Applying Machine Learning Classifiers and Data-PipelineFrameworks

The "*GeoSocial*" [1] project is a web-mapping tool that searches for tweets related to posts about different geological hazards/phenomenon (earthquakes, volcanic eruptions, landslides , floods and the aurora) and locates them as markers on a map. The constant flow of information offered by Twitter in the form of short text makes it, in theory, possible to collect real time information about all sorts of events that are being written about on social media and potentially geo-locate them. Over the past year we have been analysing data from Twitter, concentrating primarily on text that contain "aurora borealis" terms (or similar) in the first instance. However, not all the data collected are related to the "aurora borealis" (e.g. "I'm at Aurora Boulevard in Quezon City "). Therefore, it is necessary to classify those tweets in two categories ("Aurora event" or "No-Aurora event") with high accuracy for better information retrieval.

To enhance data retrieval, we have explored two supervised machine learning classification techniques, *Support Vector Machine* (SVM) [2] and *Deep Convolutional Neural Networks* (CNN) [3], for categorizing tweets before they are geo-located. Furthermore, since the data volume collected is very high, we have applied the data-pipeline approach to classify hundred of tweets in parallel by using two data-frameworks, *dispel4py* and *TensorFlow* [5]. Each approach is summarised as follows:

Support Vector Machine. It is widely regarded as one of the best text classification algorithms for two-class classification. We combined SVM with logistic regression (LR) for building the *SVM-LR* classifier. We used *Scikit-learn* library, which is the primary machine-learning library in Python. Previously, we labelled manually 1200 tweets to create our training dataset. Then, we trained the *SVM-LR* classifier, getting an accuracy of 87%. Finally, we applied the *SVM-LR* to a large collection of tweets (15,000 tweets) to classify them in parallel and make some statistics based on their location by using "*dispel4py*" framework.

Deep Convolutional Neural Networks. CNN have shown great promise in the task of text classification in the recent years. We developed a *CNN* classifier based on "Convolutional Neural Networks for Sentence Classification" [7] and inspired by the article "Implementing a CNN for Text Classification in TensorFlow" [8]. The accuracy of a *CNN* classifier is a bit lower than the *SVM-LR*, being 83%. The reason is that neural networks are data-hungry algorithms, so for better accuracy, a larger training set should be used. Nevertheless, we also applied *CNN* classifier to the same collection of tweets as before, but in this case we used *TensorFlow* for classifying them in parallel.

The next steps will be to perform further evaluations, select the one with best accuracy and integrate it in the current system. We also want to extend the algorithms to test their usefulness in obtaining data for the **other hazards within GeoSocial**. It is important to highlight that both data-frameworks provide methods to scale their solutions to large data and resources. In the future, we will investigate the possibility to use distributed resources (e.g. BGS HPC cluster, or Jasmin cloud system [9]) for classifying tweets in real-time.

[1] http://www.bgs.ac.uk/citizenScience/geosocial/home.html
[2] http://www.statsoft.com/Textbook/Support-Vector-Machines
[3] https://adeshpande3.github.io/adeshpande3.github.io/A-Beginner%27s-Guide-To-Understanding-Convolutional-Neural-Networks/
[4] http://journals.sagepub.com/doi/abs/10.1177/1094342016649766
[5] https://www.tensorflow.org/
[6] http://scikit-learn.org/stable/
[7] https://arxiv.org/pdf/1408.5882v2.pdf
[8] http://www.wildml.com/2015/12/implementing-a-cnn-for-text-classification-in-tensorflow/
[9] http://www.jasmin.ac.uk/
[10] http://spark.apache.org/