DICE @ TREC-IS 2018: Combining Knowledge Graphs and Deep Learning to Identify Crisis-Relevant Tweets

Hamada M. Zahera[1], Rricha Jalota[1], and Ricardo Usbeck[1][0000−0002−0191−7211]

Data Science Group, Paderborn University, Germany
{hamada.zahera, rricha.jalota, ricardo.usbeck}@uni-paderborn.de

Abstract. In this paper, we describe our submissions to the TREC Incident Stream (TREC-IS) challenge 2018. We investigated different machine learning approaches to classify crisis-related tweets into different information types. We incorporated knowledge graphs as features into this social media analysis, in addition to bag of words, word embeddings, time data, and event-types. Further, we evaluate state-of-the-art classification models on 31 generated features sets. Our TREC-IS results indicate that a model based on combining knowledge graphs (i.e., Babelfy), word embeddings and textual features outperformes classical machine learning models.

1 Introduction

Today, social media is playing a significant role in disaster management and communication [8]. During emergencies, the availability of real-time information is critical for effective disaster relief and preparedness. Recent studies leverage shared information on social media in mitigating disasters impact and delivering faster responses [3, 5]. For an instance, Sakaki et al. [7] built an earthquake-detection system for Japan using the real-time situational tweets posted by the twitter users. Recently, deep learning models have been successfully applied in this domain. For example, Burel et al. [1] developed a semantic-deep model to classify information categories in crisis-related social media. Their results showed an increased accuracy by incorporating semantic features (e.g., entity representation) compared to statistical and non-semantic features.

However, existing tools to effectively monitor social media are not sufficient due to large volume of information and the need to categorize, cross-refer and verify them. The TREC-IS task aims to classify social media posts, i.e. Twitter tweets, into information types. These information types are modeled as multi-layer (hierarchical) ontologies (Top→ High→ Low) based on existing crisis management ontologies such as MOAC [1]. For instance, a tweet could be categorized in top-level as report information type. Further, tweets at high-level might include information about emerging threats, significant event change or available services. This year, TREC-IS was centered around classifying tweets based on information types only at high-level. More details about TREC datasets including events, tweets and information types are discussed in section 4.2.

In this paper, we followed the intuition that combining Knowledge Graph data with novel statistical information retrieval and novel machine learning techniques can outperform more simplistic models [9]. To solve the TREC-IS task, we extracted different

[1] http://observedchange.com/moac/ns/

features from the tweets and their meta-data including textual features (bag of words), knowledge graph features (bag of concept) from external sources as Babelfy [6], sentiments polarities, date-time and word-embedding features from pretrained models. Then, we combined all possible sets of features (31 features, in total) and investigated various classical machine learning models (e.g., Logistic Regression, Support Vector Machine, etc) and deep learning techniques.

To obtain the best training results, we built a feature pyramid (all vs all) to evaluate and compare the performance of all feature sets against all classifiers. Our experimental results showed that features trained by deep learning model achieved the highest class accuracy compared to the classical machine learning models which in turn achieved higher micro f-measure.

The rest of this paper is organized as follows: we first explore the analysis of TREC training dataset in section 2. Then we describe the details of our approach and official results in sections 3, 4 respectively. In section 5, we conclude the paper with some discussion about future work.

2 Exploratory Data Analysis

TREC-IS provided us with a training data comprising 1335 tweets spread over 23 information types. There were no tweets for two categories - *GoodsServices* and *SearchAndRescue*. Furthermore, the data was unevenly distributed among the 23 given classes with majority of tweets belonging to *Continuing News, Sentiment, Irrelevant, Factoid, Multimedia Share* and *knownAlready* information types. The dataset was largely unbalanced and hence, posed a challenge in efficient classification of the tweets. Figure 1 shows the (partly skewed) distribution of tweets over 23 information types in the training data and 25 information types in the test data. In addition, we present some statistics about datasets in Table 1.

Table 1: TREC-IS Dataset.

#	Train	Test
Total No. of events	6	15
Total No. of tweets	1335	19784
No. of Information Types	23	25
Average No. of tweets/event	267	1318
Average No. tweets/Information Type	53	1736

For this year's challenge, we regarded this task as a timeless classification task and aimed to distinguish patterns in the tweet to information-type mapping. For this purpose, we generated features based on tweet time, event-type, sentiment and underlying semantics of tweets. Although, the training data was insufficient to train sophisticated models using classical machine learning or deep learning techniques, we were interested in uncovering the capabilities of these classification approaches by enriching them

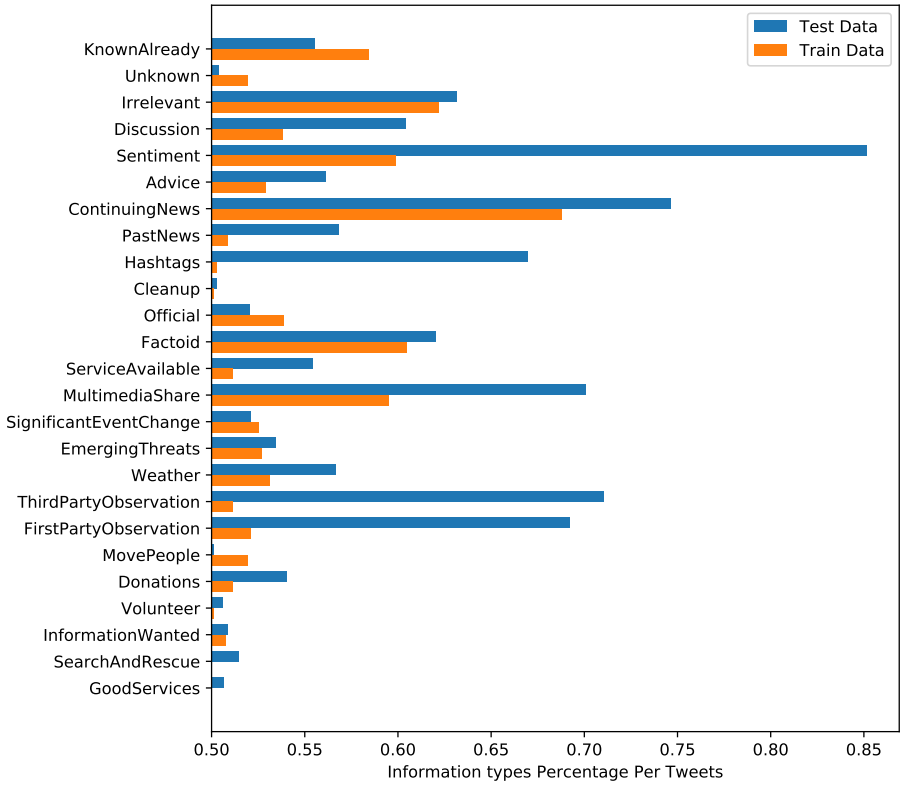


Fig. 1: Percentages of Information Types per Tweets in Training and Test Dataset.

with 31 different features sets generated from the tweets and their meta-data. **Our intention was to perform classification with minimal human effort.**

3 Approach

To account for the two missing classes, we manually augmented the dataset by adding 28 labelled tweets and their meta-data to the training set. We then preprocessed the dataset by normalizing the tweets to contain only the keywords. The normalization process involved lower-casing and lemmatization of text, decryption of emojis to natural language text. Moreover, we removed usernames, numbers, special symbols, urls, extra whitespaces, stopwords and punctuation from tweets. Also, we expanded the contracted words in tweets, for efficient removal of unimportant words (e.g., “how’d’y” to “how do you”, ”that’d” to ”that would”). Table2 shows an example of tweet-preprocessing. We then extracted five features from tweets and their meta-data, and generated all combinations of them. The process of feature-generation has been described in the following section.

3.1 Feature Engineering

In the following, we describe the features used to train the different models:

- **Bag of Words** - We generated a bag-of-words feature from tweets by training scikit-learn's[2] CountVectorizer on the keywords generated from the preprocessing of tweets in addition to the indicator terms that were given in the dataset. This trained model was then used to extract bag-of-words from tweets in the test dataset.
- **Bag of Concepts** - For representing the underlying semantics of each tweet, we used a knowledge graph-based semantic framework, Babelfy[3] which uses entity linking and word sense disambiguation to detect all potential meanings of the recognized text fragments. It selects the best candidate meaning for each fragment and returns its corresponding 'synset' by creating a dense sub-graph from the candidate meanings of the linkable tokens.
  For example, given an input text - "Shots fired, airport evacuated at LAX is what im seeing all over my timeline.", Babelfy returns the following as output: {`'Shots': 'bn:00071206n','fired':'bn:00088225v','airport':'bn:00001676n', 'evacuated':'bn:00087757v','seeing': bn:00082813v','timeline': 'bn:00077308n'`} For each tweet, we used one-hot encoding for the concepts (synsets) that were retrieved from Babelfy. The tweets were sent to Babelfy's API only after expanding word-contractions, removing '#' and 'RT' and substituting emojis with text using the python library Emojipedia[4].
- **Word Embeddings** - Here, we used the pretrained Glove word-embeddings[5] and computed a weighted average of word2vec vectors for each tweet.
- **Sentiment** - By performing sentiment analysis on each tweet, a polarity score $S \in [-1, +1]$ was computed using `TextBlob` text processing library [6]. We did not performed any preprocessing on the tweet text to keep its context for sentiment analysis.
- **Event Type and Date Time** - Since the challenge was aimed at categorizing the tweets based on information type of events, we used the event-type of the tweets as a feature and grouped it together with a date-time attribute from tweets meta-data

Table 2: An Example of Preprocessed Tweets.

Input text	**Preprocessed text**
RT @indigojourney: You dont have 2 ask permission, U see problem & solution...encourage 'leaderhsip' roles #occupy #floodrelief #coflood ht?	rt ask permission u problem amp solution encourage leaderhsip role floodrelief coflood ht
Praying for West Texas♥	pray west texas red_heart

[2]http://scikit-learn.org/
[3]http://babelfy.org/
[4]https://github.com/bcongdon/python-emojipedia
[5]https://nlp.stanford.edu/projects/glove/
[6]https://textblob.readthedocs.io/en/dev/

to create this feature. We first sorted the dataset on event-type and then on time to get an event-wise and time-based representation of the information-types. Finally, we normalized it into range [0, 1].

Feature Pyramid - Finally, from the above-mentioned single features, we generated all possible combination of features (in total, 31 sets of features) and trained our models for each of them.

3.2 Classifiers

For the actual classification task, we employed both classical machine learning as well as deep-learning models. For classical machine learning models, we used the python library scikit-learn[7] and performed hyper-parameter optimization using an automated machine learning library called tpot[8].

On the other hand, for the deep learning approach, we trained a deep neural network model with two dense layers. The first layer has 32 units with ReLU activation and the second layer computed the output probabilities using the standard Softmax function. To estimate the prediction error of the deep learning model, we used the loss function, categorical cross entropy [2] The learning parameters were then optimized using `ADAM` gradient optimizer [4].

Table 3: ML Classifiers with Their Hyper-parameters.

Model name	**Hyper-parameters**
Logistic Regression	solver='newton-cg'
KNeighborsClassifier	n_neighbors=5, weights='distance'
Linear SVM	kernel="linear", C=0.025
RBF SVM	gamma=2, C=1
Linear SVM (squared loss)	C=0.1,dual=True,loss='squared_hinge', penalty='l2',tol=0.0001
DecisionTreeClassifier	max_depth=7,criterion='gini', min_samples_leaf=2,min_samples_split=12
RandomForestClassifier	max_depth=5, n_estimators=10
MLPClassifier	alpha=1
GradientBoostingClassifier	learning_rate=0.01,max_depth=10, max_features=0.35,min_samples_leaf=10, min_samples_split=6,n_estimators=100, subsample=0.75
Deep Model (DNN)	epochs=100,batch_size=100,activation='relu', optimizer='adam',loss='categorical_crossentropy'

[7] http://scikit-learn.org/

[8] http://epistasislab.github.io/tpot/

4 Evaluation

4.1 Evaluating the Feature Pyramids

In order to evaluate the performance of the classical and deep-learning approaches, we estimated our models based on two evaluation metrics: categorical accuracy and micro F1-measure. For classical models, we performed a `Stratified K-fold cross validation`, with $K = 10$. For the deep model, we performed a *train-test* split on the dataset, keeping 90% of the dataset for training and 10% to benchmark the model performance.

We, then investigated and compared the performance of models by trying different settings of the preprocessing module and training dataset (original and augmented version with 28 tweets related to the missing information types). The results showed that the models that were trained on the original dataset had a better classification accuracy than the ones trained on the augmented dataset. Hence, while training the models for final submission we used the original training dataset from TREC and re-ran our evaluation. Figure2 depicts a detailed comparison of all the classifiers against all the feature-sets based on the micro F1-measure.

Based on the classification accuracy, we shortlisted the top four performing models for final TREC submission and generated the submission files using the corresponding feature-sets, namely, {*embedding*},{*embedding, bag-of-words, bag-of-concepts*}, {*bag-of-words, bag-of-concepts*} *and* {*bag-of-words*}.

4.2 Dataset

The TREC-IS dataset contains 1335 tweets for training and 19,784 for testing, which has been curated from different types of events. For each event, the tweets were collected using hashtags and keywords. Human annotators have labeled tweets into the multi-layer ontology of information types. In our experiments, we combined tweets from all the events provided for training, into one dataset.

4.3 Results

The official results are evaluated based on the `micro-averaged` performance metrics of *(Accuracy, Precision, Recall, F1-score)*. Table 4 shows the results of our four submissions (runs) and the median scores. The run, UPB_DICE2, generated from training the deep model with {*embedding, bag-of-words, bag-of-concepts*} feature-set performed better than our other submitted runs, with an F1-score of 0.369 which is somewhat close to the median (0.477) overall participants. In particular, it achieved a higher recall score (0.868) than the median (0.616).

Figure 3a shows the performance of our submissions against all information types. It depicts that we got a higher precision and recall for categories (*ContinuingNews, Sentiment, MultimediaShare, Factoid*) that were better represented in the training dataset than the ones with fewer examples (*GoodServices, SearchAndRescue, CleanUp, Volunteer*, etc). This clearly shows that our models learned the more prominent information types better and could not generalize well over others.

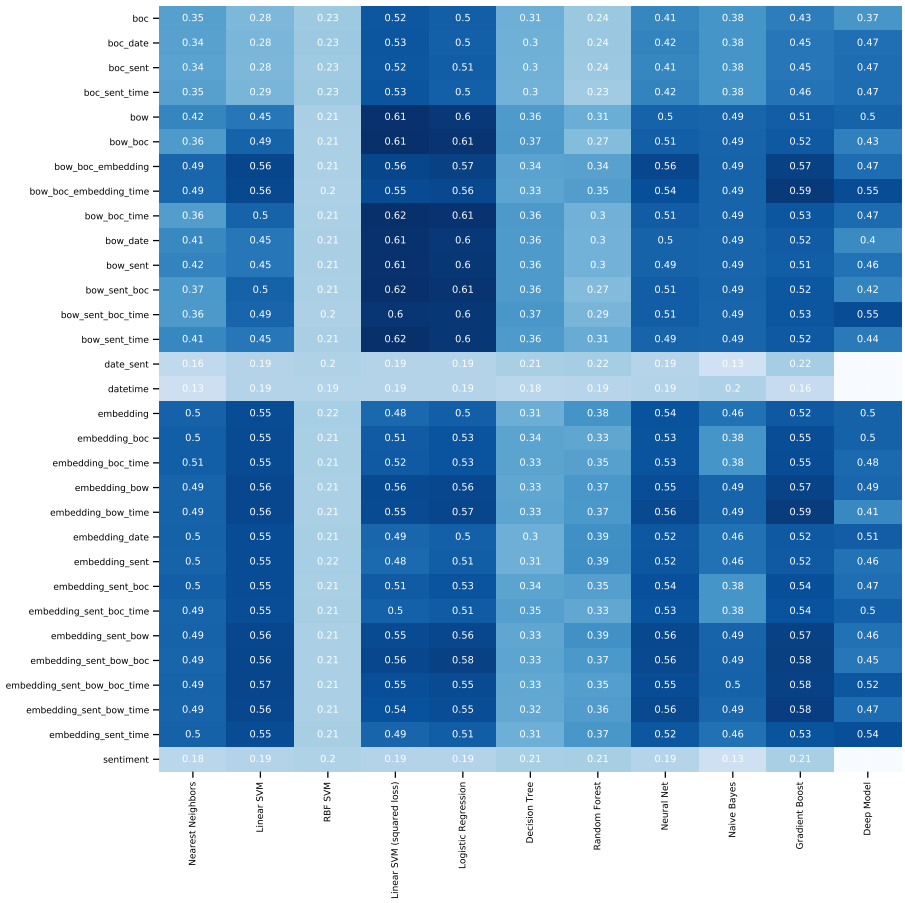


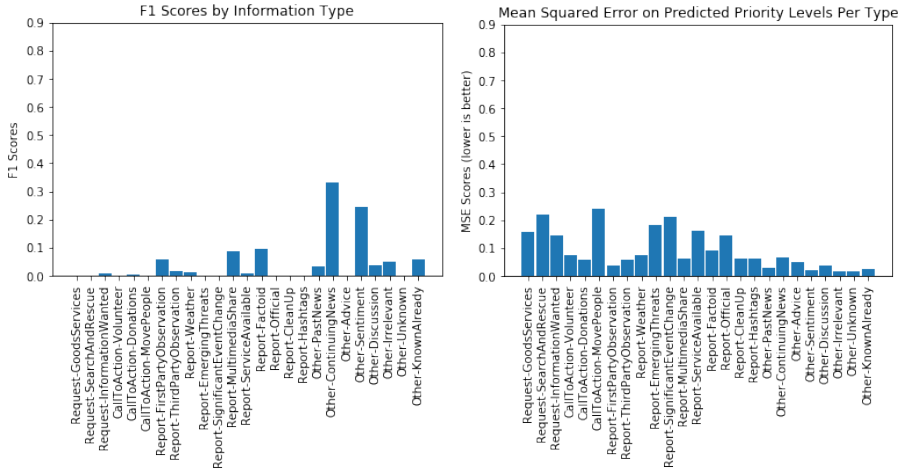
Fig. 2: F1 Scores of classifiers for All Feature-sets.

Table 4: Overall Performance (Micro Average) in All Submissions.

RUN ID	**Precision**	**Recall**	**F1**	**Accuracy**	**Priority Est. Err.**
UPB_DICE1	0.271	0.569	0.367	0.228	0.091
UPB_DICE2 (Best)	0.234	0.868	0.369	0.229	0.092
UPB_DICE3	0.194	0.415	0.265	0.180	0.094
UPB_DICE4	0.215	0.62	0.319	0.203	0.087
Median Scores	0.397	0.616	0.477	0.338	0.093

5 Conclusion and Future Work

In this paper, we described our four submissions to the TREC-IS 2018 track. Our approach extracts textual features from tweets (e.g., bag of words) and incorporates con-



(a) F1 Scores per Information Type for our run UPB_DICE2.

(b) Mean Squared Error on Predicted Priority Levels per Information Type.

ceptual features from knowledge graphs (i.e., Babelfy) to enrich social media analysis. In addition, we also extracted word embedding features from tweets using pretrained Glove embedding model and also took into account the sentiment, time and event-type of the tweets. We then evaluated all the possible combinations of these features against classical machine learning and deep learning models in terms of categorical accuracy and micro-F1 measure. In the future, we plan to re-evaluate our approach with more training data from the TREC-IS challenge as well as on other relevant datasets. Also, we will investigate more deep models with different architectures (e.g. Recurrent Neural Model) and semi-supervised classification approaches.

Acknowledgements

This work has been supported by the BMVI projects LIMBO (project no. 19F2029C), and also by the German Federal Ministry of Education and Research (BMBF) within 'KMU-innovativ: Forschung für die zivile Sicherheit' in particular 'Forschung für die zivile Sicherheit' and the project SOLIDE (no. 13N14456).

Bibliography

[1] Grégoire Burel, Hassan Saif, and Harith Alani. Semantic wide and deep learning for detecting crisis-information categories on social media. In *International Semantic Web Conference*, pages 138–155. Springer, 2017.

[2] Pieter-Tjerk De Boer, Dirk P Kroese, Shie Mannor, and Reuven Y Rubinstein. A tutorial on the cross-entropy method. *Annals of operations research*, 134(1):19–67, 2005.

[3] J Brian Houston, Joshua Hawthorne, Mildred F Perreault, Eun Hae Park, Marlo Goldstein Hode, Michael R Halliwell, Sarah E Turner McGowen, Rachel Davis, Shivani Vaid, Jonathan A McElderry, et al. Social media and disasters: a functional framework for social media use in disaster planning, response, and research. *Disasters*, 39(1):1–22, 2015.

[4] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[5] Peter M Landwehr and Kathleen M Carley. Social media in disaster relief. In *Data mining and knowledge discovery for big data*, pages 225–257. Springer, 2014.

[6] Andrea Moro, Alessandro Raganato, and Roberto Navigli. Entity linking meets word sense disambiguation: a unified approach. *TACL*, 2:231–244, 2014.

[7] Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. Earthquake shakes twitter users: real-time event detection by social sensors. In *Proceedings of the 19th international conference on World wide web*, pages 851–860. ACM, 2010.

[8] Tomer Simon, Avishay Goldberg, and Bruria Adini. Socializing in emergencies—a review of the use of social media in emergency situations. *International Journal of Information Management*, 35(5):609–619, 2015.

[9] Ricardo Usbeck. Combining linked data and statistical information retrieval - next generation information systems. In *The Semantic Web: Trends and Challenges - 11th International Conference, ESWC 2014, Anissaras, Crete, Greece, May 25-29, 2014. Proceedings*, pages 845–854, 2014.