Predicting the Response of Triple-Negative Breast Cancer Patients to Neoadjuvant Chemotherapy using Unstructured Text

Bryan Kim
Dept. of Computer Science
Stanford University
bskim96@stanford.edu

Eric Matsumoto
Dept. of Computer Science
Stanford University
ematsu@stanford.edu

Andrew Sharp
Dept. of Computer Science
Stanford University
awsharp@stanford.edu

1 Introduction

Triple-negative breast cancer is a comparatively rare form of breast cancer where the cancer is not fueled by the hormones estrogen and progesterone or the HER2 protein [20]. Many treatments for breast cancer typically target these growth factors, so triple-negative breast cancer is significantly more difficult to treat than other varieties. Often, triple-negative breast cancer patients are given neoadjuvant chemotherapy as a preliminary form of treatment before surgery [20]. A successful response to NAC is a good indicator of long-term survival, so the ability to predict early in a patient's treatment process how well they will respond to this therapy could provide a significant prediction of their long-term health and help doctors who are trying to decide on a patient's treatment [20]. Our goal was to predict how well patients with triple-negative breast cancer would respond to neoadjuvant chemotherapy (NAC) based on medical reports written before they underwent treatment. We implemented Naive Bayes, logistic regression, and RNN models which took pathology and radiology reports in the form of unstructured text as inputs and attempted to predict whether the patient later had a successful or unsuccessful response to NAC.

2 Related Work

Several groups have studied the problem of extracting information from unstructured clinical text reports. In [2], Ling et. al. studied a similar task to ours (predicting metastatic breast cancer recurrence) on a dataset consisting of medical records and clinical reports from the Oncoshare database and the California Cancer Registry. They used an NLP-based clinical text analysis package (CLEVER), leveraging domain knowledge to hand-engineer descriptive features from their text. Combining these features with other quantitative features from the medical records, they trained a logistic regression model with L2 regularization and were able to demonstrate reasonably good performance on a hand-labeled test set.

In [1], Banerjee et. al. attempted to predict BI-RADS classifications of breast cancer severity using labeled and unlabeled reports from Oncoshare and radTF. They also used domain knowledge, but in the context of an embedding-based approach, identifying key terms, learning context-aware vectors for each one, and combining them into report vectors. These report vectors were classified with a simple logistic regression model.

We also studied several survey papers reviewing a range of popular techniques for text classification tasks [3,5], as well as the FastText method for generating word embeddings [4], both of which we used to motivate our general approach.

Having seen these past works, we were interested in determining if it is possible to accurately predict patient outcomes (in our case for triple negative breast cancer patients) using a more general word embedding to sequence-based approach on the unstructured text. In other words, we were interested

in testing methods which do not require extensive pre-processing using medical domain knowledge, as was done in [1,2].

3 Dataset and Features

Our dataset consisted of a collection of 22167 anonymized pathology and 93696 anonymized radiology reports from breast cancer patients at the Stanford Medical Center. Due to privacy and patient confidentiality concerns, we are unable to share this data directly or give a publicly available citation. The reports consisted of unstructured text and ranged in length from $\sim$ 800 to $\sim$ 2000 words each. We also had access to the Oncoshare database [18], containing a variety of other medical record information on each of the patients described in the reports.

Using the auxiliary information from the Oncoshare database we filtered our dataset to include only reports from patients diagnosed with triple-negative breast cancer who received neoadjuvant chemotherapy (NAC). They were then labeled based on their response to NAC (if this information was available) or their inferred response based on survival duration (under the assumption that patients who survived for at least 10 years had responded well to the treatment and patients who survived for less than one year did not). Using this additional inferred label allowed us to expand the size of our labeled dataset. We partitioned our datasets into training and validation sets using a 90-10 split. Our dataset sizes (combined pathology and radiology reports) after filtering and labeling are shown below. The first value is the number of training examples, while the value in the parentheses is the number of validation examples:

	Negative Response	Positive Response
Base	410 (46)	152 (17)
w/ Inferred Response	2684 (298)	845 (94)

Having collected our dataset, we proceeded to pre-process the report texts by stripping non alphanumeric characters and extra whitespace, splitting numeric values as part of words, removing stopwords, normalizing, and segmenting incorrectly attached words. The word segmentation was done with the WordSeg package [14] using a corpus of common English words.

We then generated word embeddings to serve as a suitable representation of our word sequences to use as inputs to a sequence model. To learn these embeddings, we used a skip-gram variant of the FastText method [4], as implemented in the GenSim package [12]. We trained for 200 epochs on a corpus consisting of our full set of patient reports (including both triple-negative breast cancer patients and other breast cancer patients) and our final outputs were a vocabulary of 56042 embedding vectors, each of length 300. The FastText method, much like the commonly used Word2Vec method [19], is a neural network-based approach which attempts to learn embeddings which are predictive of neighboring words in each word's local context. FastText includes the additional refinement, however, of using sub-word information, which allows it to perform better on rare and out-of-vocab words. We suspected that this would be very beneficial in our use-case, as our reports contained a high proportion of rare words.

4 Methods

We began by using multinomial Naive Bayes and logistic regression as baseline models. The multinomial Naive Bayes model [11] assumes that documents are generated by first randomly selecting a class (in this case, reports on patients who either did or did not have a positive response to NAC) from some distribution and by then generating each word from a distribution conditioned on the document's class. It then finds the class probabilities ϕ_y and word probabilities $\phi_{k|y}$ that maximize the likelihood function of the training data:

$$\mathcal{L}(\phi_y, \phi_{k|y=0}, \phi_{k|y=1}) = \Pi_{i=1}^{n}(\Pi_{j=1}^{d_i} p(x_j^{(i)}|y; \phi_{k|y=0}, \phi_{k|y=1}))p(y^{(i)}; \phi_y) \tag{1}$$

Our other baseline model was logistic regression [10], which, unlike Naive Bayes, does not make assumptions about the distribution which the data comes from. Instead, it attempts to find the features (in our case words) which are most useful in distinguishing between the two classes. It does this

by finding the weights θ which maximize the following likelihood, where $h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$, the model's predicted probability that $y^{(i)} = 1$:

$$\mathcal{L}(\theta) = \Pi_{i=1}^{n}(h_\theta(x^{(i)})^{y^{(i)}}(1 - h_\theta(x^{(i)}))^{1-y^{(i)}} \tag{2}$$

For both of these baseline models, we also experimented with adding bi-grams (pairs of consecutive words) as features because these can store more context and therefore potentially include more meaningful information about the report's content.

We then implemented a recurrent neural network (RNN) [6], which again trains weights to maximize the discriminative likelihood function (2). Instead of using a separate feature for each word in the dictionary, the RNN takes the word embeddings of the words in the document as input. At each timestep, the RNN reads both the next word of the document and the output of the previous timestep, and the final output is put through a softmax layer to produce the predicted class probabilities. Our RNN has a single layer with size 128, and each node uses a $tanh$ activation function. The following is the forward pass computation for each node, where W, U, and b are the learned weights:

$$h_t = tanh(Wx_t + Uh_{t-1} + b) \tag{3}$$

We experimented with several modifications to our RNN, the first being long short-term memory (LSTM) [7]. This architecture is designed to perform better on long sequences. Because the RNN's final output comes at the end of the sequence, information from earlier in the sequence can be less likely to affect the predicted label. LSTMs attempt to fix this problem by using a memory cell to pass information on to the next timestep instead of only passing the output. At each timestep, the new input x_t and the previous output h_{t-1} and memory state c_{t-1} are put through several activation gates to determine which information should be kept and passed to the next timestep. The LSTM forward pass equations are as follows:

$$\begin{gathered} f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f), i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i), o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o), \\ c_t = f_t \cdot c_{t-1} + i_t \cdot \tanh(W_c x_t + U_c h_{t-1} + b_c), h_t = o_t \cdot \tanh(c_t) \end{gathered} \tag{4}$$

We also tried using a gated recurrent unit (GRU) [8], which functions similarly to LSTM but only uses two gates and does not pass an extra memory state to the next timestep, which makes it more computationally efficient, only requiring the following forward pass calculations:

$$\begin{gathered} z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z), r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r), \\ h_t = (1 - z_t) \cdot h_{t-1} + z_t \cdot \tanh(W_h x_t + U_h h_{t-1} + b_h) \end{gathered} \tag{5}$$

The final technique we implemented is truncated backpropagation through time (TBPTT) [9], which only passes the gradients back through a fixed number of timesteps when training and therefore does not train on the beginning of each sequence. This makes training more efficient but limits the amount of data from which the model is able to learn.

5 Experiments/Results/Discussion

Given the unbalanced classes in our dataset, we used F1 score as our main metric, as it is independent of class imbalance. We report accuracy, precision, recall, and F1 score for all experiments, but ran experiments with the goal of optimizing our F1 score, where the positive label is positive response to treatment.

For our baseline model, we performed multinomial Naive Bayes classification on a dataset of labeled pathology and radiology reports. Initially, we used the small (base) dataset composed of 625 reports with 456 reports (72.96%) labeled as negative response and 169 (27.04%) labeled as positive response. We also performed logistic regression using the same features. For both, we experimented with both unigram and bigram features, with the bigram model outperforming the unigram model slightly, as expected. The results for Naive Bayes and logistic regression are presented below.

Model	Accuracy	Precision	Recall	F1
Naive Bayes Unigram	0.810	0.650	0.722	0.684
Naive Bayes Bigram	0.873	0.778	0.778	0.778
Logistic Regression Unigram	0.857	0.800	0.667	0.727
Logistic Regression Bigram	0.857	0.800	0.667	0.727

We also experimented with various RNN architectures, including using LSTM and GRU units, as described in the methods section. Furthermore, we implemented Truncated Backpropagation Through Time (TBPTT) in order to investigate whether effectively shortening the length of our sequences during training might help the networks improve (due to our very long sequence lengths from the reports, which we were concerned might hamper learning).

Looking at our initial results, we saw the model was biased toward predicting class 0 (negative response to treatment) likely due to the class imbalance. To combat this, we weighted our loss function by the inverse ratio of the classes to balance the loss contributions from both classes and incorporated this into our models. The results of these experiments are below:

Model	Accuracy	Precision	Recall	F1
Simple RNN	0.238	0.238	1.000	0.384
LSTM	0.190	0.194	0.923	0.321
GRU	0.365	0.362	0.967	0.527
TBPTT	0.860	0.750	0.300	0.429

Given the small size of our labeled dataset and the relatively poor performance by our neural network models, we tried to create more data by using other labeling techniques to obtain more labeled data (see Dataset). The labels provided were weaker and we did not have full confidence in the labeling functions, but this greatly increased the size of our training data. With this larger labeled dataset, we ran all of our models again. The results are below.

Model	Accuracy	Precision	Recall	F1
Naive Bayes Unigram	0.791	0.573	0.579	0.576
Naive Bayes Bigram	0.804	0.596	0.589	0.592
Logistic Regression Unigram	0.835	0.742	0.484	0.583
Logistic Regression Bigram	0.840	0.742	0.516	0.609
Simple RNN	0.368	0.231	0.679	0.345
LSTM	0.482	0.542	0.614	0.576
GRU	0.547	0.455	0.212	0.289
TBPTT	0.610	0.271	0.347	0.304

Here are the confusion matrices from the Naive Bayes bigram, logistic regression bigram, and LSTM models.

Table 1: Naive Bayes Confusion Matrix

	Predicted Negative	Predicted Positive
Negative	260	38
Positive	39	56

Table 2: Logistic Regression Confusion Matrix

	Predicted Negative	Predicted Positive
Negative	281	17
Positive	46	49

Table 3: LSTM Confusion Matrix

	Predicted Negative	Predicted Positive
Negative	194	102
Positive	43	53

Overall, our baseline models outperformed the neural network architectures. Given that, we experimented with several hyperparameters to try to improve the performance of the neural network models.

Note that this means that we tried to obtain the best possible performance from our models. While we do not have an unbiased test set, we were simply trying to show the maximum possible performance because even this performance was below our baseline models. Furthermore, we wanted to use as much data as possible for training. We experimented with different learning rates and found the best results with a learning rate of 0.01. We also experimented with different batch sizes and there was no significant difference among batch sizes (we hoped that increasing the batch size might reduce any of the stochastic nature present in training). Changing the hidden state dimension and changing the number of layers had no significant effect.

Looking at the training loss and accuracy, the training loss only decreases modestly and we converge to a set loss and accuracy without decreasing the loss any further, representing a bias problem. We were unable to overtrain on our data sets unless we significantly decreased the size. We checked gradients and weights to check for vanishing and exploding gradient problems and to make sure the weights were changing. This might suggest that our models were not sufficiently complex to separate the classes given the complexity of the data in terms of length of sequences (sequence length in the thousands) and large embedding vectors.

Given that the baseline models, which took into account word counts as features, were able to predict well, it may be that representing the reports as very long sequences with embeddings makes it significantly harder to separate the data. In other words, the models have a hard time learning any useful information from the extremely long sequences versus less complex features such as counts.

There were also several major difficulties we encountered in our task. One major limitation was the lack of reliably labeled training examples. In order to obtain a significant sample size, we were forced to label patients based on their survival time after being diagnosed with cancer, which may not correlate perfectly with their response to NAC. These imperfect labels may have made it difficult for our networks to learn or for the datapoints to not be as cleanly separable. Indeed, the baselines performed worse on the enlarged datasets. Another potential problem was the relatively small amount of data on which our embeddings were trained. It may have been difficult to find meaningful representations when many words were not seen a large number of times.

6 Conclusion/Future Work

We were able to achieve moderate performance in predicting the response of triple-negative breast cancer to neo-adjuvant chemotherapy using logistic regression and Naive Bayes models trained with a bag-of-words approach. However, we had difficulties achieving comparable performance using recurrent neural network models trained on word embedding sequences. In general, our findings suggest that a representation of the semantic information in the reports as a sequence of word embeddings may be too complex to support effective learning with such a small dataset. More structured approaches to representing clinical text incorporating specific domain knowledge could help balance the limited amount of information present in our data.

Given further time and resources, we would also be interested in attempting to produce more usable data using weakly supervised learning techniques. Perhaps by using information such as the number of follow-up visits a patient required or their cause of death, we could generate more labeling methods which could be used to label a much larger set of reports. With a significantly larger dataset, the more complex RNN models would be more likely to find useful patterns in the data and predict a patient's response more effectively. Another promising alternative is developing better methods to reduce the scale/complexity of the inputs. One possible approach we are interested in pursuing is using an initial recurrent model to generate sentence embeddings from sequences of words and then training a classification model on the resulting sentence embeddings, which should form a denser and lower-dimensional representation of each report. In general, we think methods to simplify the representation of the inputs or increase the amount of available data would help significantly with our results.

We would like to thank Dr. Haruka Itakura for her support and guidance.

7 Code

Our code is located in this Github repository: https://github.com/bryankim96/cs229-project

8 Contributions

All three team members worked on writing the code, running training/experiments, developing our methods, and writing the proposal/milestone/report.

9 References

[1] Banerjee I., Bozkurt S., Alkim E., Sagreiya H., Kurian A.W., Rubin D.L. (2019) Automatic inference of BI-RADS final assessment categories from narrative mammography report findings, *Journal of Biomedical Informatics 2019 April*, 92:103137.

[2] Ling A., Kurian A.W., Caswell-Jin J.L., Sledge G.W., Shah N.H., Tamang S. (2019) A semi-supervised machine learning approach to detecting recurrent metastatic breast cancer cases using linked cancer registry and electronic medical record data. *Journal of the American Medical Informatics Association Open 2019* ooz040.

[3] Kowsari K., Meimandi K. J., Heidarysafa M., Mendu S., Barnes L., Brown D., Id L. (2019). Text Classification Algorithms: A Survey. Information (Switzerland). 10. 10.3390/info10040150.

[4] Bojanowski, P., Grave, E., Joulin, A., Mikolov, T. (2017). Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics, 5, 135-146.

[5] Joulin, A., Grave, E., Bojanowski, P., Mikolov, T. (2016). Bag of tricks for efficient text classification. arXiv preprint arXiv:1607.01759.

[6] Rumelhart, D. E., Hinton, G. E., Williams, R. J. (1988). Learning representations by back-propagating errors. Cognitive modeling, 5(3), 1.

[7] Hochreiter, S., Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735-1780.

[8] Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078.

[9] Williams, R. J., Peng, J. (1990). An efficient gradient-based algorithm for on-line training of recurrent network trajectories. Neural computation, 2(4), 490-501.

[10] Maron, M. E. (1961). Automatic indexing: an experimental inquiry. Journal of the ACM (JACM), 8(3), 404-417.

[11] McCullagh, P. (2019). Generalized linear models. Routledge.

[12] Rehurek, R., Sojka, P. (2010). Software framework for topic modelling with large corpora. In In Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks.

[13] Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., ... Lerer, A. (2017). Automatic differentiation in pytorch.

[14] Bernard, M., Thiolliere, R., Saksida, A., Loukatou, G. R., Larsen, E., Johnson, M., ... Cristia, A. (2019). WordSeg: Standardizing unsupervised word form segmentation from text. Behavior research methods, 1-15.

[15] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... Vanderplas, J. (2011). Scikit-learn: Machine learning in Python. Journal of machine learning research, 12(Oct), 2825-2830.

[16] McKinney, W. (2010, June). Data structures for statistical computing in python. In Proceedings of the 9th Python in Science Conference (Vol. 445, pp. 51-56).

[17] Van Der Walt, S., Colbert, S. C., Varoquaux, G. (2011). The NumPy array: a structure for efficient numerical computation. Computing in Science Engineering, 13(2), 22.

[18] Weber, S. C., Seto, T., Olson, C., Kenkare, P., Kurian, A. W., Das, A. K. (2012). Oncoshare: lessons learned from building an integrated multi-institutional database for comparative effectiveness research. In AMIA Annual Symposium Proceedings (Vol. 2012, p. 970). American Medical Informatics Association.

[19] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In Advances in neural information processing systems (pp. 3111-3119).

[20] Foulkes, W. D., Smith, I. E., Reis-Filho, J. S. (2010). Triple-negative breast cancer. New England journal of medicine, 363(20), 1938-1948.