IIT BHU at TREC 2019 Incident Streams Track

Akanksha Mishra * and Sukomal Pal

Department of Computer Science and Engineering,
Indian Institute of Technology (BHU), Varanasi - 221005,
India.

Abstract

The paper describes the participation of the IIT BHU at the TREC 2019B Incident Streams track. We submitted two fully automatic runs for categorizing information within tweet into multiple high-level information types and determining the criticality score for each tweet given in the test set.

1 Introduction

The track is to automatically process the social media streams at times of emergency events to categorize information in one or the other high-level information types that can be used by the emergency personnel for immediate action. Track organizers provided participants with the tweet streams covering different event types such as bombing, wildfire, earthquake, flood, typhoon/hurricane, and shooting. They developed a multi-layer ontology of the information types for obtaining generic to specific information. Top-level, high-level, or low-level represents different levels of ontology for information types. For instance, someone is requesting a rescue for himself or herself; hence, top-level intent will be 'Request', high-level intent will be 'SearchandRescue' and low-level intent will be 'SelfRescue'. Tweet streams is also classified into one of the importance scores, such as critical, high, medium, low, and irrelevant. Tweets that require immediate action can be classified into critical or high priority.

*akanksham.rs.cse17@itbhu.ac.in

2 Models

2.1 Multi-label k Nearest Neighbour (IITBHU_run1)

We perform preprocessing of the tweet text by removing punctuation and special symbols. We kept hashtags after removing the hash symbols. We removed non-letters and lowercased the text. After preprocessing, we used the TF-IDF vectorizer for generating feature vectors. We define a dictionary for mapping of the labels with unique integers. We generated a nested list of labels using a dictionary and created a sparse matrix of labels.
We used multi-label k-nearest neighbors [4, 3] to find nearest examples to test class, which uses Bayesian inference to select assigned labels. We considered a range of 1 to 3 neighbors of each input instance and used three different values of the smoothing parameter. We used GridSearchCV[2] to obtain an optimal set of parameter values for a number of neighbors to be considered and smoothing parameters.

2.2 SLEEC(IITBHU_run2)

We perform preprocessing on the text data, as mentioned in run1. We used the TF-IDF vectorizer for feature extraction. We generated a sparse matrix of labels that can be used further by the model. We used Sparse Local Embeddings for Extreme Multi-label Classification[1] for multi-label classification. Specifically, divide the dataset into several clusters, and in each cluster, it detects embedding vectors by capturing non-linear label correlation and preserving the pairwise distance between labels. We used cosine distance to build a kNN graph to learn embeddings. Instead of Singular value projection, we used alternating least square to learn low-rank embedding without L1 regularization. The SLEEC method uses ADMM with l1; however, we have used ElasticNet to optimize the linear regressor.

3 Experiments

For training the models, we used the TREC 2018 train set, TREC 2018 test set, and TREC 2019A test set covering 28 events with a total of around 28800 tweets. Track organizers evaluated the participant system majorly on three metrics - Alerting, Information Feed, and Prioritization. The main focus during metric development was to identify actionable information. In the process of analyzing actionable information, participant system is analyzed for high and critical priority tweets and for performance over six actionable types, i.e., 'Request-GoodsServices', 'Request-SearchandRescue', 'CallToAction-MovePeople', 'Report-EmergingThreats', 'Report-NewSubEvent', and 'Report-ServiceAvailable'. They followed the multi-type classification metric only in the positive class. Table 1 shows results for alerting accumulated alert worth, infofeed information type, and prioritization priority error. Infofeed information type is calculated only for the positive class using a multi-type scheme and macro-averaged across events.

Table 1: Results for submitted runs

	Alerting Accumulated Alert Worth		**Infofeed Infotype**			**Priority RMSE**	
Runs	**High Priority**	**All**	**F1 Actionable**	**F1 All**	**Accuracy**	**Actionable**	**All**
TREC Median	-0.9197	-0.4609	0.0386	0.1055	0.8583	0.1767	0.1028
IITBHU_run1	-0.8313	-0.4367	0.0191	0.1238	0.8139	0.1879	0.1128
IITBHU_run2	-0.9794	-0.4897	0.0275	0.0325	0.7892	0.2174	0.1192

4 Conclusion

We submitted two runs in the incident streams track. We achieve 81.39% accuracy on run1. We can incorporate hand crafted features along with deep learning techniques in future.

References

[1] Kush Bhatia, Himanshu Jain, Purushottam Kar, Manik Varma, and Prateek Jain. Sparse local embeddings for extreme multi-label classification. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, pages 730–738, Cambridge, MA, USA, 2015. MIT Press.

[2] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[3] P. Szymański and T. Kajdanowicz. A scikit-based Python environment for performing multi-label classification. *ArXiv e-prints*, February 2017.

[4] Min-Ling Zhang and Zhi-Hua Zhou. Ml-knn: A lazy learning approach to multi-label learning. *Pattern recognition*, 40(7):2038–2048, 2007.