Classifying Secure Patient Messages at Stanford Health Care to Streamline Patient-Provider Communication Processes

Jasmine Bilir jbilir@stanford.edu Tran Le tranle@stanford.edu Christopher Moffitt cmoffitt@stanford.edu

1 Introduction

1.1 Problem

Within medicine, secure messages serve as a critical tool for communication and collaboration within healthcare teams and with patients [3]. At Stanford Health Care, the MyHealth application is used to communicate between the patient and care team. Through our interviews with clinical informatics fellow and internal medicine physician Dr. Julian Genkins at Stanford Health Care, we found that there is a large volume of messages which is causing the system to slow and staff to burn out from large amounts of work [4]. These effects also lead to delays in patient care, quality, and satisfaction [4].

1.2 Solution

This project's goal is to solve this issue for Stanford Health Care to improve the speed and quality of patient care. As inputs, we take in real de-identified data from the Stanford Medical Center. The inputs include the text of the messages, response, ID of the sender, response time, provider type, department, and more. Using this data, we develop and use NLP tools such as Doc2Text, removing stop-words, Bidirectional Encoder Representations from Transformers as well as other concepts to analyze secure message text to better understand at a high level what text is contained in messages and to characterize messages by their contents as they relate to which type of medical professional should respond to the message.

The hope is to direct the messages coming in from patients to the correct provider so that the work flow is optimized and patient care is delivered at a faster pace by spreading work out among different departments. Excitingly, this project is strongly supported by the Stanford Healthcare leadership and is a part of a multi-year project to optimize secure clinical messages and address the issues of burnout and dissatisfaction that have reached crisis levels at Stanford Medicine.

1.3 Outcomes

In this research project, we used logistic regression as our baseline model with the Doc2Vec model [5]. We first cleaned, tokenized, and removed stop words. Then, we implemented the Doc2Vec model and got these results. [TODO: insert eval/compar].

For our main approach, we used the BERT (Bidirectional Encoder Representations from Transformers) model [6] to develop a more comprehensive NLP model. We implemented the DistilBERT model and got these results. [TODO: insert eval/compar].

Identify applicable funding agency here. If none, delete this.

2 Literature Review

2.1 Patient2Vec: A Personalized Interpretable Deep Representation of the Longitudinal Electronic Health Record

In the research paper titled "Patient2Vec: A Personalized Interpretable Deep Representation of the Longitudinal Electronic Health Record" [1], the authors propose a computational framework named Patient2Vec. Patient2Vec's purpose is to learn from eletronic health record (EHR) data personalized to each patient. The paper stemmed from the fact that in healthcare, there are huge amounts of patient data as medicine goes digital. However, there is currently very little work to explore this data which could tremendously improve healthcare system. This is the same goal we are trying to achieve in our research project as the Stanford Medicine message data has not been explored before.

The Patient2Vec model uses deep recurrent neural networks to fully pick up on and analyze complex relationships in the patient's healthcare data. First, the model converts medical visit information which is represented as a string into vectors. This is similar to our approach as we also convert patient messages into vectors of words that we can analyze.

After the data is processed, Patient2Vec applies the deep recurrent neural network self-attention mechanism to train the network and learn the weights. Additional static information like age, gender, and previous hospitalization history is added to get a more complete representation of the patient. This is a difference with our method since our data is de-identified. We do not include nor have information on the patient such as age, gender, and previous hospitalization history. After speaking with healthcare professionals, we believe that using de-identified data would be more effective since it prevents the model from classifying data

based on the demographic details of a patient. Instead, the decision is based solely on the text of the message since it should convey what the patient needs. Another difference is that the Patient2Vec model is then used to predict future patient hospitalization risks, whereas our goal is to categorize patient messages to be redirected to the correct medical personal.

2.2 Probing Patient Messages Enhanced by Natural Language Processing: A Top-Down Message Corpus Analysis

In the research paper titled "Probing Patient Messages Enhanced by Natural Language Processing: A Top-Down Message Corpus Analysis," [2] authors Mastorakos et al. performed preliminary NLP research on 3,000 patient portal messages from a variety of departments in order to produce a descriptive analysis of patient text that can contribute to more sophisticated NLP applications in the future. In their project, they manually categorized each message into different buckets such as Active Symptom, Logistical, Prescription, and Update. Then, they performed NER (named entity recognition) on the dataset and found interesting relationships between named entities and the corresponding general categorization buckets. This research shows that there are promising NLP techniques that can be applied to patient messages.

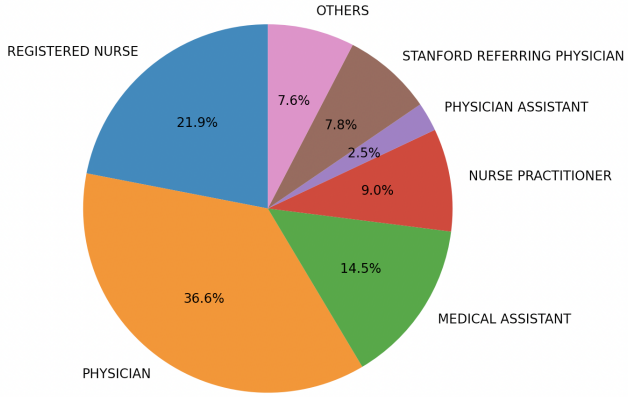
This research project will be extending on the work of Mastorakos et al. However, instead of categorizing patient messages into Active Symptom, Logistical, Prescription, and Update, we will be categorizing the messages into the type of provider that the message should go to. This categorization will provide actionable steps for the healthcare provider team. Additionally, this paper uses only around 3,000 patient portal messages. However, we believe that this number is too low to fully represent the diversity and complexity of healthcare data. For our project, we were able to obtain 391,386 patient portal messages that we will be using to train our models on.

3 Dataset

After completing various privacy and medical training modules and signing forms, we were able to gain access to patient portal message data from the Stanford Medical Center. The data is stored on Google Cloud Platform with various collections, documents, and fields. We first sorted through all the fields to find dependencies and relationships. Then, we created an SQL query to clean and organize our data for export.

The dataset we will be using includes 391386 patient portal messages. Each message includes the message id, message text, subject, message type, and respondent type. All of these features have a data type of a string. First, we will describe the input features. The message id is a 7 digit string that uniquely identifies the message. The original data we were given separately all the messages line-by-line. Therefore, we preprocessed the data to combine all messages of the same id onto one line. The message type is always "Patient Medical Advice Request" as this is what we filtered out messages coming from non-patients. We are focusing on messages sent by the patient.

The respondent type represents which medical personal responded to the patient message. This field represents the output of our data. There are 44 unique types of respondents in our dataset. The distribution of classes is shown below:



However, in attempting to train the model with all 44 unique types, there was too little data in certain categories to accurately train the model. Thus, after experimenting with different numbers, we came to the decision that the most optimal method would be to omit any labels with less than 9000 message data points. This leaves us with the 6 main labels seen on the chart above: registered nurse, physician, medical assistant, nurse practitioner, physician assistant, and Stanford referring physician. This narrows down our dataset from 391386 to 361,649 total messages for training and testing, which we feel is a reasonable tradeoff for a more accurate model learning from more data in each label.

To split our data, we used scikit package [7] and sklearn.model_selection.train_test_split method. This method allows us to accurately and quickly split the our dataset into random train and test sets. For our baseline model, we used a test size of 0.3 with a random state of 42. For our main approach, we used a test size of 0.2 with a random state of 0. We chose different random state numbers between the two approaches to ensure that the split of the dataset is as random as possible. We also decreased the test size of the main approach to 0.2 because after many experiments, we feel that having a slightly large train set leads to better results since the BERT model is more computationally heavy and requires more inputs to achieve optimal outcomes.

4 Baseline

For our baseline model we wanted to see if we could produce reliable classifications using a basic logistic regression. We did not see this as the most comprehensive model to use on this problem, but we viewed this as a good first step in our efforts.

4.1 Implementation:

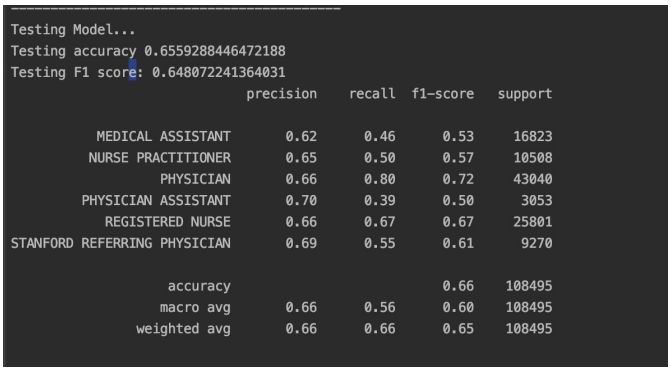
For our logistic regression features, we used the NLP approach of Doc2Vec [5]. Doc2Vec uses the distributed memory and distributed bag of words models published by Quoc Le and Tomas Mikolov [8] at Google to learn paragraph and document embeddings. Doc2vec is an NLP tool for representing documents as a vector. We choose this over a Bag of Words approach because Doc2Vec allowed us

to better featurize the words appearing in our message texts when training our weights. Rather than simply hot encoding words in the document, Doc2vec does a better job at capturing the adjacency's between words. We also chose not to include stop words, or words that don't contribute to the meaning of a sentence like "or" or "a" so as not to skew our weighting process. With these steps, we were able to tokenize and use each health message text as our set of features. We then used logistic regression to train/test classification of each message to one of the 6 providers: medical assistant, nurse practitioner, physician, physician assistant, registered nurse, referring physician. We used sklearn and scikit to help in the overall implementation of our logistic regression.

4.2 Results:

Since we did not view our baseline as a comprehensive version of our model, we measured and analyzed are results using the straightforward accuracy measurements of verifying our model precision on the test set. All around, our model accuracy was correct 65.59% of the time. On specific classification groups, we got the following accuracy ratings:

1) Medical Assistance - 62% accuracy
2) Nurse Practitioner - 65% accuracy
3) Physician - 70% accuracy
4) Registered Nurse - 66%
5) Referring Physician – 69% accuracy



```
Testing Model...
Testing accuracy 0.6559288446472188
Testing F1 score: 0.648072241364031
                              precision    recall  f1-score   support

          MEDICAL ASSISTANT       0.62      0.46      0.53     16823
         NURSE PRACTITIONER       0.65      0.50      0.57     10508
                  PHYSICIAN       0.66      0.80      0.72     43040
        PHYSICIAN ASSISTANT       0.70      0.39      0.50      3053
           REGISTERED NURSE       0.66      0.67      0.67     25801
STANFORD REFERRING PHYSICIAN       0.69      0.55      0.61      9270

                   accuracy                           0.66    108495
                  macro avg       0.66      0.56      0.60    108495
               weighted avg       0.66      0.66      0.65    108495
```

Overall, we were pleased that our baseline was able to give these results, particularly because we had not made very many customization to the specific needs of our project. Likewise, it was great to sanity check that the data was in fact able to reveal correlations that would be helpful for our project. This gave us helpful insights as we moved on to work on a more robust model for our main project.

5 Main Approach

5.1 BERT

In order to develop a more comprehensive NLP model that would glean more specific word encoding and decoding in our health messages, we used a BERT (Bidirectional Encoder Representations from Transformers model) [6] to enhance our classification and predictions of which health professional should be contacted. BERT models are more effective because of greater contextualization in the process of generating and correlating words. Rather than just looking at and weighting the words strictly as they appear, BERT looks for correlations and similarities between words and specific contexts. This thereby produces more clear results that reflect the intricacies and symmetries in written language. BERT also differs as it involves a pre-training step, that builds associations prior to training to begin understanding the relationships between the words in the training Data.

5.2 Implementation:

For our implementation, we used a modified Distil-BERT model, which is a smaller, faster, and cheaper version of a BERT model. Specifically, we used distilbert-base-uncased from the Huggingface Transformers library. To come to this model, we first considered the training budgets that we had, which was $50 on Google Cloud Platform. Thus, we knew we wanted a lightweight model that would still be able to produce accurate results without significant loss. After reading the paper *DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter* [9], we learned that the Distil-BERT model is 40% smaller than the size a BERT model. Despite this reduction, the Distil-BERT model retains 97% of its language understanding capabilities is 60% faster. We were willing to trade off the 3% accuracy in exchange for the large increase in speed.

To implement the Distil-BERT model, we tried various parameters for the test and train dataset split, number of epochs, cut off for labels (which we ended at 9000 as stated in the Dataset section), dropout layers, and activation type. It took various experiments to tweak the model so that it would best fit the healthcare data we are working with.

We originally trained the Distil-BERT model without filtering out any data and labels. However, we achieve an accuracy rate of 39.5% on the test set. After examining outputs, we found that it is not possible to accurately predict messages with labels that have less than 9000 data points since there is simply not enough data for the model to learn from. Thus, like our baseline model, we filtered out labels that have less than 9000 messages. After various changes, we trained our optimal model on Google Cloud Platform for about 6 hours.

5.3 Results:

Overall Accuracy: 0.70.3%
Mathew's Correlation Coefficient: 0.59

Label Type	Precision	Recall	F1-Score
Medical Assistant	0.73	0.50	0.6
Nurse Practitioner	0.76	0.54	0.63
Physician	0.70	0.83	0.76
Physician Assistant	0.69	0.5	0.58
Registered Nurse	0.66	0.75	0.70
Stanford Referring Physician	0.79	0.60	0.68

Analysis is listed in section 7

6 Evaluation Metrics

In order to test the success of our system, we plan to use a standard approach of splitting our data into train,

validation, and test sets. We will train the model with train and validation sets. Then, we will evaluate the accuracy of our model with the test set to see how well our model can generalize to unseen outputs. We will use the following formulas to understand to understand these outputs in a more comprehensive manner. These metrics work best for our problem because they encompass all the differences between true and false assignments. They also provide more ways of discussing the sensitivity of the model.

6.1 F1-Score

The F1-Score [12] is a way of seeing the mean of precision and recall. This is a more comprehensive way of checking whether the model is performing optimally as it provides more information regarding not only the times the model is correct, but the number of times the model is making a classification prediction. It also shows us how precision and recall are balanced over the model. This is a useful metric as it can provide more information then accuracy when a model has data that is unevenly distributed amongst classes, like in our case.

6.2 Precision

Precision [12] is the number of true positives over all positive classifications. Meaning it addresses the number of the times a model predicts correctly for a given class.

6.3 Recall

Recall [12] is the number of times the model predicted correctly out of all the true labels that were in the data set. This gives us an idea of how the model performed for a class, and gives us a metric of understanding how often the model is missing positive classifications.

6.4 Matthew's Correlation Coefficient

Mathew's Correlation Coefficient [10] is a formula which helps compress and provide a singular value to describe the confusion matrix of a model. In other words, we will use this matrix to assess the true positives and negatives as well as the false positives and negatives for how the model assigns notification status for the message inputs in our data set. We want a high Mathew's correlation coefficient score as this indicates a more precise model. Using this information, we can determine how well our model assigns messages to provider type.

6.5 Receiver Characteristic Graphs (ROC)

We will use ROC graphs [11] because they help describe the threshold for model sensitivity as well as the risk of false alarm, we can see whether our model is performing well in terms of not over assigning or under-assigning messages. Likewise, the ROC graph also provides an indicator for the cumulative distribution function which will give us the cumulative distribution function, which will tell us the probability of detection or false alarm. We want our ROC curve to stay above the line y=x which is the ROC rate of a random model. We would like our model to be more accurate than random assignments and aim for as close to perfect classification as possible.

7 Results Analysis

This section gives result analysis for our main approach. For more information on the baseline, see section 4.2.

1) **Broad Evaluators:**
   a) **Accuracy Score: 70%** We were pleased with our results for multiple reasons. For one, our model reached a bare standard of 70% accuracy. Given that this is a 6 class multi-class model, we were happy to achieve a high test accuracy. To us this was a first significant indicator that our software could be helpful to the problem at hand for Stanford Medical center. However, we were very aware of how a raw accuracy score can be deceptive when data is imbalanced. Therefore, we looked to additional metrics to help us understand how our model was performing.
   b) **Matthew's Correlation Coefficient: 0.59** Matthew's correlation coefficient encompasses information contained in the general confusion matrix, indicating whether the model is accurately associating the inputs with their correct labels. The range for a correlation coefficient is 0-1, 1 being the model is entirely fluent with the data. This gave us a more tempered estimate of our model performance in comparison to our accuracy score. Overall, it seems that our model is doing fairly well, it is seeing correlations and giving correct results most of the time. However, given that this model is ideally aimed for real-world use, we see that there is more need and room to optimize. Given our goals for this quarter, this metric definitely indicates that finding correlations given our data set is possible and that more intricate and optimized models are worth pursuing.
2) **Granular Analysis**
   a) **Medical Assistant:** Medical assistants had a high precision rate, but a low recall rate. This is reflected in the f1 score as well. This indicated that our model does not often classify for medical assistants, but when it does, it usually does so correctly.
   b) **Nurse Practitioners:** Like Medical assistants, this class had higher precision than recall, but higher rates for both. This reflects in the f1-score. Overall, this shows that while our model didn't classify all instances for this class, when it did it primarily did so correctly. It also had fewer cases where it did not classify, as shown by recall.
   c) **Physician:** The physician class had high rates of recall and a fairly high precision. This gave it the highest f1 score. Overall, this means our model was very well tuned to determining cases where a physician was needed and had very few instances of false positives or false negatives.

d) **Physician Assistant:** Physician Assistants had fairly low rates of both precision and recall amongst our classes. This also yielded a low f1 score. This means there were many false positive and false negative cases for this class.
e) **Registered Nurse:** This class had higher recall than precision. This meant that this class does well at finding all the correct classifications, but that it may over classify for other classes mistakenly for this one. However, overall, it is fairly high scores for both, giving it the second highest f1 score of all the classes.
f) **Registered Nurse:** This class had higher precision that it did recall. This means that it may have missed some of its classifications, but for the positive labels it provided, it was correct and precise. This gave this class a moderately high f1 score.
g) **Overall patterns: The model was best at classifying registered nurses and physicians. The model had a difficult time distinguishing Medical Assistants, Nurse Practitioners, and physician assistants.**

3) Note about ROC curves: Given the timing of our assignment and the limited GCP credits we had. We were unable to generate ROC graphs for each class. It was our plan to make additional ROC curves for each class using a One over rest approach. However, we felt that the metrics we used provided sufficient insight into our model performance, and it is likely that ROC would have provided similar information. If given more time, we would have liked to do this on a more optimized version of our model.

8 Error Analysis

We recognize that our model worked best on identifying the need for physicians and registered nurses. We think this might be because of the language use and the general perception that patients and others might have when writing messages. For instance, a patient might clearly say I need to see a doctor, or I would like to get information from my nurse. However, the other classes like physician assistant, medical assistant, etc are not always widely known to a layman and there may not have been language distinguishing their roles very frequently. Another possible explanation could be that the remaining categories might have had more overlap in roles. For instance, medical assistants and physician assistants may complete some of the same tasks for a patient given the specialty, the number of employees on staff, etc. We think this might contribute to why it was harder for our model to accurately distinguish which messages needed these professionals specifically. Nurses and doctors likely have more specific characteristics that overlap less with other categories.
Overall, we also believe some of our errors could be due to disproportionate appearances of labels in our data set (for instance it could be that physician was just most common in our data set and that influenced our weights. Additionally, because we received a mix message types and requests, we're not entirely sure exactly what factors might've led to errors. For instance, if we had more outpatient messages than inpatient, this could lead to different requirements and needs that the model may have picked up on.

9 Future Work

Our plans moving forward are to take the following steps:

1) With the amount of GCP credits and limited time we were given, we were only able to train the model on 361,649 message data points. However, given more resources and time, we would like obtain, clean, and train the model on the millions of message data points that the Stanford Health Care team has. We hypothesize that this would allow us to increase the amount of labels we could take in since there will be more data points for each label. With more data, it would also likely make our model more accurate.
2) We would like to continue investigating other types of BERT and BERT specific optimizations that we can do to improve our model. We think that post restructuring the output classes, implementing these improvements should get us to higher accuracy for classification.
3) If time allows and we have machine capacity, we might try other modifications including changing our epochs for more than 2.
4) A significant challenge for us during this project was obtaining data. This was a multi-week process of going through training, pushing for access, understanding the data, and cleaning the data. Since this project has not been done before, we understand the significant amount of time it took to get the data. In the future, we hope that there is a more streamlined process to access healthcare data in a safe and ethical manner so that computer science methods can be applied to improve healthcare.

10 Code

https://github.com/cgmoffitt/patient-message-classification/

11 Video

$https://drive.google.com/file/d/1oztaQdNHgmyQZw9hhMBV6_Wn4rtyba/view?usp=sharing$

12 Works Cited

References

[1] J. Zhang, K. Kowsari, J. H. Harrison, J. M. Lobo and L. E. Barnes, "Patient2Vec: A Personalized Interpretable Deep Representation of the Longitudinal Electronic Health Record," in IEEE Access, vol. 6, pp. 65333-65346, 2018, doi: 10.1109/ACCESS.2018.2875677.
[2] Mastorakos, George, et al. "Probing Patient Messages Enhanced by Natural Language Processing: A Top-down Message Corpus Analysis." Health Data Science, vol. 2021, 2021, pp. 1–10., https://doi.org/10.34133/2021/1504854.

[3] Manage your care with myhealth. Stanford Health Care (SHC) - Stanford Medical Center. (n.d.). Retrieved June 1, 2022, from https://stanfordhealthcare.org/for-patients-visitors/myhealth.html
[4] Genkins, Julian. Clinical Informatics at Stanford Health Care. (2022, May 15). Personal communication [Personal interview].
[5] Genism. (2022, May 6). Gensim: Topic modelling for humans. models.doc2vec – Doc2vec paragraph embeddings - gensim. Retrieved June 1, 2022, from https://radimrehurek.com/gensim/models/doc2vec.html
[6] Huggin Face. (n.d.). Distilbert. Distil-BERT. Retrieved June 1, 2022, from https://huggingface.co/docs/transformers/model_doc/distilbert
[7] Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.
[8] Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32 (ICML'14). JMLR.org, II–1188–II–1196.
[9] Sanh, V., Debut, L., Chaumond, J., amp; Wolf, T. (2020, March 1). Distilbert, a distilled version of Bert: Smaller, faster, cheaper and lighter. arXiv.org. Retrieved June 1, 2022, from https://arxiv.org/abs/1910.01108
[10] Chicco, D., Jurman, G. The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation. BMC Genomics 21, 6 (2020). https://doi.org/10.1186/s12864-019-6413-7
[11] Google. (n.d.). Classification: Roc curve and AUC nbsp;—nbsp; machine learning crash course nbsp;—nbsp; google developers. Google. Retrieved June 1, 2022, from https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc: :text=An
[] "The F1 score", Medium, 2022. [Online]. Available: https://towardsdatascience.com/the-f1-score-bec2bbc38aa6. [Accessed: 03- Jun- 2022].