2: Class distribution of ED notes in test sets.

model	TP	FP	FN	TN	Rec	Prec	Spec	F1	F0.5	Acc
Dataset 1										
ED	216	104	78	17,709	0.735	0.675	0.994	0.704	0.686	0.990
ME	138	126	156	17,687	0.469	0.523	0.993	0.495	0.511	0.984
Alv	124	90	170	17,723	0.422	0.579	0.995	0.488	0.539	0.986
Best	141	90	153	17,723	0.480	0.610	0.995	0.537	**0.579***	0.987
Avg	154.8	109.2	139.2	17,703.8	0.527	0.588	0.994	0.553	0.573	0.986
	±16.9	±18.8	±16.9	±18.8	±0.058	±0.021	±0.0011	±0.030	±0.016	±0.00046
Dataset 2										
ED	734	6	266	994	0.734	0.992	0.994	0.844	0.927	0.864
ME	952	62	48	938	0.952	0.939	0.938	0.945	0.941	0.945
Alv	617	12	383	988	0.617	0.981	0.988	0.758	0.877	0.803
Best	912	27	88	973	0.912	0.971	0.973	0.941	**0.959***	0.943
Avg	912.1	28.6	87.9	971.4	0.912	0.970	0.971	0.940	0.958	0.942
	±17.1	±6.1	±17.1	±6.1	±0.017	±0.0058	±0.0061	±0.0076	±0.0037	±0.0069
Dataset 3										
ED	734	36	266	964	0.734	0.953	0.964	0.829	0.900	0.849
ME	880	125	120	875	0.880	0.876	0.875	0.878	0.876	0.878
Alv	617	72	383	928	0.617	0.896	0.928	0.731	0.821	0.773
Best	831	79	169	921	0.831	0.913	0.921	0.870	**0.895***	0.876
Avg	832.1	84.2	167.9	915.8	0.832	0.908	0.916	0.868	0.892	0.874
	±28.8	±12.2	±28.8	±12.2	±0.029	±0.0096	±0.0122	±0.0125	±0.0045	±0.0095

Table 3: Summary of the best model against ED doctors and the baselines on three datasets. The baseline for the statistical significance tests is underlined and statistically significant improvements ($p < 0.05$) are marked with '*'. ME stands for Maxent, Alv stands for Alvarado, Best stands for Best CR2, and Avg stands for Avg CR2.

In acute appendicitis diagnosis, there is an existing well-known scoring system, namely Alvarado score (Alvarado, 1986). It is also known as MANTRELS score, which is a mnemonic to remember the score factors (signs, symptoms, and lab readings) – **M**igration of pain to the right lower quadrant, **A**norexia, **N**ausea or vomiting, **T**enderness in the right lower quadrant, **R**ebound pain, **E**levated temperature (fever), **L**eukocytosis (high white blood cell count), and **S**hift of neutrophils to the left. The score for each factor is 1(M), 1(A), 1(N), 2(T), 1(R), 1(E), 2(L), and 1(S) respectively. The score for each factor present in a patient will be added together to obtain the final score. A higher score indicates that a patient is more likely to have appendicitis. The aforementioned 8 factors are detected through a regular expression (with negation) on the ED notes that have been preprocessed with Negex. Different threshold values (scores strictly greater than the threshold will be classified as positive, and negative otherwise) are explored and the threshold with the best $F_{0.5}$-score is chosen. The thresholds for Alvarado scoring in datasets 1, 2, and 3 are 6, 5, and 5 respectively.

Our neural network model (CR2) outperforms the two baselines in $F_{0.5}$-score on all three datasets. We also perform a statistical significance test ($p < 0.05$) to determine whether the obtained improvement is statistically significant. We found that our neural network improvements against maxent on all datasets are statistically significant. This shows that our neural network model is superior to the maxent classifier and Alvarado scoring system.

Based on the first row in Table 3, we can see that ED doctors' performance is better compared to our model. This is mainly caused by class imbalance (1.6% positive and 98.4% negative). Learning and predicting on a dataset with extremely skewed class distribution is challenging. However, as we can see from the results, the performance of our best model is close to that of ED doctors, with 14 fewer FP instances and 75 more FN instances out

of 18,107 ED notes in the test set.

Based on the results of dataset 2 and 3, our model achieved lower FP+FN (in other words, higher accuracy) when compared to ED doctors. With equal distribution of positive and negative ED notes, our model performs better than ED doctors with much lower FN in exchange for slightly higher FP. Our model's $F_{0.5}$-score exceeds that of ED doctor on dataset 2 and is very close to that of ED doctor on dataset 3. Our model also consistently achieves better sensitivity (recall) than the ED doctor.

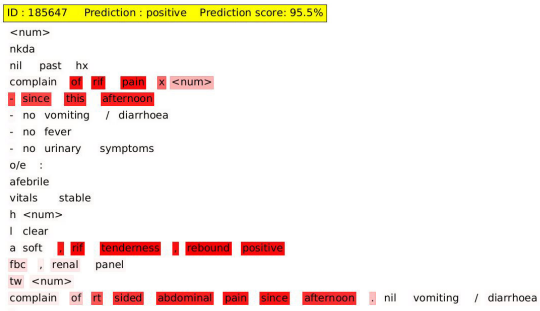

Figure 3: Visualization of how our model interprets a positive ED note.

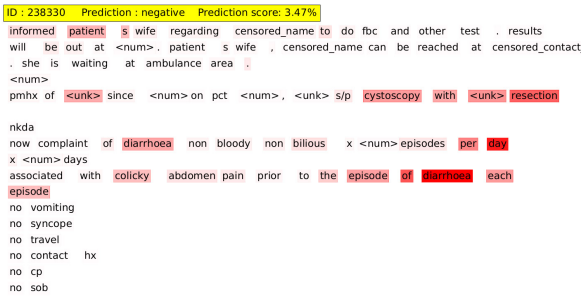

Figure 4: Visualization of how our model interprets a negative ED note.

To visualize the model and gain insights into how the model assigns importance to words and phrases, we retrieve the weights of the attention layer. The weights can be used to show the degree of importance of words and phrases in an ED note. From our observation, the model is able to pick up meaningful signs and symptoms of appendicitis most of the time. Figure 3 shows the visualization of our model, with appendicitis features highlighted, such as rif pain, and tenderness with rebound. In Figure 3, darker shade of red color indicates a higher weight assigned to a word. These signs and symptoms have been validated and used in practice as features of the Alvarado scoring scheme (Alvarado, 1986). On the other hand, the model is also able to pick up the features of non-appendicitis. In Figure 4, the model is able to pick up diarrhea and a few other features suggesting non-appendicitis.

We will explore other neural network architectures and more (deeper) layers in the future. We will also design our experiments to be able to fully utilize the entire 180,000 ED notes to train and validate our model.

5 Conclusion

In this paper, we tackle the task of automated diagnosis using free-text ED notes. We present a machine learning model which is able to learn from free text and optional additional features without any feature engineering. We show that the performance of our novel neural network architecture is promising and close to the performance of ED doctors. Analysis of the visualization shows that the attention layer is able to meaningfully learn the importance of words and phrases in ED notes and to change its emphasis depending on the context of the words. This is helpful in highlighting certain key description (i.e., signs and symptoms) that might have been missed otherwise by medical doctors in a real-life setting.

Acknowledgments

This research is supported by Singapore Ministry of Education Academic Research Fund Tier 1 grant T1-251RES1513.

References

David G Addiss, Nathan Shaffer, Barbara S Fowler, and Robert V Tauxe. 1990. The epidemiology of appendicitis and appendectomy in the United States. *Am. J. Epidemiol.*, 132(5):910–925.

Alfredo Alvarado. 1986. A practical score for the early diagnosis of acute appendicitis. *Ann. Emerg. Med.*, 15(5):557 – 564.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of the 3rd International Conference on Learning Representations*.

Wendy W Chapman, Will Bridewell, Paul Hanbury, Gregory F Cooper, and Bruce G Buchanan. 2001. A simple algorithm for identifying negated findings and diseases in discharge summaries. *J. Biomed. Inform.*, 34(5):301–310.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1724–1734.

Shamil Chollampatt and Hwee Tou Ng. 2018. A multilayer convolutional encoder-decoder neural network for grammatical error correction. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pages 5755–5762.

Shamil Chollampatt, Kaveh Taghipour, and Hwee Tou Ng. 2016. Neural network translation models for grammatical error correction. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pages 2768–2774.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, 12:2493–2537.

Yann Dauphin, Harm de Vries, and Yoshua Bengio. 2015. Equilibrated adaptive learning rates for non-convex optimization. In *Advances in Neural Information Processing Systems 28*, pages 1504–1512.

Louise Deleger, Holly Brodzinski, Haijun Zhai, Qi Li, Todd Lingren, Eric S Kirkendall, Evaline Alessandrini, and Imre Solti. 2013. Developing and evaluating an automated appendicitis risk stratification algorithm for pediatric patients in the emergency department. *J. Am. Med. Inform. Assoc.*, 20(e2):e212–e220.

Jeffrey L Elman. 1990. Finding structure in time. *Cognitive Science*, 14(2):179–211.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems 28*, pages 1693–1701.

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

Souvik Kundu and Hwee Tou Ng. 2018. A nil-aware answer extraction framework for question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4243–4252.

Yann LeCun, Bernhard Boser, John S. Denker, Donnie Henderson, Richard E. Howard, Wayne Hubbard, and Lawrence D. Jackel. 1989. Backpropagation applied to handwritten zip code recognition. *Neural Comput.*, 1(4):541–551.

Sheraz R Markar, Diluka Pinto, Marta Penna, Alan Karthikesalingam, Bulathsinghalage Kalana Sandun Bulathsinghala, Kumaralingam Kumaran, Majid Hashemi, and Ranil Fernando. 2014. A comparative international study on the management of acute appendicitis between a developed country and a middle income country. *Int. J. Surg.*, 12(4):357–360.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26*, pages 3111–3119.

Patrick M Rao, James T Rhea, Robert A Novelline, Amy A Mostafavi, and Charles J McCabe. 1998. Effect of computed tomography of the appendix on treatment of patients and use of hospital resources. *N. Engl. J. Med.*, 338(3):141–146.

Michael D Repplinger, Andrew C Weber, Perry J Pickhardt, Victoria P Rajamanickam, James E Svenson, William J Ehlenbach, Ryan P Westergaard, Scott B Reeder, and Elizabeth A Jacobs. 2016. Trends in the use of medical imaging to diagnose appendicitis at an academic medical center. *J. Am. Coll. Radiol.*, 13(9):1050–1056.

Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 379–389.

Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15:1929–1958.

Kaveh Taghipour and Hwee Tou Ng. 2016. A neural approach to automated essay scoring. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1882–1891.

Steven Kester Yuwono, Hwee Tou Ng, and Kee Yuan Ngiam. 2016. Automated anonymization as spelling variant detection. In *Proceedings of the COLING 2016 Workshop on Clinical Natural Language Processing*, pages 99–103.