Sentiment analysis of adverse vaccine event reports

Gregory J. Lambert
Sandia National Laboratories
glamber@sandia.gov

Abstract

This article examines various approaches for using convolutional neural networks in capturing the implicit or explicit judgments based on documented symptom text from the Vaccine Adverse Event Reporting System (VAERS). Convolutional neural networks with random initialization and pre-trained word vector embeddings are compared to a baseline logistic regression model for accuracy in classifying seriousness of adverse events in VAERS. Empirical results demonstrate that CNNs can be effectively used for text classification tasks such as sentiment analysis on unstructured electronic health records.

1 Introduction

The adoption of electronic health record (EHR) systems in hospitals and clinical facilities has created increasingly larger digital-health databases. Screening of these patient health records presents a large time commitment and has the potential to take away scarce health care resources that could be better allocated to patient care activities. Development of automated patient screening algorithms has the potential to not only save time but assist in delivering better quality care to individual patients. Determining the severity of adverse vaccine events is both a time consuming process for reviewers and an essential step in gaining deeper insights into potential associations with vaccines and adverse event reports.

Two deep learning algorithms were considered for the sentiment analysis, convolutional neural networks (CNNs) and recurrent neural networks (RNNs). RNNs make more intuitive sense for a text classification task than CNNs since text is naturally sequential but we chose to focus our analysis on CNNs for several reasons. One is the exceptional speed increases seen in computing convolutions when coupled with GPUs which is a key ingredient for processing and updating large amounts of electronic health records. Secondly, current research using character-level convolutional networks for text classification [1], where the focus is on purely raw data that does not require knowledge about the syntactic or semantic structure of the language. Additionally, due to the varying sources of symptom text in the observed data (e.g. patients and medical professionals) and varying changes in grammatical style, we decided that convolutions over the word vectors might fair well at text classification of the VAERS data.

2 Background

Literature was examined for the analysis both to gain a better understanding into the current state-of-art in text classification using CNNs but also to explore potential avenues for future work. It has been shown that CNNs are good at extracting raw signals [2] [3] and can be directly applied to distributed or discrete word embeddings [4] [5] without prior knowledge of the syntactic or semantic structures of a language. This approach to using CNNs for text classification has some potentially interesting applied applications. For example, the ability to conduct advanced text classification

methods on streaming data is particularly attractive within the context of biosurveillance NLP applications. There is also recent research using CNNs that treat text as a raw signal at the character-level for language processing [1] [6], this approach removes the need for knowledge of words, in addition to not requiring knowledge about the syntactic or semantic structure of language.

3 Model

We will use a single layer CNN that is based on the paper from Kim, Y. [4] with a few additions for dealing with unbalanced class labels for the sentiment analysis. Let $x_i \in \mathbb{R}^k$ be the i-th word in the sentence of the k-dimensional word vector. Then a sentence can be expressed as a concatenated vector $x_{1:n} = x_1 \oplus x_2 \oplus \ldots \oplus x_n$, where $w \in \mathbb{R}^{hk}$ is a filter applied to all possible windows (concatenated vectors) of h words to produce a new feature. This results in a feature map: $\mathbf{c} = [c_1, c_2, \ldots, c_{n-h+1}] \in \mathbb{R}^{n-h+1}$. Two treatments of word embeddings will be compared, a neural network based representation using word2vec [7] and simple random initializations. Random initializations are implemented using a $unif(-1, 1)$. For word2vec we test both CBOW and skip-gram training algorithms. These word embeddings are then fed into 2-D convolutional layers with multiple filter widths. Additionally, the window stride is set to 1 in the 2-D convolutional layers to capture all possible word combinations given a specified filter width. We test both a ReLU (rectified linear unit) activation function

$$f(x) = max(0, z) \tag{1}$$

where z are the 2-d convolutions plus a bias term (b) and a tanh activation function

$$tanh(z) = \frac{e^{(z)} - e^{(-z)}}{e^{(z)} + e^{(-z)}}. \tag{2}$$

Once the convolutional layers are computed and the activation function is applied, a max-over-time pooling layer is applied on the activations

$$\hat{c} = max\{\boldsymbol{c}\}. \tag{3}$$

This allows us to ideally capture the most important activations for each filter weight. Our final softmax layer is then applied

$$y = softmax(W^{(s)}z + b) \tag{4}$$

where $z = [\hat{c}_1, \ldots, \hat{c}_m]$ (assuming m filters in w) and add class weights are added to the logits to help combat the unbalanced class labels.

3.1 Regularization

Two different types of regularization were used during the model building process to prevent overfitting, dropout and L2 regularization. Dropout is a technique that randomly drops hidden units that can help prevent co-adaptation during training and can be especially useful for small datasets. We apply dropout to the output of our pooling layer, $z = [\hat{c}_1, \ldots, \hat{c}_m]$. Additionally we apply L2 regularization penalties to the weights to our final softmax layer in the network. Figure 1 is a graphical representation of our CNN using TensorFlow [8].

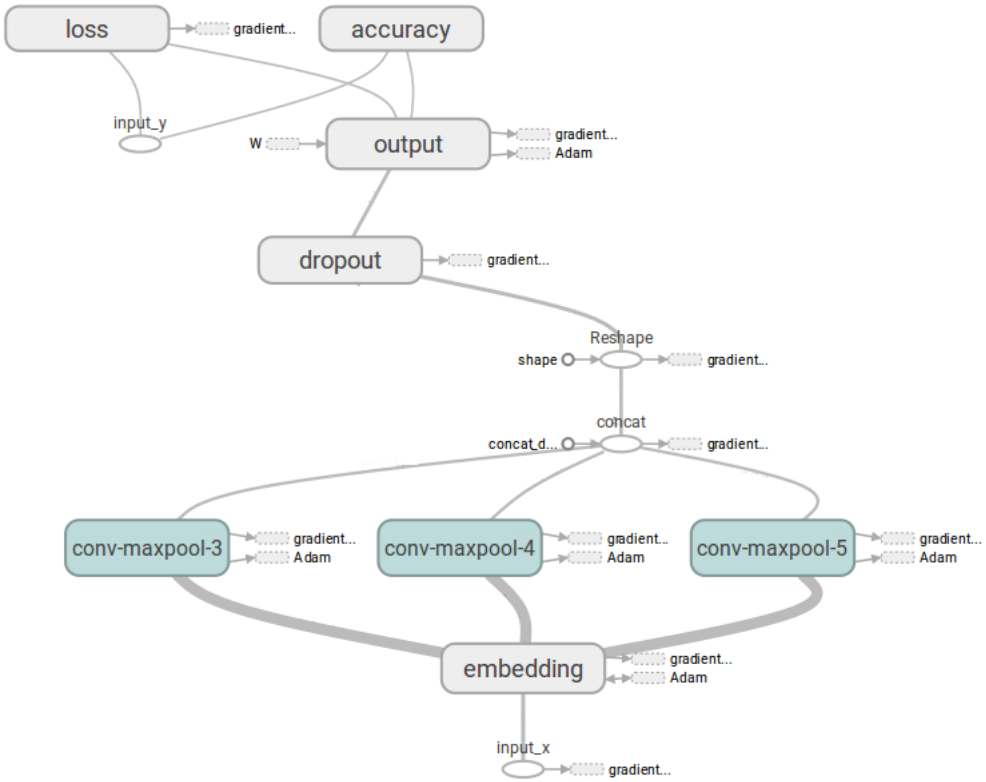

Figure 1: TensorBoard CNN graph

3.2 Model performance

Classification accuracy will serve as our starting point for quantifying model performance. Due to the unbalanced class labels we will focus our performance measurements on the notions of precision and recall. Precision is the number of true positives divided by the number of true positives and false positive and recall is the number of true positives divided by the number of true positives and the number of false negatives. The F_1 score describes the balance between precision and recall and will be used to determine the balance between the precision and recall in the fitted CNN models

$$F_1 = 2\left[\frac{precision \times recall}{precision + recall}\right]. \tag{5}$$

4 Experiment

The Food and Drug Administration (FDA) and the Center for Disease Control and Prevention (CDC) created VAERS to store reports about adverse events that may be associated with vaccines. VAERS receives an average of approximately 36,000 vaccine adverse event reports per year (2009 -2013) making it a suitable dataset for conducting sentiment analysis for classifying severity of adverse events based on symptom text. The symptom text from VAERS is voluntarily submitted by patients, their caregivers, or other individuals. Each symptom text is then coded as an event category: serious or not serious, where serious adverse events include death, life-threatening illness, hospitalization or prolongation of hospitalization, or permanent disability. These event categories

will be considered as our ground truth in this analysis, though it should be noted that certain observations could potentially be mislabeled due to input error.

We select 17,678 VAERS reports with 15,732 labeled as nonserious events and 1,946 labeled as serious for our sentiment analysis. Training, validation sets are split using stratified resampling based on the distribution of the class labels to help alleviate class recall. The text is then preprocessed by converting individual VAERS reports into a sequence of words, while retaining stop words.

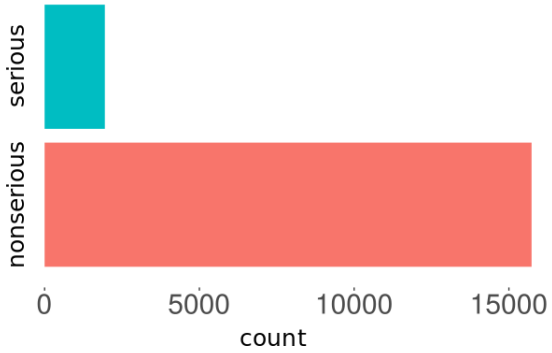

Figure 2: Frequency of unbalanced labels in the VAERS dataset

4.1 Baseline model

A logistic regression classifier using stochastic gradient descent with L2 regularization will be used as our baseline model. To deal with different lengths of symptom text for our baseline, we took a simple average of all the word vectors in a given symptom text. Since our class labels are unbalanced (e.g. 11% serious and 89% nonserious), we will use the same stratified sampling approach employed for the CNN models to get consistent proportions of class labels in our test and training sets. Additionally, word vectors are normalized prior to fitting the logistic regression model, where positive labels represent serious adverse vaccine events. The baseline logistic regression model achieved 0.75614 test accuracy. The ROC curve shown in figure 3 displays the trade off between true positive rate (sensitivity) and the false positive rate (1- specificity) with AUC 0.61.

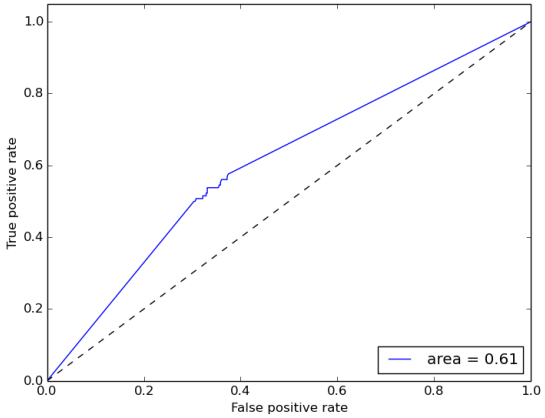

Figure 3: ROC curve for baseline logistic regression

4.2 CNN model

Two CNN model variations will be explored and compared to our baseline model for classifying sentiment in the VAERS data.

- **CNN_R**: model has randomly initialized word embeddings.
- **CNN_W2V**: model has word embeddings that are pre-trained on word2vec vectors.

Both CNN models fared well in terms of precision and recall, easily beating the baseline logistic regression.

Model	Accuracy	Precision	Recall	F_1 score
logistic	0.75614	0.4103	0.1212	0.1871
CNN_R	0.9148	0.9230	**0.9865**	0.9537
CNN_W2V	**0.9170**	**0.9272**	0.9840	**0.9547**

Table 1: Comparison of results for the CNN models and the baseline logistic regression.

4.3 Hyperparameters and training details

For all model runs training was done with stochastic gradient descent using the Adam algorithm [9]. ReLU and tanh were both examined during training, but tanh was abandoned due to difficulties getting the model to converge properly during training, perhaps due to a vanishing gradient problem. The following values where chosen for our final model through trial and error: filter windows (h) of 3, 4, 5, 100 feature maps, dropout rate 0.5, L2 regularization 0.02, batch size 50 with a learning rate for the Adam optimizer 0.001. Finding the correct learning rate, regularization settings, and filter window sizes were the key in getting the model to converge and obtain good model performance. We trained word2vec using both CBOW and Skip-Gram models and found that the CBOW pre-trained word embeddings did slightly better in terms of model performance, possibly due to the relatively small sample size of our training set and the fact that CBOW tends to smooth the entire context of a word as one observation.

5 Conclusion

This article offers an empirical study on the use of CNNs for classifying sentiment of adverse vaccine event reports based on symptom text. While this study was brief, we have shown that there are potential applications for using deep learning NLP methods on unstructured electronic health records. Further analysis needs to be conducted to determine why within the context of this dataset and model setup there was very little difference between the two word embeddings used. Additionally since random initialized embeddings did reasonably well on this text, it would be interesting to further investigate if character-level convolutional networks could be run on large amounts of electronic health records in an almost automated fashion by treating the text as a raw signal, limiting the amount of preprocessing needed on the data. Currently there are large amounts of unstructured data being collected and stored in hospitals and clinical facilities that are underutilized. With the right technology and creativity these data streams have the potential to make a significant impact on public health.

References

[1] Zhang, X., Zhao, J., & LeCun, Y. (2015). Character-level convolutional networks for text classification. In Advances in Neural Information Processing Systems (pp. 649-657).

[2] LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), 2278-2324.

[3] LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., & Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. Neural computation, 1(4), 541-551.

[4] Kim, Y. (2014). Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882.

[5] Santos, C. D., & Zadrozny, B. (2014). Learning character-level representations for part-of-speech tagging. In Proceedings of the 31st International Conference on Machine Learning (ICML-14) (pp. 1818-1826).

[6] Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., & Kuksa, P. (2011). Natural language processing (almost) from scratch. The Journal of Machine Learning Research, 12, 2493-2537.

[7] Le, Q. V., & Mikolov, T. (2014). Distributed representations of sentences and documents. arXiv preprint arXiv:1405.4053.

[8] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., ... & Ghemawat, S. (2016). TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. arXiv preprint arXiv:1603.04467.

[9] Kingma, D., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

[10] dos Santos, C. N., & Gatti, M. (2014). Deep Convolutional Neural Networks for Sentiment Analysis of Short Texts. In COLING (pp. 69-78).

[11] Zhang, Y., & Wallace, B. (2015). A Sensitivity Analysis of (and Practitioners' Guide to) Convolutional Neural Networks for Sentence Classification. arXiv preprint arXiv:1510.03820.