Using natural language processing to predict Spine Assessment Clinic outcomes from textual MRI reports

Zahin Sufiyan   *Shannon Clark*

Abstract

The Spine Assessment Clinic (SAC) in Edmonton, AB determines whether a patient experiencing chronic back or neck pain could benefit from spine surgery. This decision is based on a family doctor's description of the problem, a patient questionnaire, MRI results, and discussion with the patient. The aim of this paper is to support the decision making process of the SAC by developing a machine learning system that could identify those patients who need surgery based on the textual MRI reports. We preprocess the texts before vectorization using bag-of-words, term frequency–inverse document frequency, or word2vec techniques. The generated vectors are used as the features for machine learning algorithms including logistic regression, SVM, an ensemble of the above two, and complement naïve Bayes. Our best performing system used word2vec text vectorization and logistic regression. We report a sensitivity of 0.55 and specificity of 0.58 for this system using five-fold cross validation. Our work provides a foundation for future research on this task. Possible avenues include refinement of the text preprocesing step and further investigation of data augmentation techniques to address the class imbalance of the dataset.

1 Introduction

Chronic pain impacts 18.9% of Canadians over the age of 18. Of those with chronic pain, 37.2% experience it primarily in the lower back, upper back, or neck. [Schopflocher et al., 2011]. In many cases, this pain can be debilitating and prevent those who experience it from attending work and social functions [Henry, 2008]. Surgical intervention may be beneficial for some people facing back or neck pain. After referral from a family doctor, the Spine Assessment Clinic (SAC) in Edmonton, AB determines a patient's need for surgery. A nurse reviews documents sent by the family doctor, including a description of the patient's problem, a patient questionnaire, and MRI results. The nurse may call the patient to discuss their pain over the phone or set up an appointment at the SAC for a spine examination. Based on the gathered information, the nurse and a surgeon assess whether the patient could benefit from spine surgery. Their decision is referred to as the SAC outcome. To expedite this process, it could be valuable to have a computer-based system that predicts the SAC outcome based on the referral information. In this way, patients who require surgery could be identified more quickly and sent directly to a surgeon for consultation, thereby reducing surgical wait times. As such, the objective of this paper is to use natural language processing techniques to develop a system that can predict the SAC outcome from the provided textual MRI report.

2 Related Work

There have been previous attempts to apply natural language processing techniques to radiology reports. Pandya et al. [2019] investigated the effectiveness of searching x-ray and CT reports for certain keywords in order to identify vertebral fractures. They found that their rule-based system could identify 84% of all true vertebral fractures when using the search terms "Loss of Height", "Compression Fracture", and "Crush Fracture". Tan et al. [2018] compared the effectiveness of rule-based and machine learning systems for identifying findings of x-ray and MR reports related to pain in the lower back. The machine learning and rule-based systems had similar specificity (0.97 and 0.95, respectively), but the machine learning system outperformed in terms of sensitivity (0.94

compared to 0.83). However, it is important to note that search patterns from the rule-based system were included as features for the machine-learning model. Our work differs from these in several ways. Firstly, our task is not identification of medical imaging findings but rather identification of a patient's need for surgery. This is arguably a more difficult task as the imaging reports often do not explicitly contain this information, whereas they may directly state the findings. Secondly, we attempt to create a solely machine-learned system without the addition of human-curated rules.

3 Dataset

The SAC dataset contains 763 textual MRI reports labelled with the SAC outcome for the patient. A label of 1 (positive) indicates that surgery was recommended for the patient, while 0 (negative) indicates that surgery was not recommended. Of the 763 samples, 134 are labelled 1. As such, the dataset is class imbalanced. The average length of the MRI reports is 320 ± 165 words, with no significant difference between positively labelled samples (327 ± 143 words) and negatively labelled samples (319 ± 169 words). The MRI reports typically contain the following sections: reason for exam, technique, findings, and impression. However, some reports deviate from this format or use different section titles.

4 Methods

4.1 Text preprocessing

We preprocessed the textual MRI reports to increase their suitability for machine learning methods. First, we converted all uppercase words to lowercase so that the system would not differentiate between the two. This step is useful because both the upper- and lowercase versions of a word convey the same meaning in the context of the MRI reports. Next, we removed words that occur in more than 90% of the training documents. These tend to be frequently used words, such as 'a', 'the', and 'is'. We chose to exclude these words because they contain very little semantic value and therefore are not convincing predictors. We opted for this method rather than removing a predefined list of stopwords since the words that are removed would be tailored to the SAC dataset. We also removed words that occur in less than two of the training documents. This was in an effort to eliminate misspelled words and other irregularities. We did not lemmatize the words because the lemmatizers we experimented with did not prove suitable for our task. In particular, lemmatization introduced errors such as converting "does" → "doe", while leaving "abdomen" and "abdominal" as two separate forms. We also did not remove numeric values because in some contexts they appear to convey important information. For example, numeric values are used for dimensions and in the names of vertebrae. After these preprocessing steps, the textual MRI reports were ready for text vectorization.

4.2 Text vectorization

Text vectorization involves converting a string of text into a vector of real numbers that can be used in machine learning algorithms. We experimented with three different methods for text vectorization: bag-of-words, term frequency–inverse document frequency (TF-IDF), and word2vec. Bag-of-words text vectorization keeps track of word counts in a text but disregards the order of those words. TF-IDF text vectorization quantifies the relevance of a word to a text document based on its statistical significance [Jones, 2004]. This quantification is done by multiplying the frequency of a word in a particular document to the frequency of that word in a collection of documents. Like bag-of-words, TF-IDF vectorization ignores word order. Both bag-of-words and TF-IDF vectorization can take some degree of context into consideration by using n-grams, which are sequences of n words in a row. We experimented with unigrams and bigrams for these vectorization techniques. However, the results for bigrams are ommitted from the report for brevity as they were worse than for unigrams. Lastly, we used word2vec for text vectorization. Word2vec is a technique that allows a word to be represented by a vector of numbers [Mikolov et al., 2013a,b]. The vectors are constructed such that words that are semantically similar and share similar contexts in the text have high cosine similarity. We constructed word vectors from the training texts using the word2vec implementation provided by the gensim library [Rehurek and Sojka, 2011]. We specified the length of the vector as 100 and the window count as 5 to use the five preceding and succeeding words as context. The generated word vectors for each MRI report were averaged element wise to create a single 100-length vector for each report. We additionally experimented with doc2vec [Le and Mikolov, 2014]. For brevity, the results are not reported here as they were comparable to those obtained using word2vec with averaging.

4.3 Machine learning algorithms

We experimented with several machine learning algorithms, such as logistic regression (LR) [McCullagh and Nelder, 1989], support-vector machine (SVM) [Cristianini and Ricci, 2008], an ensemble of the above two, and complement naïve Bayes (CNB) [Rennie et al., 2003]. Our ensemble model was a majority-rules voting classifier. We used scikit-learn [Pedregosa et al., 2011] to implement these models. We used the vectors generated from text vectorization as the features. To address class imbalance, we conducted our experiments with balanced class weights for LR and SVM. The balanced mode adjusts class weights to make them inversely proportional to class frequencies, thereby placing more importance on the positively labelled samples in our task. A complement naïve Bayes classifier was selected over Gaussian and multinomial naïve Bayes classifiers as it is more suited towards imbalanced data. Due to the limited number of data samples, we opted for machine learning models instead of deep learning or transformer models.

4.4 Handling class imbalance

The SAC dataset is relatively small and exhibits class imbalance since only 17.5% of samples belong to the positive class. This imbalance makes it challenging for machine learning algorithms to learn how to accurately classify new samples. We took two approaches to correct the imbalance: augmenting the positive class and undersampling the negative class. We experimented with data augmentation to address the class imbalance problem by generating new positively labelled text samples by making slight modifications to the existing ones. The nlpaug library [Ma, 2019] supports character-level, word-level, and sentence-level text augmentation. We chose to experiment with word-level synonym replacement to generate new positively labelled samples. This technique was chosen over the others because it can generate a textual MRI report without changing its overall meaning by simply replacing a few keywords with their synonyms. We only augmented the training texts during the five-fold cross-validation procedure used to evaluate performance. We used the nlpaug library to generate 250 samples from the positively labelled training texts for each fold by replacing 20% of the text with synonymous words. Synonyms were retrieved using pre-trained word2vec embeddings developed from a Google News dataset (archive) [Mikolov et al., 2013b]. Each training set during cross-validation contains approximately 107 positively labelled samples. This means that some positive samples were used multiple times to generate new samples. Aside from data augmentation, we also experimented with dropping some of the negatively labelled training texts to create a class ratio of 1:2 for positive to negative samples. We also performed experiments using only the original textual MRI reports without any data augmentation or undersampling to evaluate the difference in performance.

4.5 Evaluation metrics

We performed five-fold cross-validation to evaluate each of the models. We computed typical natural language processing metrics including precision, recall, and F1-score. We also computed metrics relevant to the field medical science, such as sensitivity and specificity. The definitions of the metrics in terms of true positives (TP), true negatives (TN), false positives (FP) , and false negatives (FN) are given below. Note that sensitivity and recall have the same definition.

$$\text{Precision} = \frac{TP}{TP+FP} \qquad \text{Recall} = \frac{TP}{TP+FN} \qquad \text{F1} = \frac{2\cdot\text{Precision}\cdot\text{Recall}}{\text{Precision}+\text{Recall}}$$

$$\text{Specificity} = \frac{TN}{TN+FP} \qquad \text{Sensitivity} = \frac{TP}{TP+FN} \qquad \text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}$$

5 Results and Discussion

5.1 Bag-of-words

Tables 1, 2, and 3 report the evaluation metrics for experiments performed with bag-of-words text vectorization. The performance of all models trained on the original dataset was poor, with complement naïve Bayes performing the best. Dropping some of the negative samples in training enhanced the performance of all models. In particular, sensitivity increased dramatically at the expense of some specificity. In this case, the logistic regression and SVM ensemble outperformed the others in terms of F1 score. Data augmentation also helped improve performance, but generally to a lesser extent than dropping negative samples. One exception is complement naïve Bayes, whose performance appears to increase dramatically with the help of data augmentation. One limitation of data augmentation is the possibility of introducing errors into the training data. For example, by changing certain words in a training text during augmentation, the label of the new text may no longer truly be a 1.

Model	**Sensitivity/Recall**	**Specificity**	**Precision**	**F1**	**Accuracy**
LR	0.1558	0.8777	0.2164	0.1805	0.7510
SVM	0.2058	0.8108	0.1894	0.1971	0.7051
LR + SVM	0.1396	0.8839	0.2102	0.1670	0.7536
CNB	0.2370	0.8044	0.1993	0.2148	0.7039

Table 1: Results of five-fold cross-validation for bag-of-words text vectorization using unigrams. The model was trained on the original data.

Model	Sensitivity/Recall	Specificity	Precision	F1	Accuracy
LR	0.3675	0.6982	0.2054	0.2615	0.6396
SVM	0.2935	0.7063	0.1830	0.2242	0.6344
LR + SVM	0.3383	0.7504	0.2243	0.2695	0.6789
CNB	0.3766	0.6880	0.1995	0.2574	0.6330

Table 2: Results of five-fold cross-validation for bag-of-words text vectorization using unigrams. The model was trained on data with some of the negative samples dropped.

Model	Sensitivity/Recall	Specificity	Precision	F1	Accuracy
LR	0.2565	0.8206	0.2325	0.2416	0.7209
SVM	0.2545	0.7934	0.2101	0.2290	0.6986
LR + SVM	0.1864	0.8585	0.2215	0.1998	0.7405
CNB	0.6234	0.4260	0.1890	0.2895	0.4613

Table 3: Results of five-fold cross-validation for bag-of-words text vectorization using unigrams. The model was trained on augmented data.

5.2 TF-IDF

Tables 4, 5, and 6 show the performance for experiments using TF-IDF text vectorization. Results for complement naïve Bayes are excluded as they were extremely poor for this vectorization method. TF-IDF outperformed bag-of-words vectorization in terms of F1 score when trained on the original data. This is likely because TF-IDF places more importance on infrequent words, while bag-of-words weighs all words equally. This is beneficial because infrequent words likely provide more meaningful information for predicting SAC outcomes. Once again, dropping negative samples improved model performance. However, with this vectorization technique, augmenting the training data did not improve results. It is likely that data augmentation introduces some infrequent words into the training texts, which would be given more importance when using TF-IDF vectorization. However, these words are fabricated and do not necessarily occur in the test set. This discrepancy could explain the poorer performance of the data augmentation technique for TF-IDF as compared to bag-of-words text vectorization. Logistic regression was the overall best performing model in this category.

Model	Sensitivity/Recall	Specificity	Precision	F1	Accuracy
LR	0.3747	0.7607	0.2538	0.2963	0.6920
SVM	0.1740	0.8838	0.2400	0.2016	0.7589
LR + SVM	0.1364	0.9077	0.2370	0.1724	0.7720

Table 4: Results of five-fold cross-validation for TF-IDF text vectorization using unigrams. The model was trained on the original data.

Model	Sensitivity/Recall	Specificity	Precision	F1	Accuracy
LR	0.4968	0.6491	0.2306	0.3121	0.6212
SVM	0.3532	0.7203	0.2111	0.2627	0.6553
LR + SVM	0.3279	0.7998	0.2589	0.2872	0.7169

Table 5: Results of five-fold cross-validation for TF-IDF text vectorization. The model was trained on data with some of the negative samples dropped.

Model	Sensitivity/Recall	Specificity	Precision	F1	Accuracy
LR	0.3266	0.7773	0.2350	0.2707	0.6973
SVM	0.2149	0.8759	0.2656	0.2365	0.7602
LR + SVM	0.1416	0.9108	0.2458	0.1784	0.7759

Table 6: Results of five-fold cross-validation for TF-IDF text vectorization. The model was trained on augmented data.

5.3 Word2Vec

Tables 7, 8, and 9 report the experimental results for word2vec text vectorization. Results for complement naïve Bayes were neglected since the generated word vectors contained negative values, making them incompatible with that classifier without further preprocessing. The word2vec technique outperformed both bag-of-words and TF-IDF in terms of F1 score. Once again, a drop in specificity is traded for an improvement in sensitivity. In contrast to the results for the other word vectorization approaches, the performance slightly decreased when dropping negative samples from the training data. This is likely because the model was already able to perform relatively well on the original data, so the excess negative samples seem to have provided some value in this case. The performance is also slightly worse when using the augmented data. It seems that the more sophisticated the text vectorization technique, the less negative impact that class imbalance has on performance. Therefore, the techniques used to combat class imbalance did not prove beneficial.

Model	Sensitivity/Recall	Specificity	Precision	F1	Accuracy
LR	0.5487	0.5790	0.2170	0.3097	0.5728
SVM	0.5825	0.5234	0.2065	0.3040	0.5334
LR + SVM	0.4948	0.6061	0.2116	0.2952	0.5859

Table 7: Results of five-fold cross-validation for word2vec text vectorization. The model was trained on unaugmented data.

Model	Sensitivity/Recall	Specificity	Precision	F1	Accuracy
LR	0.5487	0.5345	0.1987	0.2905	0.5360
SVM	0.5740	0.4979	0.1936	0.2884	0.5098
LR + SVM	0.4877	0.5790	0.1963	0.2784	0.5622

Table 8: Results of five-fold cross-validation for word2vec text vectorization. The model was trained on data with some of the negative samples dropped.

Model	**Sensitivity/Recall**	**Specificity**	**Precision**	**F1**	**Accuracy**
LR	0.4896	0.6072	0.2081	0.2912	0.5859
SVM	0.4571	0.6357	0.2103	0.2878	0.6042
LR + SVM	0.4123	0.6818	0.2155	0.2828	0.6344

Table 9: Results of five-fold cross-validation for word2vec text vectorization. The model was trained on augmented data.

6 Conclusion and Future Work

The goal of this paper was to develop a machine-learning system that could expedite the process of identifying patients of the SAC who need surgery. This paper evaluates various natural language techniques applied to textual MRI reports and functions as a foundation on which further work can be done. Among the three word vectorization techniques investigated, word2vec yielded the highest performance and logistic regression was the best performing machine learning model. Future work on this task could involve collecting more data samples to increase the range of potentially feasible models to include deep learning and transformer based methods. If not possible, further investigation into data augmentation techniques and other methods for improving learning from imbalanced data could prove beneficial. Aside from this, developing a more sophisticated approach to data preprocessing could be valuable as this task deals with raw textual data. In turn, this could produce more statistically significant text vectors that can be used in the machine learning models. Another avenue would be to incorporate domain expert knowledge into the features used for learning to increase the model performance. This could involve an expert curating a list of keywords that are indicative of a patient needing surgery. It would also be possible to combine this work on textual MRI reports with information from the patient questionnaire to increase model performance.

References

Nello Cristianini and Elisa Ricci. *Support Vector Machines*, pages 928–932. Springer US, Boston, MA, 2008.

James L Henry. The Need for Knowledge Translation in Chronic Pain. *Pain Research and Management*, 13:465–476, 2008.

Karen S. Jones. 60 years of the best in information research: A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 60(5):493–502, 2004.

Quoc V. Le and Tomas Mikolov. Distributed representations of sentences and documents, 2014. URL https://arxiv.org/abs/1405.4053.

Edward Ma. Nlp augmentation. https://github.com/makcedward/nlpaug, 2019.

P. McCullagh and J.A. Nelder. *Generalized Linear Models, Second Edition*, pages 107–111. Chapman and Hall/CRC Monographs on Statistics and Applied Probability Series. Chapman & Hall, 1989.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space, 2013a. URL https://arxiv.org/abs/1301.3781.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality, 2013b. URL https://arxiv.org/abs/1310.4546.

Jay Pandya, Kirtan Ganda, Lloyd Ridley, and Markus J. Seibel. Identification of Patients with Osteoporotic Vertebral Fractures via Simple Text Search of Routine Radiology Reports. *Calcified Tissue International*, 105:156–160, 2019.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830, 2011.

Radim Rehurek and Petr Sojka. Gensim–python framework for vector space modelling. *NLP Centre, Faculty of Informatics, Masaryk University, Brno, Czech Republic*, 3(2), 2011.

Jason D. M. Rennie, Lawrence Shih, Jaime Teevan, and David R. Karger. Tackling the Poor Assumptions of Naive Bayes Text Classifiers. *ICML*, 3:616–623, 2003.

Donald Schopflocher, Paul Taenzer, and Roman Jovey. The Prevalence of Chronic Pain in Canada. *Pain Research and Management*, 16:445–450, 2011.

W Katherine Tan, Saeed Hassanpour, Patrick J Heagerty, Sean D Rundell, Pradeep Suri, Hannu T Huhdanpaa, Kathryn James, David S Carrell, Curtis P Langlotz, Nancy L Organ, Eric N Meier, Karen J Sherman, David F Kallmes, Patrick H Luetmer, Brent Griffith, David R Nerenz, and Jeffrey G Jarvik. Comparison of Natural Language Processing Rules-based and Machine-learning Systems to Identify Lumbar Spine Imaging Findings Related to Low Back Pain. *Academic Radiology*, 11:1422–1432, 2018.