Use of Bayesian approaches for Crisis Management using Tweets

Haritha Guttikonda
University of Virginia
hg5mn@virginia.edu

Sana Syed
University of Virginia
ss8xj@virginia.edu

Suchetha Sharma
University of Virginia
ss4jg@virginia.edu

***Abstract*—Large volumes of data are generated through social media tweets during and after any disaster. The analysis and classification of data obtained at the time of the disaster is essential for conveying information to appropriate rescue personnel. In this project, a tweet classification and inference model using Bayesian methods is proposed. Classification by type will be a way for us to quantify and act on uncertainty. Our goal is to use different thresholds for action for different information types and to understand what affected populations, response agencies and other stakeholders can expect from these data in wild fire natural disaster crisis disaster situations.**

***Index Terms*—tweets, disasters, Bayesian methods, Nested Naive Bayes Classifier, Supervised Deep Belief Network Classification, Bayesian Belief Networks**

I. Introduction/ Problem description

Tweets sent during crisis situations may contain information that could contribute to situational awareness [1]. The goal of this project is to analyze tweets using: 1) Nested Naïve Bayes Classifier, 2) Supervised Deep Belief Network Classification and 3) Bayesian Belief Networks. Classification by type would allow us to quantify and act on uncertainty whereas Belief Networks would allow for development of intuition for the data. We used an open data set from CrisisLexT26, which contained tweets from 26 crises, labeled by informativeness, information type and information source. We selected the 2012 Colorado wildfires data set and 2013 Australian Bushfire data set which was a comma-separated values (.csv) file containing tweet-ids, along with the tweet text and labels for the all the tweets. These are labeled by crowd sourced workers according to informativeness (informative or not informative), information types (e.g. caution and advice, infrastructure damage, etc.), and information sources (governments, NGOs, etc.). Knowing that some information types will have higher threshold for classification than others based off context e.g. even if the probability of a tweet being classified as 'affected individuals' / 'caution and advice' is less (say 0.4) since it would be important to act on a human life we would still consider this tweet as actionable.

II. Bayesian methods used and Mathematical linkage between the problem and the methods

A. Nested Naive Bayes Classifier

The first classifier we explore is a Naïve Bayes classifier that chooses a hypothesis with the maximum probability, i.e., provides the maximum a posteriori (MAP) estimate. In our case, we want to first categorize if a tweet is informative and then determine the information type of the tweet. Therefore, we use two Naïve Bayes Classifiers in a nested fashion, where one determines if a tweet is informative and the other determines the information type, by computing two separate MAP estimates. These classifiers are "naïve" because they assume an *i.i.d* (independent and identically distributed) *bag-of-words* representation for the tweets [5]. More specifically, both classifiers maximize the posterior probabilities based on how many times each word appears in each tweet and how many times it appears in each category (both in terms of informativeness and information type).

Our hypothesis is that, the tweet belongs to the informativeness (I) class "related and informative", and that it fits into information type (T). The "evidence" would be the words (W) occurring in the tweet, i.e., we are interested in $P(I|W)$ and $P(T|I,W)$. Since we assume a *bag-of-words* representation, we break down W into a word vector $W = \{W_1, W_2, ...W_n\}$, and compute the probabilities $P(I|W_1, W_2, ...W_n)$ and $P(T|I, W_1, W_2, ...W_n)$. To obtain the word vector, we use the *CountVectorizer()* utility from the *sklearn* package. This also lets us strip out nonwords, numbers, articles, and other things that are not useful for predicting informativeness (I) and information type (T) from our counts (W). For example, articles (*a*, *an*, and *the*) can be safely ignored because they don't affect the informativeness or the information type of the tweets.

To train both classifiers, we use a default 75-25 training/test data split. Our classifiers are Multinomial Naïve Bayes as they are suitable for classification with discrete features (e.g., word counts for text classification). We also use Leave One Out Cross Validation (LOOCV) to avoid over-fitting our models to the training data. In the first stage of our nested classifier, we predict the informativeness $P(I|W)$. If we predict that a tweet is "related and informative", in the next stage, we predict the information type $P(T|I,W)$.

B. Supervised Deep Belief Network Classifier

This method takes a probabilistic approach on neural networks and is sometimes referred to as *Stochastic Neural Network* with multiple layers of Restricted Boltzmann Machine. Fig -1 shows an Infinite Belief Network with alternative Hidden and Visible layers stacked over each other. The downward arrows represent the generative model [4]. We start at the

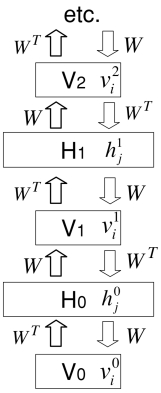

Fig. 1. Infinite belief network with weights [4]

deepest hidden layer with random configuration. Binary state of each random variable is a Bernoulli Distribution determined by its active parents in the layer above which is similar to any other directed acyclic belief network. But the advantage of using a deep belief network is that we can directly sample from the true posterior of the hidden layer distribution onto the visible layer. Then we can use the transpose of the weight matrix to infer factorized distribution of the next hidden layer.

Since we can sample from true posterior, we can get derivatives of the log probability of the data. Derivative for a generative weight, w^{00}, from a unit j in layer h^0 to unit i in layer v^0 is:

$$\frac{\partial logp(v^0)}{\partial w_{ij}^{00}} = h_j^0(v_i^0 - \widetilde{v_i^0})$$

$\widetilde{v_i^0}$ is the probability that unit i would be turned on if the visible vector was stochastically reconstructed from the sampled hidden states.

This is the maximum likelihood learning rule for a single data vector in a logistic belief network. The full derivative for a generative weight is obtained by summing the derivatives of the generative weights across all pairs of layers:

$$\frac{\partial logp(v^0)}{\partial w_{ij}^{00}} = h_j^0(v_i^0 - \widetilde{v_i^1}) + v_i^1(h_j^0 - \widetilde{h_j^1}) + h_j^0(v_j^1 - \widetilde{v_i^2}) + ...$$

To get the maximum likelihood using Restricted Boltzmann Machine, we use the difference between two correlations. For each weight, w_{ij}, where i is used for visible layer and j is used for hidden layer, we measure the correlation $v_i^0 h_j^0$ when a data vector is clamped on the visible units and the hidden states are sampled from their conditional distribution, which is factorial. Then, using alternating Gibbs sampling on the hidden layers till it reaches a stationary distribution and measure the correlation $v_i^\infty h_j^\infty$ [4]. All of the pairwise products except the first and last cancel, leaving the gradient of the log probability as:

$$\frac{\partial logp(v^0)}{\partial w_{ij}^{00}} = v_i^0 h_i^0 - v_i^\infty h_j^\infty$$

Maximizing the log probability of the data is exactly the same as minimizing the Kullback-Leibler divergence, $KL(P^0||P_\theta^\infty)$, where P^0 is the distribution of the data, and P_θ^∞ is the equilibrium distribution defined by the model.

We implemented this method using a python package dbn.tensorflow where we can give number of hidden layers and number of units in each layer.

C. Bayesian Belief Network

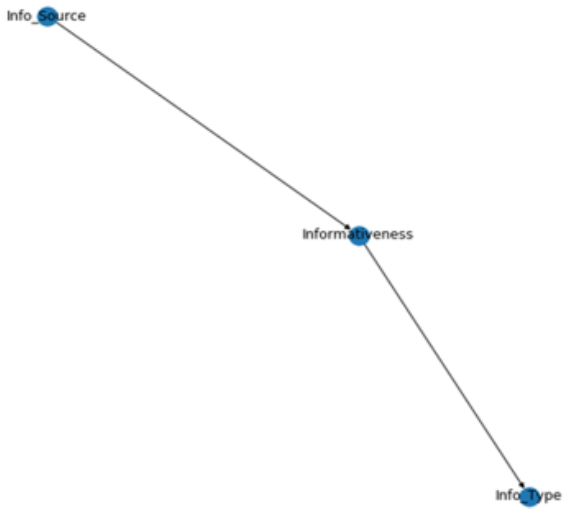

Fig. 2. Bayesian Belief Network Representation for Tweet data

Bayesian Belief (BB) Network or Bayesian Network or Belief Network is a Probabilistic Graphical Model (PGM) that represents conditional dependencies between random variables through a Directed Acyclic Graph (DAG) [2]. Belief networks are used by experts to encode selected aspects of their knowledge and beliefs about a domain. Once constructed, the network induces a probability distribution over its variables. The main objective of a Bayesian network is to try to understand the structure of causality relations and to encode uncertain knowledge in complex domains. These networks rely on inference algorithms to compute beliefs of alternative hypotheses in the context of observed evidence. In our tweet data we have three main variables of interest: A- informativeness (informative or not informative), B- information types (e.g. caution and advice, infrastructure damage, etc.), and C- information sources. Given these, we construct our Belief Network and when a test tweet is evaluated, we can infer

which of the these three labels are most likely to be generated by providing probabilities for each label.

$$P(X_1, ..., X_n) = \prod_{i=1}^{N} P(X_i|Parents(X_i))$$

For our model our random variables were:
A - Information Source (Media, Govt, NGOs, Outsiders, Business, Eyewitness)
B - Informativeness (Informative, Not Informative)
C - Information type (Affected Individuals, Caution, Donation, Infrastructure, Sympathy, Others)

From the Bayesian Belief Network that we assumed in Fig. 2, the joint distribution can be written as:

$$P(A, B, C) = P(A) * P(B|A) * P(C|B)$$

This method is implemented in pybbn python library which can compute belief networks with Discrete Random Variables [3]. We added all the nodes and their corresponding transition probabilities and arcs to specify conditional relationship between the Random Variables.

III. Results

A. Nested Naive Bayes Classifier and Supervised Deep Belief Network Classifier

1) Naïve Bayes Classifier: The following confusion matrix characterizes the performance of our classifier.

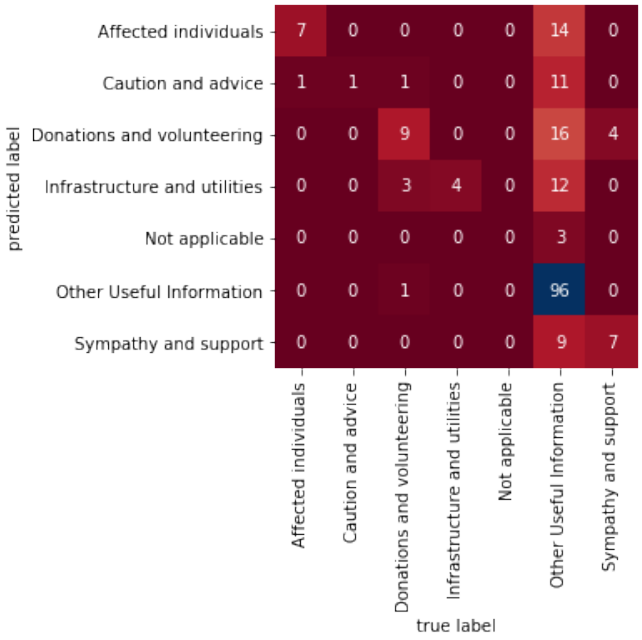

Fig. 3. Confusion Matrix for the Naïve Bayes Classifier given that the tweet is "related and informative".

In summary, the accuracy of the nested Naïve Bayes Classifier given that the tweet is "related and informative" is 62.3%.

2) Deep Belief Network: Accuracy of the Deep neural network depends largely number of hidden layers used and number of units in the hidden hidden layers [6]. This implementation gives best results when the input data or the features in input data have binary state. But the input in our problem is a sparse matrix with integer values which we got from the count vectorizer. We got best results when we use 2 hidden layers with 50 units in each. The accuracy obtained in that configuration is 56%. We can also get dummy variables from the sparse matrix to feed into the model, but that takes lot of memory and needs more time for training.

B. Bayesian Belief Network

Bayesian Belief Networks does not give a prediction as such but provide inference on the evidence. We checked the causality of the inference we got from the evidence that the Information Source is Outsiders with probability 1. Logically, if a person outside of the crisis tweet about it, it is most likely to be not informative and the type of information is more likely to be Sympathy and Support than about Infrastructure or Affected Individuals, which is what we were able to infer from the Belief Network as well.

```
1|Informativeness|Related_Informative,Related_Not_Informative
1=Related_Informative|0.42037
1=Related_Not_Informative|0.57963
---------------------->
0|Info_Source|Business,Eyewitness,Government,Media,NGOs,Outsiders
0=Business|0.00000
0=Eyewitness|0.00000
0=Government|0.00000
0=Media|0.00000
0=NGOs|0.00000
0=Outsiders|1.00000
---------------------->
2|Info_Type|Affected Individuals,Caution,Donations,Infra,Other,Sympathy
2=Affected Individuals|0.12823
2=Caution|0.02931
2=Donations|0.07392
2=Infra|0.04683
2=Other|0.37213
2=Sympathy|0.34957
---------------------->
```

Fig. 4. Inference from Evidence (Information Source = Outsiders with probability 1)

IV. Conclusions

Our wildfire disaster data analysis reveals that tracking tweets during crisis can help us understand the situation better for providing rescue personnel assistance as needed. Our proposed methods include classification as the model output from the Naïve Bayes and Deep Belief Network classification models along with the use of Bayesian Belief Networks which give us the probability of the tweet being in different states. Our purpose in including different modalities was to allow us to get probabilities to better develop intuition about the tweet data. Follow up steps for the Bayesian Belief Networks would be to include use of tweet text data as an additional predictor via sentiment analysis, by categorizing sentiment score say- high, low, medium. Further, for all the models an additional variable to consider would be time-stamp data to check if the tweet is made before, after, or during the disaster. In conclusion, automated classification and belief network systems can allow improved understanding of crisis management and help prioritize relief resources.

REFERENCES

[1] A. Olteanu, S. Vieweg, C. Castillo. "What to Expect When the Unexpected Happens: Social Media Communications Across Crises, " in Proceedings of the ACM 2015 Conference on Computer Supported Cooperative Work and Social Computing (CSCW), 2015.

[2] Atakan Guney. "Introduction to Bayesian Belief Networks" in Towards Data Science, Nov 2019.

[3] C. Huang and A. Darwiche, "Inference in Belief Networks: A Procedural Guide," in International Journal of Approximate Reasoning, vol. 15, pp. 225–263, 1999.

[4] G. E. Hinton, S. Osindero, Y. W. Teh. "A fast learning algorithm for deep belief nets, " in Neural Comput, 2006;18(7):1527–1554.

[5] K. Epley. "Naive Bayes Document Classification in Python." in Towards Data Science, Jun 2019.

[6] Himanshu Singh. "Deep Belief Networks — An Introduction" in Towards Data Science, July 2018.