HLP@UPenn at SemEval-2017 Task 4A: A simple, self-optimizing text classification system combining dense and sparse vectors

Abeed Sarker **Graciela Gonzalez**
Institute of Biomedical Informatics
Department of Biostatistics, Epidemiology and Informatics
The Perelman School of Medicine
University of Pennsylvania
{abeed,gragon}@upenn.edu

Abstract

We present a simple supervised text classification system that combines sparse and dense vector representations of words, and the generalized representations of words via clusters. The sparse vectors are generated from word n-gram sequences (1-3). The dense vector representations of words (embeddings) are learned by training a neural network to predict neighboring words in a large unlabeled dataset. To classify a text segment, the different vector representations of it are concatenated, and the classification is performed using Support Vector Machines (SVMs). Our system is particularly intended for use by non-experts of natural language processing and machine learning, and, therefore, the system does not require any manual tuning of parameters or weights. Given a training set, the system automatically generates the training vectors, optimizes the relevant hyper-parameters for the SVM classifier, and trains the classification model. We evaluated this system on the SemEval-2017 English sentiment analysis task. In terms of average F1-Score, our system obtained 8^{th} position out of 39 submissions (F1-Score: 0.632, average recall: 0.637, accuracy: 0.646).

1 Introduction

Text classification is one of the most fundamental natural language processing tasks, and involves the categorization of texts based on their lexical contents. In its simplest form, text classification is binary in nature, such as the categorization of spam *vs.* non-spam email (Youn and McLeod, 2007). Researchers from distinct fields are exposed to a wide range of text classification problems. Even within a specific domain, such as the medical domain, there is a multitude of text classification tasks and problems, such as assessing the qualities of published papers (Kilicoglu et al., 2009; Sarker et al., 2015), outcome polarity classification (Sarker et al., 2011), biomarker classification (Davis et al., 2015), and adverse drug reaction mention detection (Sarker and Gonzalez, 2015) to name a few. Early automated text classification systems were rule-based in nature (*e.g.*, Sarker and Mollá-Aliod (2010)) mostly because of the absence of sufficient annotated data. However, such rule-based systems are generally limited in terms of performance and/or overfit to the target problem. They particularly suffer in terms of performance when exposed to unseen datasets. With the rapid increase in text-based data in all domains (*e.g.*, the largest medical database, Medline,[1] now indexes over 23 million articles), most efficient text classification systems now use machine learning. Such systems utilize annotated data, and automatically extract features from large volumes of annotated text to perform classification. The rules that are used in classification are not hard-coded or predetermined, but are learned from the annotated data automatically. Text classification strategies for various tasks have been thoroughly explored in the literature, and relatively recent progress has seen the use of techniques such as distant supervision (Mintz et al., 2009).

While the automated classification of text is useful for numerous tasks involving natural language, and in a variety of domains, setting up and running text classification systems is very challenging for non-experts. The task generally requires in-depth knowledge of natural language processing (NLP), particularly for preprocessing,

[1] https://www.nlm.nih.gov/bsd/pmresources.html. Accessed: 2/17/2017.

Proceedings of the 11th International Workshop on Semantic Evaluations (SemEval-2017), pages 640–643,
Vancouver, Canada, August 3 - 4, 2017.

feature extraction/generation and analysis, and at least basic knowledge of machine learning so that the classification system can be optimized. For researchers working on interdisciplinary projects (*e.g.*, clinicians working on biomedical informatics projects), the time or opportunity to learn the relevant topics are often unavailable. While state-of-the-art classification techniques now apply sophisticated techniques such as deep neural networks (*e.g.*, Lai et al. (2015)), such techniques are rarely used by non-experts of machine learning in practice. Therefore, there is motivation to design simple text classification systems that are easy to setup and run, and also perform well on real-world text classification problems.

In this paper, we describe a simple text classification system that was initially designed to teach text classification to biomedical informatics students from non-computing backgrounds. The system was modified for application to a 3-class problem (from its initial implementation for binary classification).[2] The system employs a Support Vector Machines (SVMs) (Vapnik, 1995) classifier and some simple features. SVMs are particularly useful for text classification as they are capable of handling large feature vectors. However, to optimize the performance of the SVM classifiers, several hyperparameters have to be tuned (Chang and Lin, 2011). In our system, the *cost* parameter of the classifier and weights for classes (useful for imbalanced datasets) are learned automatically via 5-fold cross validation over the training set. To evaluate the performance of our simple system, we used it for the SemEval-2017 sentiment analysis task in English language (task 4). Our system obtained an average F1-Score of 0.632 (8^{th} out of 39 teams), average recall of 0.637, and accuracy of 0.646. The balance in simplicity and performance of the system suggests that it can be very useful for researchers who are non-experts in the natural language processing and machine learning domains. The system is publicly available, and is open source.[3]

[2]Notes for the workshop for training students about the system are available at: http://diego.asu.edu/Publications/Textclassif_workshop_v2-3.pdf.

[3]Available at: https://bitbucket.org/pennhlp/hlp-upenn_semeval_2017_task4.

2 Methods

The sentiment analysis task for SemEval Task-4A requires the classification of English tweets into one of three sentiment classes: *positive*, *negative* and *neutral*. For the 2017 task, all Twitter sentiment annotations from past years were made available. We downloaded all these annotations and used them for training our system. Specific details about the task can be found in Rosenthal et al. (2017).[4] We used a total of 49,484 tweets from the past annotations of which 19,597 (39.6%) were tagged as positive, 7692 (15.5%) as negative, and 22,195 (44.9%) as neutral. We used an SVM classifier with an RBF kernel for the classification task. In our system, the training set was used to compute suitable weights for each of the classes and an optimal value for the *cost* parameter via 5-fold cross validation. We now briefly discuss our features and parameter/weight optimization approach.

2.1 Feature sets

We used three simple feature sets, which are as follows:

2.1.1 N-grams

To generate sparse vectors, we used traditional word n-grams (n = 1–3). Standard preprocessing steps such as stemming and lowercasing was performed prior to generating the sparse vectors. Stemming was performed using the Porter stemmer (Porter, 1980). We limited the total number of features to 5000.

2.1.2 Word clusters

In an attempt to include more generalized representations of terms, we used word clusters, which have proven to be useful for Twitter-based classification tasks in our past work. The clusters represent groups of terms that are semantically similar. We used a set of publicly available word clusters[5] that were generated by first learning distributed representations of Twitter terms and then clustering the word vectors (Owoputi et al., 2012). The clusters are used in a bag-of-words manner, and feature vectors are generated for these in the same way as the n-gram features.

[4]The task website is: http://alt.qcri.org/semeval2017/task4/. Accessed: 2/20/2017.

[5]Available at: http://www.cs.cmu.edu/~ark/TweetNLP/. Accessed: 2/20/2016

2.1.3 Dense vectors

We obtained dense vector representations of each tweet simply by adding dense representations of individual terms. To obtain dense vector representations of the terms, we used publicly available pretrained vectors[6] (Godin et al., 2015). The vectors were learned from 400 million tweets, and each word is represented using a dense vector of size 400.

2.2 Optimization and classification

A good value for the cost parameter of the SVM classifier was determined via grid search. The grid search included all powers of 2 between 1 and 5. Ideally, identifying the optimal value requires a more thorough search, with an extended search space. We used this small search space to speed up the searching process. To determine the appropriate weight for each class, first each of the three classes were assigned a weight which is equal to the total number of instances in the training set divided by the number of instances for that class in the training set. Thus, for example, the initial weight assigned to the neutral class was $\frac{49484}{22195} = 2.23$. Iterating through possible weights in imbalanced classification tasks can be a tricky problem, and require some expertise in applied machine learning. Without optimization of weights, classification problems involving imbalanced datasets may perform poorly. As mentioned, our system was originally designed to provide simple text classification solutions to non-experts. Therefore, we devised a simple weight optimization strategy. First, the search *interval* is computed as the *variance* of the vector of the initial weight values. For a given class, the possible weight then lies in the range given by Equation 1:

$$range = [max(0.1, \textit{class\_weight} - (2 \times interval)), \textit{class\_weight} + (2 \times interval)] \quad (1)$$

Possible values for weight for the class can then be iterated through within the given range using suitable step sizes. In our work, the initial weight of the positive class (middle class), was kept constant while a range of values were iterated through for the neutral (larger class) and negative (smaller class). Within a given range, we used step sizes of $\frac{interval}{2}$, and chose the weight combination that produced the best results for the cross validation task. A larger search space is likely to result in a better classifier, but also requires longer time for searching.

As described in the previous subsections, during feature generation, a single vector (dense or sparse) is generated for each feature set for each instance. All the three feature vectors for an instance are simply concatenated to form a single vector prior to training. For the system used for this task, each combined vector consisted of a total of 6400 features.

3 Results, Comments and Conclusion

Despite the simplicity of our approach, and limited tuning, it obtained 8^{th} position in terms of average F1-Score out of 39 systems. In addition, the system obtained average recall of 0.637 (11^{th}), and accuracy of 0.646 (8^{th}). Due to time-constraints associated with the submission deadline for the shared task, we only performed 5-fold cross validation, and estimated optimal class weights and values for the cost parameter from a small set of possibilities, as described in the previous section. Therefore, we suspect that the performance of the system could be further improved by searching through a larger set of values. Furthermore, our system was optimized for average F1-Score, resulting in better overall ranking for F1-Score, rather than for average recall.

The key advantage of our system is its simplicity. The parameter optimization is handled automatically, and does not require any manual interpretation. At the same time, the optimization approach is easily interpretable and customizable by non-experts. For example, if better accuracy is required for a specific task, a more thorough search for optimal weights can be performed simply by increasing the number of steps. Such modification does not require deep understanding of SVMs.

Our system is simple and is applicable to any social media text classification task. In the future, we will assess the true performance of the optimized system over the SemEval-2017 test set, via more thorough automated optimization. We will also compare the performance of our simple system with other similar automated systems (*e.g.*, the TPOT system Olson et al. (2016)) in terms of speed and performance.

[6] Available at: http://www.fredericgodin.com/software/. Accessed: 2/20/2016

Acknowledgments

This work was supported by National Institutes of Health (NIH) National Library of Medicine (NLM) grant number NIH NLM 5R01LM011176. The content is solely the responsibility of the authors and does not necessarily represent the official views of the NLM or NIH.

References

Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology* 2(27).

J. Davis, M. Maes, A. Andreazza, J. J. McGrath, S. J. Tye, and M. Berk. 2015. Towards a classification of biomarkers of neuropsychiatric disease: from encompass to compass. *Molecular Psychiatry* 20:152–153.

Fréderic Godin, Baptist Vandersmissen, Wesley De Neve, and Rik Van de Walle. 2015. Multimedia Lab @ ACL W-NUT NER Shared Task: Named Entity Recognition for Twitter Microposts using Distributed Word Representations. In *Proceedings of the ACL 2015 Workshop on Noisy User-generated Text*. pages 146–153.

Halil Kilicoglu, Dina Demner-Fushman, Thomas C. Rindflesch, Nancy L. Wilczynski, and Brian Hynes. 2009. Towards Automatic Recognition of Scientifically Rigorous Clinical Reserc Evidence. *Journal of the American Medical Informatics Association* 16(1):25–31.

Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Recurrent convolutional neural networks for text classification. In *Proceedings of the Conference of the Association for the Advancement of Artificial Intelligence (AAAI)*. pages 2267–2273.

Mike Mintz, Steven Bills, Rion snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*. pages 1003–1011.

Randal S. Olson, Ryan J. Urbanowicz, Peter C. Andrews, Nicole A. Lavender, La Creis Kidd, and Jason H. Moore. 2016. *Applications of Evolutionary Computation: 19th European Conference, EvoApplications 2016, Porto, Portugal, March 30 – April 1, 2016, Proceedings, Part I*, Springer International Publishing, chapter Automating Biomedical Data Science Through Tree-Based Pipeline Optimization, pages 123–137.

Olutobi Owoputi, Brendan O'Connor, Chris Dyer, Kevin Gimpel, and Nathan Schneider. 2012. Part-of-speech tagging for twitter: Word clusters and other advances. Technical report, Carnegie Mellon University. http://www.cs.cmu.edu/ ark/TweetNLP/owoputi+etal.tr12.pdf.

Martin F. Porter. 1980. An algorithm for suffix stripping. *Program* 14(3):130–137.

Sara Rosenthal, Noura Farra, and Preslav Nakov. 2017. SemEval-2017 task 4: Sentiment analysis in Twitter. In *Proceedings of the 11th International Workshop on Semantic Evaluation*. Association for Computational Linguistics, Vancouver, Canada, SemEval '17.

Abeed Sarker and Graciela Gonzalez. 2015. Portable automatic text classification for adverse drug reaction detection via multi-corpus training. *Journal of Biomedical Informatics* 53:196 – 207.

Abeed Sarker, Diego Mollá, and Cécile Paris. 2011. Outcome Polarity Identification of Medical Papers. In *Proceedings of the ALTW 2011*. pages 105–114.

Abeed Sarker, Diego Mollá, and Cécile Paris. 2015. Automatic evidence quality prediction to support evidence-based decision making. *Artificial Intelligence in Medicine* 64(2):89–103.

Abeed Sarker and Diego Mollá-Aliod. 2010. A Rule-based Approach for Automatic Identification of Publication Types of Medical Papers. In *Proceedings of the ADCS Annual Symposium*. Melbourne, Australia, pages 84–88.

Vladimir N. Vapnik. 1995. *The nature of statistical learning theory*. Springer-Verlag New York, Inc., New York, NY, USA. http://portal.acm.org/citation.cfm?id=211359.

Seongwook Youn and Dennis McLeod. 2007. Spam Email Classification using an Adaptive Ontology. *Journal of Software* 2(3):43–55.