Extract Patient Voice from Social Media Posts

Binary Text Classification of Unlabeled Data on Twitter Corpus

Laxmi Prasanthi Desiraju

School of Professional Studies, Northwestern University

Prasanthi.Desiraju@gmail.com

May 2022

1. Abstract

The advent of deep learning in recent years has introduced many state-of-the-art algorithms in Natural Language Processing. But they all come at a cost, models requiring huge amounts of training data. This creates a bottleneck in leveraging these solutions for most real-world applications where we do not have such readily available data. This paper explores ways to handle this problem while attempting to solve a particular use case within the health care industry - filtering patient experiences from social media websites. We demonstrate two data labeling techniques, namely Weak Supervision and Active Learning to create a huge, labeled corpus. Experiments to quantify the quality of this generated training dataset to build a binary classification model on BERT and BioBERT architectures achieve an accuracy of 87% and 89%. Our study shows that by providing minimal annotated data, these labeling techniques can generalize well to create a huge, labeled dataset to help leverage the best models from the literature.

Keywords: Data Labeling, Weak Supervision, Active Learning, BERT, BioBERT

2. Introduction

As social media platforms grow, the world around us is witnessing a rapid increase in textual data, data that conveys people’s stories and experiences on varying subjects, including medical experiences. Streamlining this information can help pharmaceutical companies curate various insights such as patient experiences, views, and symptoms that help gain leverage over competitors. With this motivation, we aim to utilize AI to create a classification model to extract patient-specific information from the digital haystack of social media data and address the issues in due process.

This paper focuses on information extracted from Twitter, a microblogging platform where users can update their views in text format consisting of 140 characters or less. These posts, called tweets, could be on any topic, and in this paper, we focus on the tweets related to specific auto-immune disease drugs. We extract required data using Twitter API by passing drug names as search terms. Given the sheer volume of tweets, the data can contain tweets where patients share their experiences, questions, or concerns and off-topic tweets such as recent research articles and information by/on drug manufacturers. We classify these tweets into two groups, 'Patient' and 'Others.' While this seems to be a simple binary classification problem, the main problem we have at hand is that none of this data is labeled. We have huge amounts of tweets but no reference examples (training data) to feed a machine learning algorithm to learn from and develop a classification model. If we intend to use SOTA algorithms like BERT, one will have to annotate thousands of data points (sentences) manually. This study examines the following:

- Data Labeling: We look at ways using weak supervision and active learning that help reduce manual annotation efforts and come up with functionalities to create labeled datasets starting by annotating a minimal number of records. Manual annotation of a few records can help us compare the performance of our models and tweak the data labeling as required. Successful implementation of this feature can help save a lot of manual effort and time.
- Classification Model: Create a binary classification model using deep neural networks using the labeled training data created using the above techniques. We hope this final model will help seamlessly classify data related to any auto-immune disease and not just on the drugs/disease it is trained on. We chose to use BERT and its slightly

modified version BioBERT trained on the medical domain, for classification as they have shown promising results in the keys area of the NLP domain.

3. Literature Review

The use of AI in healthcare is drawing a slew of research opportunities, with one of the key applications to provide patient-centric services. It is imperative to identify the right data that convey patients' experiences to conduct such healthcare analytics. This seems to be an active research area, and we have a few papers published along similar lines (Jiang et al., 2018; Alex et al., 2021). Using deep learning to achieve the results appears effective, but as discussed earlier, the main challenge is the lack of labeled data, and most of the effort is spent manually annotating the data to feed into the deep learning models.

To reduce the burden of annotating data, several methods are proposed of which weak supervision and active learning have gained popularity over recent years. Weak supervision refers to training a model on a small dataset with noisy labels, created via heuristics or programmatic labeling based on specific patterns or rules. It essentially blends in knowledge from various sources, many of which are weak and of low quality (Ratner et al., 2017). Active learning a semi-supervised learning technique aims to train the model with a small dataset and teach the active learner by manually adding more samples at each iteration until we achieve the required accuracy (Jacobs et al., 2021). Compared to weak supervision, active learning still requires clean labels to perform better but uses more advanced query strategies to fetch additional samples for the learner model to generalize better. Few such techniques include Prediction Entropy, Breaking Ties, Random Sampling among others (Small-Text, n.d).

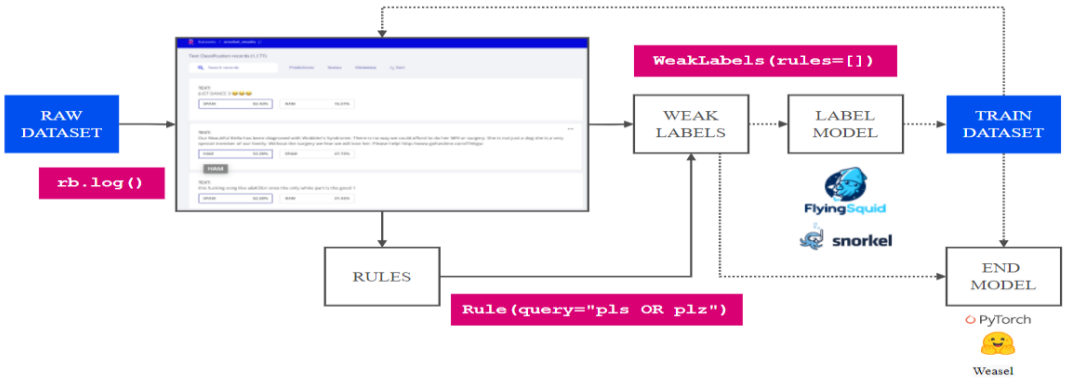

Figure 1. *Weak supervision with Rubrix*
Copyright: Rubrix (https://rubrix.readthedocs.io/en/master/tutorials/weak-supervision-with-rubrix.html)

Today, deep neural networks have proven to deliver high accuracies if we have the right data, and to verify this, we have several models that demonstrate state-of-the-art (SOTA) results for text classification tasks. NLP models generally rely on learning word representations using algorithms such as Word2Vec (Mikolov et al., 2013) and GloVe (Pennington et al., 2014) whose word embeddings represent context-independent learning. This has been overtaken by transformers (Vaswani et al., 2017), techniques that focus on learning context-based word representations. The success of BERT (Devlin et al., 2019), a first of its kind based on transformer architecture has revolutionized the NLP space ever since its publication, and there were many advances made in a variety of tasks, from question-answering to sentiment analysis. BERT is a general-purpose language representation model trained on English Wikipedia and Brown Corpus and thus is so well trained that it can be applied on any new data sets and still get good performance. However, since we are dealing with medical text, we would additionally explore BioBERT (Lee et al., 2019), a domain-specific BERT trained on biomedical corpora containing PubMed Extracts and PMC articles.

4. Data

For experiments described in this paper, tweets relevant to treatment drugs prescribed for auto-immune disease, Multiple Sclerosis are extracted from Twitter. The names of specific drugs were given by a subject matter expert based on their relevance. These search terms will help restrict the data we are interested in. There are multiple APIs that interface with Twitter to extract attributes such as text expressed by the user (tweet), user Id, the language of the tweet, date and time of when it was tweeted, etc. Here we are interested only in the tweet and determine whether they are relevant to patients, and any mention of data in the following steps refers to this tweet/text information.

4.1 Data Collection

We used Snscrape (Snscrape, 2021) to collect historical tweets based on the given search terms. Drugs prescribed for 'Multiple Sclerosis' were given as input to retrieve relevant tweets posted between Jan 1, 2019 – Mar 1, 2022, for up to 20k tweets for each drug, the latest tweets first.

```
multipleSclerosis=['cladribine','mavenclad','aubagio','teriflunomide','tecfidera','dimethyl fumarate',
                   'dimethylfumarate', 'gilenya','fingolimod','mayzent','siponimod','tysabri','natalizumab',
                   'ocrevus','ocrelizumab','lemtrada','alemtuzumab']
dateRange   = ' since:2019-01-01 until:2022-03-01'
```

Figure 2. *Search Terms used in Paper*

4.2 Data Preparation

We retrieved around 121k tweets using the API. The next step is to perform exploratory data analysis (EDA), and the following are the series of steps performed to get more refined data, followed by a tabular illustration. Data extracted contain tweets in languages other than English, and such records are dropped from the list. However, it would be a good idea to retain such tweets and use python libraries such as Googletrans to translate the data if we were to gain any

insights based on the geographic location. We dropped the data further by remove the tweets that do not contain the drug names we used.

We removed all the redundant information such as emojis, smileys, mentions, and URLs. Additionally, the text extracted has HTML content to represent special characters such as '&amp' for &, '&quote' for '' converted to general text using 'Beautiful Soup.' After examining the data, we retained the hashtags as they contain relevant information, such as drug names and symptoms. We then dropped the tweets, which are just duplicates, because of retweets. Finally, we checked the word count of the tweets and, after manually examining the records, we dropped the tweets that contain six or less words, as they do did carry much meaningful information.

EDA Steps	Tweets at the beginning step	Tweets at the end of the step	Tweets Dropped	Library Used
Extract Tweets using API	0	120,883		Snscrape
Remove non-English Tweets	120,883	91,796	28,907	
Remove tweets that do not contain specified drug names	91,796	89, 526	2,270	
Remove URLs, mentions, emojis, and smileys	89,526	89,526	0	tweet-preprocessor
Decode HTML to general text (ex: '&amp' to &)	89,526	89,526	0	Beautiful Soup
Remove Duplicate Tweets	89,526	64,705	24,821	
Remove short tweets, with a word count of less than 6.	64,705	63,321	1,384	

Table 1. *EDA on extracted tweets*

This concludes the data preparation step, and the next steps include explore data labeling techniques to label the data and perform binary classification. We retain stop words and do not perform any stemming or lemmatization to ensure the data is legible for manual annotation.

4.3 Manual Annotation

Given the volume of the dataset, manual annotation is a tedious and time-consuming task. We still proceed with manually annotating a few records to understand the dataset and which can be used as a test dataset. Our dataset contains 63,321 records, and we start by annotating around 5% (3200 records) of the data and assigning one of the two labels to each data point.

1. 'Patient': Any tweet that describes patients' questions, concerns, or opinions.

   Example: *Took my first dose after eating at am. Im relieved its done. Feeling a bit nauseous but nothing major and I have a headache but Im trying to stay hydrated.*

2. 'Other': Any Tweet that does not satisfy the above criteria.

   Example: *Merck to acquire rights to develop cladribine for myasthenia gravis, NMOSD*

3200 records are manually annotated by going through each tweet and categorizing them as Patient or Other to the best of human judgment. Calculating even a minute to annotate each record, the whole process by itself consumes around 54 hours, and this clearly states the challenge with manual annotation as the data complexity and volume increase. In the following sections, we try to use this minimal labeled data to create a more extensive training dataset to feed into the neural network models.

Mode	No.Examples	Patient Voice	Other
Unlabeled Data	60121	Unknown	Uknown
Labeled Data (Manual Annotation)	3200	1270	1930

Table 2. *Final Data Split*

5. Method

The first step is to create labels for our training dataset with minimal examples, and here we illustrate two approaches to achieve it. We use Rubrix, a human-in-the-loop python framework, to iterate through data and create labels.

5.1 Weak Supervision for Data Labeling

We use labeling functions (LFs) to express weak supervision based on specific patterns or heuristics. Here we define labeling functions that return a label ‘Patient’ or ‘Other,’ based on pattern matching. Manually annotated tweets help measure the coverage and performance of these rules. With multiple rules, there is a possibility to assign different labels to the same tweet, and to denoise such weak labels, we can take several ML approaches such as Majority Voter (sklearn.ensemeble.VotingClassifier) or even train a neural network with the help of manually annotated data and assign weights to these rules (Ratner, 2017). This study used the second approach, Snorkel, to create weak labels for our training dataset (Appendix A).

	label	coverage	annotated_coverage	overlaps	conflicts	correct	incorrect	precision
(Merck KGaA) OR (European Commission) OR (EMD Serono) OR (PubMed) OR (Clinical) OR (Research) OR (Scientists) OR (*Trail*) OR (Session) OR (Study)	{Other}	0.08	0.08	0.08	0.01	170	12	0.93
(I OR MY OR MINE OR OWN OR ME) AND (*FEEL* OR *BETTER* OR 'SYMPTOMS' OR *WORSE* OR *REPLASE* OR *PROBLEM* OR 'HAVE' OR 'HAD' OR 'DEVELOP')	{Patient}	0.07	0.06	0.07	0.02	128	14	0.90
patient	{Other}	0.10	0.10	0.10	0.01	208	15	0.93
FDA	{Other}	0.04	0.04	0.04	0.00	87	3	0.97
I OR MY OR MINE OR OWN OR ME	{Patient}	0.40	0.39	0.07	0.02	802	76	0.91
NOT (I OR MY OR MINE OR OWN OR ME)	{Other}	0.60	0.61	0.19	0.01	1272	90	0.93
total	{Other, Patient}	1.00	1.00	0.26	0.03	2667	210	0.93

Figure 3. *Rules-based on Patterns - Weak Supervision - Label 1(Patient), 0(Other)*

5.2 Data Labeling with Active Learning

Active learning is iterative supervised learning, in which the learning algorithm can interactively query a user to label new data points. To train the model, we start by labeling a small sample, say 100 tweets, and then we iteratively improve it by adding a few more examples (20 tweets) every time to help it learn. The active learning process contains an initialization strategy and query strategy (Schröder et al. 2022).

- Initialization: Instantiate the active learner; for this, we need to convert raw text data into a format similar to any transformer-based classification. Here we use Hugging Face pretrained bert-base-uncased model as a base model for active learner.
- Query Strategy: There are several ways to query the data for iterative steps, and here we use Prediction Entropy (Small-Text, n.d.), which prioritizes the variables/sentences for which the model is most uncertain about the outcome class. An Oracle (manual annotator) provides labels for these queries, and the process is repeated iteratively until the desired performance is achieved.

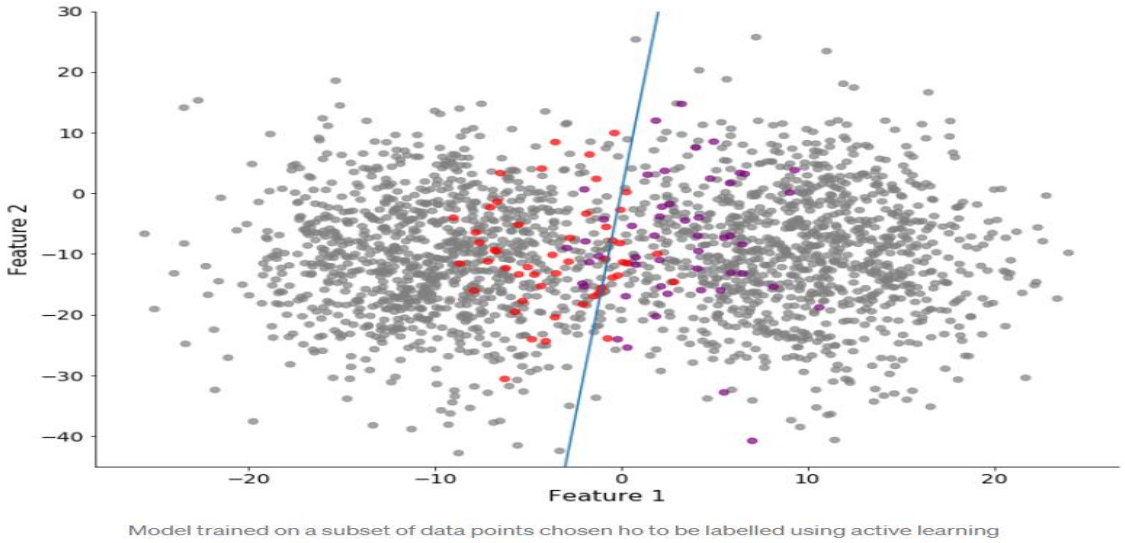

Figure 4. *Example model using Active Learning, starting by labeling a few data points* Copyright: Ana Solaguren-Beascoa, (https://towardsdatascience.com/active-learning-in-machine-learning-525e61be16e5)

5.3 Binary Classification Model

Once we have labeled data created with the above two models, we train a downstream classification model with BERT and BioBERT and evaluate the performance of each model. We used the pre-trained model's weights and fine-tuned them on our custom dataset to build the classification models.

6. Experiments

6.1 Weak Supervision with Snorkel - Rule-Based Classification

This is a programmatic labeling technique enhanced with neural networks where we come up with some rules based on which a label is assigned to the text. We have come up with a few rules after examining the data at random and based on generic linguistic features (Appendix A). We used Snorkel, a data-centric AI application that can help generate probabilistic labels based on the rule classifier, where 70% of the labeled data(2240 records) is used to train the rule classifier and performed the test using the remaining 30% labeled data (960

records). We ended up with an accuracy of 93%, which is pretty good given the simplicity of the rules.

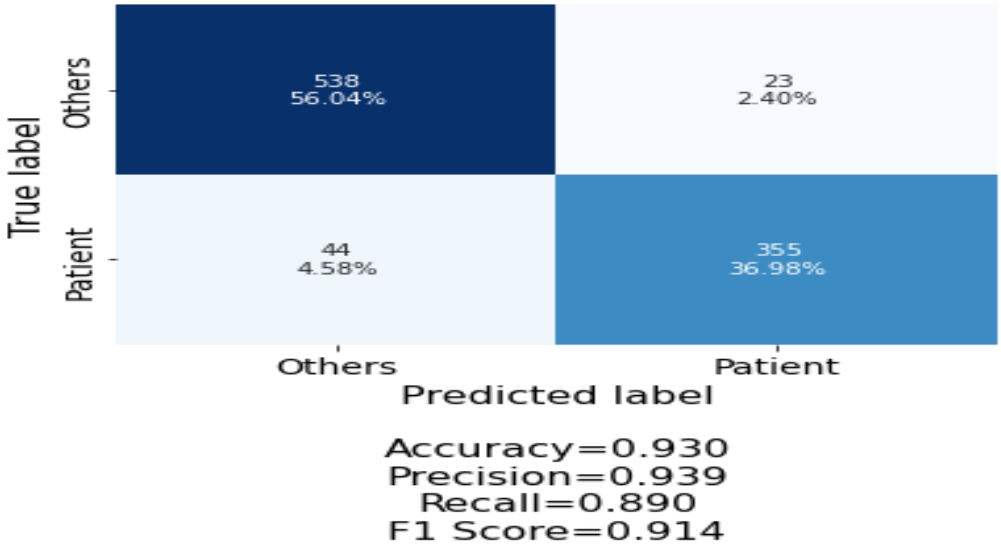

Figure 5. *Confusion Matrix - Model with weak supervision*

Once we reach an acceptable accuracy or the metrics of interest (precision, F1-score), etc., we can use the model to generate labels for the unlabeled data. We can further revise the labels manually for low predictions with low probability, which will now be very minimal to creating the final training dataset, which can be used in the final binary classification model.

	CleanText	Pred1	Pred2	LabelPredText
When your body is still throwing a hissy fit and vomiting every day week post lemtrada you know just because glitterbrainproblems On the upside being housebound has meant more booking adventures for later in the year		0.71	0.29	Other
	day to lemtrada feeling nervous now	0.71	0.29	Other

Table 3: *Couple of Examples with Incorrect Label and Low Confidence Prediction*

6.2. Active Learning

6.2.1 Active Leaning with Multinomial NB

We used a multinomial Naive Bayes classifier to classify text, and the features for the classifier will be the counts of different word n-grams (n =1,2,3). However, the model did not perform that well in classifying the data as there is no improvement in accuracy even with multiple

iterations (Figure 6). We believe the reason could be the usage of the English model. Since we rely on tweets made in an informal tone, the bag of words model might not classify the text correctly due to missing context and grammatical structure across all tweets. We drop this model given there is no significant improvement in train accuracy.

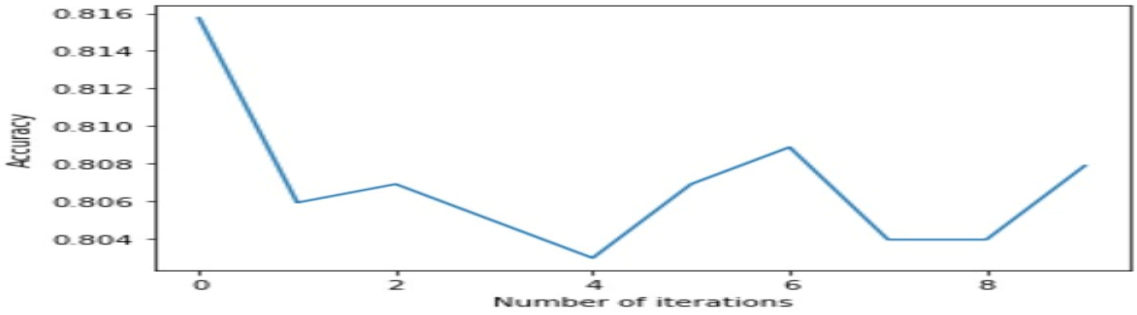

Figure 6: *Active Learning - Model Accuracy Over Iterations*

6.2.2 Active Learning with small-text

As an alternative, we want to explore active learning with the well-proven novel architecture, Transformers. We use small-text as it enables us to apply active learning by integrating with transformers robustly and quickly. As opposed to weak supervision, we need to have at least a few training examples to initialize the model. As the documentation suggests, few instances are sufficient to initialize the model; we pick around 30% (1000 records) to train the active learning model and evaluate it using 2200 records.

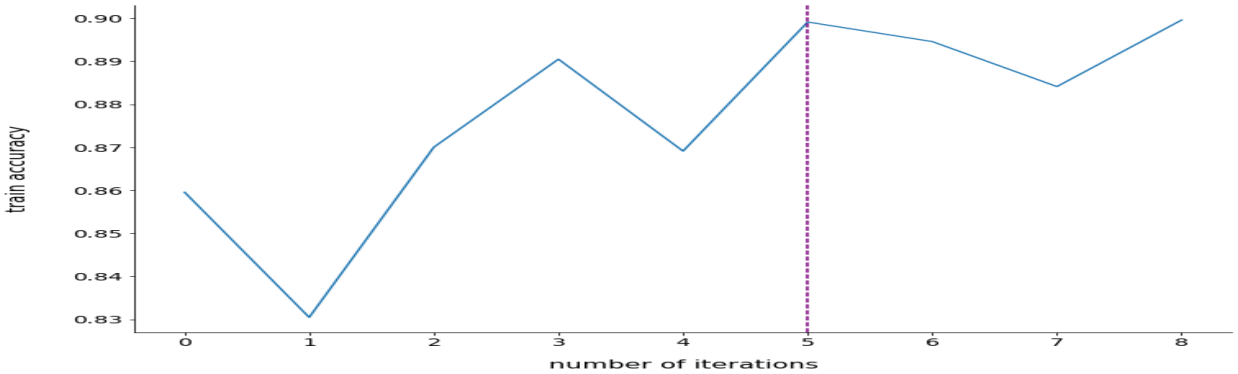

Figure 7: *Improvement of Active Learner – Train Accuracy. Line at 5 indicates stop criteria to avoid overfitting.*

The model has performed well, with an accuracy of 91%, which is quite promising as the only effort required here is to annotate a few samples. Here we have the confusion matrix illustrating some of the metrics for the model.

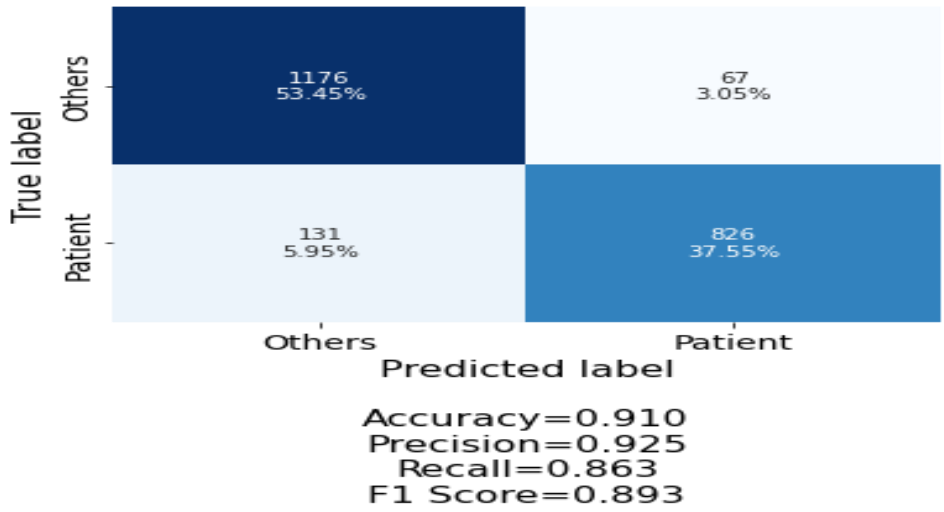

Figure 8: *Metrics on Unseen dataset (Transformer based AL)*

6.3 Binary Classification

6.3.1 BERT Model

BERT models are pre-trained on a large text corpus, enabling powerful text representations. While it is costly to train such a model, we can download the pre-trained weights and encodings from the saved model. The code is implemented with python TensorFlow library, and we used Adam optimizer per the recommendation. In this experiment, we use these encodings to create numeric word representations for our sentences, and the model is fine-tuned on the training data created using a labeled dataset generated from Active learning (6.2.1).

The labeled dataset is split into training and validation datasets each containing records, respectively. The model is trained on four epochs and achieved a training accuracy of 93.2% and a validation accuracy of 94%. The following figure demonstrates the model's performance on the initial manually annotated data of 3200 records with an accuracy of 86.9%

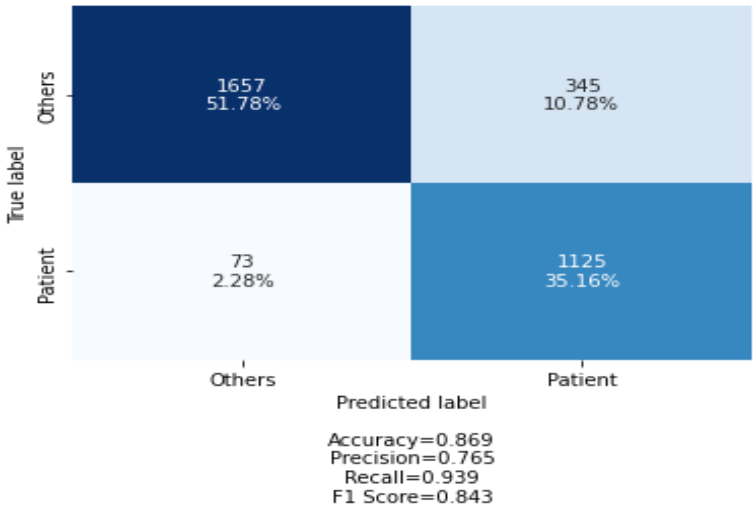

Figure 9: *Metrics on Test Dataset (Bert Model)*

6.3.2 BioBERT Model

As we are dealing with medical text, it seemed to be a good idea to experiment with at least one of the medical domain variants of BERT. We will be fine-tuning the pre-trained weights of the BioBert model for this text classification task at hand. We used Pytorch for BioBert to work with Hugging Face API and picked BertAdam optimizer closer to the optimizer we used in the above model with TensorFlow. We split the data between training and validation datasets and are converted into tensors as required by the model. The model is run for four epochs on a batch size of 16.

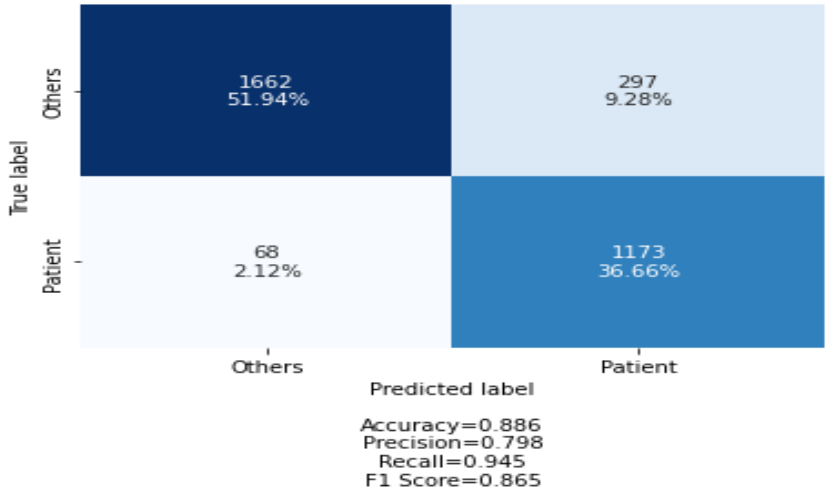

Figure 10: *Metrics on Test Dataset (BioBert Model)*

7. Results

Data Labeling Techniques

Experiment	Annotated Data Fed to Model	Test Examples	Test Accuracy	Comments
Weak Supervision	2240	960	92.0	Based on rules and performs better with more labeled data fed to the system. Also, we could not test on more records given the limited annotated records.
Transformed Based Active Learning	1000	2200	91.0	This model is selected to build our training dataset, given no additional rules are required, except for annotating a few records.

Table 4: *Metrics of Data Labeling Techniques*

Binary Classification Models

Experiment	Optimizer	Train Examples	Validation Examples	Test Examples (Manually Annotated Data)	Train Accuracy	Validation Accuracy	Test Accuracy
BERT, 4 epochs	Adam	45,090	15,031	3,200	93.27	94.0	86.94
BioBERT, 4 epochs	BertAdam	45,090	15,031	3,200	96.20	95.0	88.60

Table 5: *Metrics of Binary Classification Models (Accuracy)*

Experiment	Accuracy	Precision	Recall	F1 Score
BERT	86.9	76.5	93.9	84.3
BioBERT	88.6	79.8	94.5	86.5

Table 6: *Additional Metrics for Binary Classification*

As expected BioBert model outperformed BERT, given it has better word representations for a medical domain. The above results show that by just annotating 3200 records, we are able to create a model that can classify with an accuracy of 88.6%. Both models have better recall, which is good because the system was able to identify most of the patient-relevant data, and which metric is relevant depends on the downstream NLP tasks that consume this patient-relevant data.

8. Future Work

This study explores different techniques on baseline models without any hyperparameter training. We also assume the labels generated by the labeling techniques are correct and as the next steps, we would like to correct some of those labels based on the confidence levels and annotate them. The results indicate there is some overfitting involved, given the discrepancy between validation and test accuracy. Hence, we like would also continue working with different hyperparameters, like learning rate, batch size, and optimizers before we conclude the final model. Once, we have a final working model, we believe it opens various possibilities for creating downstream NLP pipelines for medical sentiment analysis, classifying patient-specific

data into sub-classes (questions, symptoms), or conducting specific patient-based analysis that helps deliver precision medicine.

9. Conclusion

In this paper, we suggest ways to generate labeled data to utilize the state-of-the-art algorithms without requiring spending hours manually annotating the data. The results demonstrate than Active Learning and Weak Supervision techniques significantly reduce the cost and difficulty of generating hand-labeled datasets. These techniques are very much applicable to any domain where labeled is difficult to procure, a very common scenario for most real-world applications. The binary classification model's performance advocates that these techniques are robust and help leverage any deep learning models without having to worry about having a huge, labeled dataset and focus on building research-oriented use cases that add value.

10. Acknowledgments

I am thankful to Mr. Prathamesh Kalmkar for guiding me through possible business use-cases of AI in the healthcare sector and the frequent challenges encountered while implementing ML solutions at an enterprise level. His support has been instrumental in combining and developing the ideas outlined in the paper. I extend my thanks to Dr. Alianna J. Maren for providing her valuable comments on restructuring the work and highlighting important aspects. In addition, I would like to thank Riddhi Gandhi, Vivian Xia and Wing Chan for providing their suggestions on refining this paper. Lastly, I am grateful to the communities behind 'Rubrix ML,' an open-source production-ready python framework that made it easy to explore the ideas presented in the paper. Additional resources mentioned are in Appendix B that helped formulate the experiments conducted in this study.

11. References

Alex, B., Whyte, D., Duma, D., Owen, R. E., and Fairley, E. 2021. "Classifying patient and professional voice in social media health posts." *BMC Med Inform Decis Mak* 21 (1): 244. doi: 10.1186/s12911-021-01577-9. https://doi.org/10.1186/s12911-021-01577-9

Devlin, J., Chang, M. W., Lee, K., and Toutanova, K. 2019. "Bert: Pre-training of deep bidirectional transformers for language understanding." doi: 10.18653/v1/N19-1423. https://doi.org/10.18653/v1/N19-1423

Jacobs, P. F., Maillette De Buy Wenniger, G., Wiering, M., and Schomaker, L. 2021. "Active Learning for Reducing Labeling Effort in Text Classification Tasks." *Benelux Conference on Artificial Intelligence* 1530 : 3-29. doi: 10.1007/978-3-030-93842-0_1. https://doi.org/10.1007/978-3-030-93842-0_1

Jiang, K., Feng, S., Song, Q. *et al.* 2018. "Identifying tweets of personal health experience through word embedding and LSTM neural network." *BMC Bioinformatics 19* (210). doi: 10.1186/s12859-018-2198-y. https://doi.org/10.1186/s12859-018-2198-y

Lee, J., Yoon, W., Kim, S., Kim, D., Kim, S., So, C. H., and Kang, J. 2020. "BioBERT: a pre-trained biomedical language representation model for biomedical text mining." *Bioinformatics* 36 (4): 1234-1240. doi: 10.48550/arXiv.1901.08746. https://doi.org/10.48550/arXiv.1901.08746

Mikolov, T., Chen, K., Corrado, G., & Dean, J. 2013. "Efficient estimation of word representations in vector space." doi: 10.48550/arXiv.1301.3781. https://doi.org/10.48550/arXiv.1301.3781

Pennington, J., Socher, R., and Manning, C. D. 2014. “Glove: Global vectors for word representation.” doi: 10.3115/v1/D14-1162. https://doi.org/10.3115/v1/D14-1162

Ratner, A., S. H. Bach, H. Ehrenberg, J. Fries, S. Wu, and C. Ré. 2017. “Snorkel: Rapid training data creation with weak supervision”. *Proc VLDB Endowment* 11 (3): 269-282. doi: 10.14778/3157794.3157797. https://doi.org/10.14778/3157794.3157797

Ratner. A. *et al.* “An overview of weak supervision.” n.d. Accessed on April 28, 2022. https://www.snorkel.org/blog/weak-supervision

Rubrix. n.d. “Welcome to Rubrix - Rubrix 0.13 documentation.” Accessed on April 28, 2022. https://rubrix.readthedocs.io/en/stable/

Schroder, C., Muller, L., Niekler A., and Potthast, M. 2021. "Small-Text: Active Learning For Text Classification in Python". doi: 10.48550/arXiv.2107.10314. https://doi.org/10.48550/arXiv.2107.10314

Small-Text. n.d. “Small-Text — Small-Text Documentation". Accessed May 2022. https://small-text.readthedocs.io/en/latest/#

Snscrape. 2021. “Snscrape.” Accessed April 2022. https://github.com/JustAnotherArchivist/snscrape.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser,L. and Polosukhin, I. 2017. "Attention is all you need." *Advances in neural information processing systems*, 30. doi: 10.48550/arXiv.1706.03762. https://doi.org/10.48550/arXiv.1706.03762

Appendix A

There are many open-source libraries that help define labeling functions to automatically label the data like Snorkel, Skweak, Prodigy etc. A labeling function (LF) is a function which takes a data point and assigns a class to it based on some rules we define. These rules are defined based on our experience and domain knowledge. In this paper, we used Snorkel, which provides a unified framework to combine these LFs and implements an ML to prioritize these LFs (assign weights) to generate a unique label for each data point. The quality of the weak supervision model depends on the quality of these labeling functions and who well they cover the patterns of the unlabeled dataset.

We tried to incorporate as many labeling functions as we can by manually examining the data. Most of these rules are based on linguistics, for example, someone talking about their symptoms would typically use personal singular pronouns. That helps create a labeling function that assigns ‘Patient’ to a sentence having such pronouns and ‘Other’ to the one missing such pronoun. We call this weak label because while the system can always pick the true positive (real patient experience), it also tends to pick a lot of false positives. For example:

> S1: My symptoms are much better after taking the drug ‘XXX.’ (True positive, Singular Pronoun, so system marks as ‘Patient’)
>
> S2: I was part of XXX drug research,; moreresults to follow soon. (False Positive, Singular Pronoun, the system still marks as ‘Patient’, even if it’s not a patient-relevant tweet).

So, we need to come up with another rule, say if there are any words containing ‘research,’ ‘study’ or ‘Clinical trails’, mark is as “Other.” By defining this rule, S2 now has a conflict,

which is where the system uses training data to come up with appropriate weights for these models and assigns a label.

Below are the rules created for our weak supervision model if any of the search condition (we use elastic search) is met a label is assigned automatically. As mentioned, most of these rules are all based on noting the common terms and patterns within the data and we need to create as many labeling functions as possible to get total coverage of 100%.

An additional rule possible rule might be to collect all the possible symptoms associated with the disease and include it in the rule, to check if the tweet talks about the symptom. The more domain knowledge we add into the labeling functions, we can expect better and less noisy labels.

	label	coverage	annotated_coverage	overlaps	conflicts	correct	incorrect	precision
(Merck KGaA) OR (European Commission) OR (EMD Serono) OR (PubMed) OR (Clinical) OR (Research) OR (Scientists) OR (*Trail*) OR (Session) OR (Study)	{Other}	0.08	0.08	0.08	0.01	170	12	0.93
(I OR MY OR MINE OR OWN OR ME) AND (*FEEL* OR *BETTER* OR 'SYMPTOMS' OR *WORSE* OR *REPLASE* OR *PROBLEM* OR 'HAVE' OR 'HAD' OR 'DEVELOP')	{Patient}	0.07	0.06	0.07	0.02	128	14	0.90
patient	{Other}	0.10	0.10	0.10	0.01	208	15	0.93
FDA	{Other}	0.04	0.04	0.04	0.00	87	3	0.97
I OR MY OR MINE OR OWN OR ME	{Patient}	0.40	0.39	0.07	0.02	802	76	0.91
NOT (I OR MY OR MINE OR OWN OR ME)	{Other}	0.60	0.61	0.19	0.01	1272	90	0.93
total	{Other, Patient}	1.00	1.00	0.26	0.03	2667	210	0.93

Once we have these labels, the model is trained on a training dataset to help the system learn in an unsupervised way to know what rules to be applied when. Of course, one disadvantage is we have no control over tuning the ML that runs behind the underlying ML model when used with Snorkel.

APPENDIX B

Code related to all the experiments can be found in the github link

https://github.com/PrasanthiDesiraju/TextClassification-of-Unlabeled-Data

I would like to extend my thanks and acknowledge multiple open source libraries and guiding tutorials from which the above code has been adapted and refined.

1. Dennis, T., Utility Function For Confusion Matrix
   https://github.com/neheller/isic18/blob/master/make_confusion_matrix.py
2. Rubrix, Active Learning with ModAl
   https://rubrix.readthedocs.io/en/stable/tutorials/05-active_learning.html
3. Rubrix, Weak Supervision with Snorkel
   https://rubrix.readthedocs.io/en/stable/tutorials/weak-supervision-with-rubrix.html
4. Small-Text, Active Learning with Small-Text
   https://github.com/webis-de/small-text/blob/main/examples/notebooks/02-active-learning-with-stopping-criteria.ipynb
5. Drew Perkins, BioBert Implementation with PyTorch
   https://towardsdatascience.com/tagging-genes-and-proteins-with-biobert-c7b04fc6eb4f
6. CodeBasics, Text Classification with BERT
   https://github.com/codebasics/deep-learning-keras-tf-tutorial/blob/master/47_BERT_text_classification/BERT_email_classification-handle-imbalance.ipynb
7. Hugging Face, Pipelines and Examples
   https://huggingface.co/