Bayesian Combination of Crowd-Based Tweet Sentiment Analysis Judgments

Matteo Venanzi
University of Southampton
Southampton, UK
mv1g10@ecs.soton.ac.uk

John Guiver
Microsoft Research
Cambridge, UK
joguiver@microsoft.com

Gabriella Kazai
Microsoft Research
Cambridge, UK
a-gabkaz@microsoft.com

Pushmeet Kohli
Microsoft Research
Cambridge, UK
pkohli@microsoft.com

Abstract

In this paper we describe the probabilistic model that we used in the CrowdScale – Shared Task Challenge 2013 for processing the CrowdFlower dataset, which consists of a collection of crowdsourced text sentiment judgments. Specifically, the dataset includes 569,786 sentiment judgments for 98,979 tweets, discussing the weather, collected from 1,960 judges. The challenge is to compute the most reliable estimate of the true sentiment of each tweet from the judgment set while taking into account possible noise and biases of the judges and other properties of the text contained in the tweets. To address this challenge, we developed a Bayesian model, which is able to infer the true sentiment of the tweets by combining signals from both the crowd labels and words in the tweets. The model represents the reliability of each judge using a confusion matrix model and the likelihood of each dictionary word belonging to a certain sentiment class using a mixture of bag of words models. Both these models are combined together to learn the latent true tweet sentiments. We discuss our scalable model implementation using the Infer.NET framework, and our preliminary results which show that our model performs better than the majority voting baseline.

Introduction

In one task of the CrowdScale – Shared Task challenge 2013[1], participants were asked to design methods, which are able to infer the most reliable classification answers based on the ratings of multiple judges about the sentiment of a large set of weather related tweets. The dataset includes 569,786 judgments for 98,979 tweets collected from 1,960 judges. Each rating can belong to five possible classes: 0 = Negative, 1 = Neutral, 2 = Positive, 3 = Tweet not related to weather conditions, 4 = Can't tell. Some example tweets , posted by users at different geographical locations, are shown in Figure 1. The challenging task is then to derive the true sentiment of each tweet based on the multiple subjective judgments available.

[1]`sites.google.com/site/crowdscale2013/shared-task`

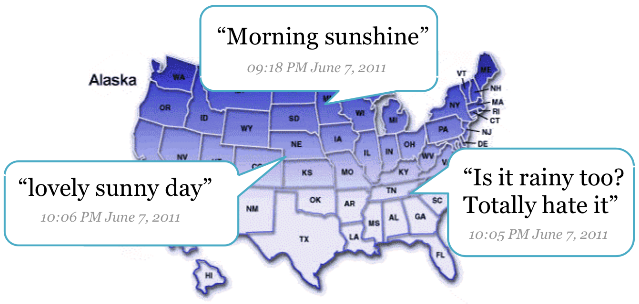


Figure 1: Example picture of three tweets randomly sampled from the CrowdFlower dataset. The tweets are posted by users from different locations and express different sentiment about the weather. The challenge is to infer the true sentiment of each tweet from the available set of judgments.

To solve this task, there are various technological and uncertainty challenges which we must be addressed for the efficient aggregation of text sentiment judgments. First, there is subjectivity in the judgments due to the ambiguous wordings of the tweet which do not allow for an easy distinction between the five rating classes. Second, the judges can be inaccurate due to their individual skills as raters, or biased towards a particular class. Third, the method must be able to handle web-scale amount of tweets using feasible computing resources. So far, prior work on aggregating crowd labels focused on methods that are able to jointly learn the latent true classification answer of each object and the individual accuracy of each judge. However, these methods are typically designed for general classification tasks such as image labeling or text annotation, and try to infer these latent traits by only considering the set of observed labels. Therefore, when applied to our problem, these methods are unable to exploit information contained in the text of the tweets, which is crucial to achieve more reliable sentiment estimation in text analysis.

In contrast, we design a new aggregation model, which is able to combine signals from both the crowd labels and the text properties of the tweets using a principled Bayesian learning framework. More specifically, our model represents each worker's reliability as a confusion matrix (Ghahramani and Kim 2003), representing the probability of a workers providing a specific judgment for each possible true sentiment class. Furthermore, our model includes a bag of words model to represent the probability of each dictionary word to be present in a specific sentiment class. Both these latent features are related together by the latent true sentiment of the tweet. We then use Bayesian inference to learn the posterior distributions of the worker reliabilities, the true tweet sentiments, and the word probabilities from the observed judgment data. Furthermore, we design an efficient implementation of our model using the inference decomposition framework provided by Infer.NET, which allows us to perform batch inference over the 569,786 judgments of the CrowdFlower dataset in approximately 50 minutes.

In what follows, we discuss our model in more detail, describing its probabilistic inference and scalable implementation. Then, we present our preliminary results for the CrowdFlower dataset. Finally, we conclude with a discussion of our results and future improvements for our model.

Our Bayesian Combination Model

We start by introducing our notation. There is a crowd of K judges (or workers) expressing their judgment about the sentiment of N tweets over a range of C possible sentiment classes. Each tweet i has an unknown true sentiment $t_i \in C$. The judgment of worker k for tweet i is denoted as $c_i^{(k)}$. Also, we assume that each word $w_d^{(i)}$ which appears in the tweet i is part of a dictionary of size D. For notational simplicity, we assume a dense set of judgments in which each judge rates all the N tweets. However, the implementation of our model can support sparsity in the dataset (which also the case for the CrowdFlower dataset).

The factor graph of our Bayesian Combination Model, referred to as *BCCWords*, is shown in Figure 2. The model assumes that each worker draws judgments for tweets of class t_i from a categorical distribution with parameters $\pi_c^{(k)}$:

$$\boldsymbol{c}_i^{(k)} | \boldsymbol{\pi}_c^{(k)}, \boldsymbol{t}_i \sim \text{Cat}(\boldsymbol{\pi}_c^{(t_i)})$$

where, $\boldsymbol{\pi}_c^{(k)}$ is the accuracy vector of worker k for tweets of class c. That is, $\boldsymbol{\pi}_c^{(k)}$ is the confusion matrix of worker k for each possible $c \in C$. Then, we assume that the probability or a word $\boldsymbol{w}_d^{(i)}$ of tweet i follows a Categorical distribution with parameters $\boldsymbol{w}_d^{(c)}$:

$$\boldsymbol{w}_d^{(i)} | \boldsymbol{w}_d^{(c)}, t_i \sim \text{Cat}(\boldsymbol{w}_d^{(t_i)})$$

where $\boldsymbol{w}_d^{(c)}$ is the probability of word $\boldsymbol{w}_d$ to appear in a tweet of class c. In particular, this probabilistic representations of text in tweets corresponds to a mixture of bag of words model (Salton and McGill 1983), where each mixture component is a bag of words model associated with one particular object class. In particular, the dictionary used in our method was taken as the union of the first 300 most frequent words in the tweets, after removing a list of common stop words.

To infer the latent variables of interest under our model, we use conjugate Dirichlet priors for the parameters $\boldsymbol{w}_d^{(c)}$

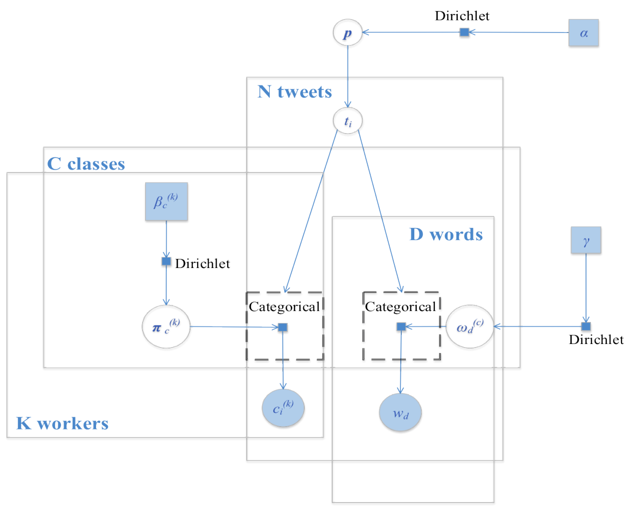


Figure 2: Factor graph of BCCWords. The four plates included in the graphs describe (i) the set of K workers, (ii) the N tweets, (iii)the C possible sentiment classes and (iiii) the D words contained in the global dictionary from the union of the words of each tweet.

and $\boldsymbol{\pi}_c^{(k)}$:

$$\boldsymbol{\pi}_c^{(k)} | \boldsymbol{\beta}_c^k \sim \text{Dir}(\boldsymbol{\beta}_{\boldsymbol{c}}^{\boldsymbol{k}})$$
$$\boldsymbol{w}_d^{(c)} | \boldsymbol{\gamma} \sim \text{Dir}(\boldsymbol{\gamma})$$

where $\boldsymbol{\beta}_{\boldsymbol{c}}^{\boldsymbol{k}}, \forall c$ is the per-judge confusion matrix hyperparameter, which we set to having a slightly higher diagonal value (i,e. meaning that judges are better than random a priori). Also, $\boldsymbol{\gamma}$ is the words Dirichlet vector hyperparameter, which we initialize to a uniform vector (i.e. meaning that a priori words have a uniform probability in each class).

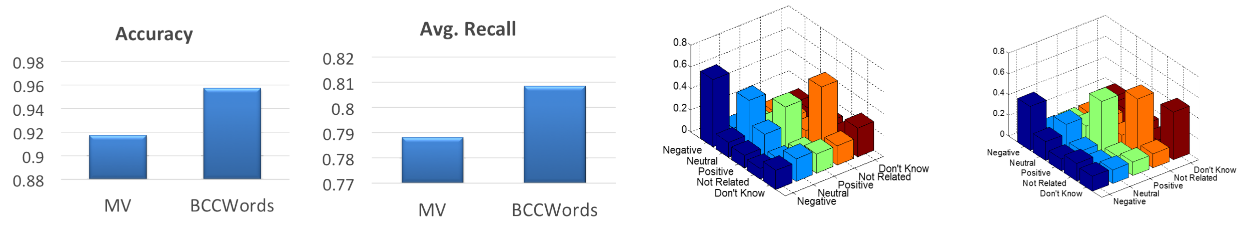
In this setting, we can compute approximate inference of the model parameters using standard message passing algorithm. In particular, our implementation utilizes the Expectation Propagation (EP) algorithm (Minka 2001), provided by the Infer.NET probabilistic programming framework (Minka et al. 2013).

Scalable Infer.NET Implementation

To train our model on large-scale datasets (i.e., in the order of hundreds of thousands of labels), we developed a scalable implementation of BCCWords based on the shared variable inference decomposition method using Infer.NET.[2] This method is based on splitting the model into a number of sub-models which share some variables. The advantage of this implementation is that we can run cheaper and parallel inference on the different sub-models and merge the inference results by extracting the shared variable output message of each sub-model.

Specifically, our shared variable implementation of BCCWords includes two sub-models: an object model and a set of worker models. The object model manages the inference

[2]For an example, see `bit.ly/1ajlMWN`



(a) Accuracy (b) Average recall (c) Pessimist worker (d) Extremist worker

Figure 3: Accuracy and inference results of BCCWords on the CrowdFlower dataset. The first two bar plots show the accuracy (a) and the average recall (b) of the predictive tweet sentiment classes. The second two plots are the confusion matrices of two workers(with the true label on the left axis and worker label on the right axis) which represent an inferred worker's profile of a pessimist worker (c) and an extremist worker (d).

over the true sentiment variables t_i and dictionary variables $\boldsymbol{w}_d^i$ and $\boldsymbol{w}_d^c, \forall i, d, c$. The worker models manage the inference over worker variables for defined batches of workers. To define the worker models, we split the set of workers $\boldsymbol{K}$ into a complete and disjoint partition: $\boldsymbol{K} = \{\boldsymbol{S}_1, \ldots, \boldsymbol{S}_z\}$ with $z = \sqrt{K}$. Each sub-set $\boldsymbol{S}_i$ is associated with a single worker model instance, which infers the confusion matrices for the workers in S_i. Then, the inference of all the sub-models is connected together through the shared variables t_i. In particular, we iteratively run message passing based inference in each of the sub-models in a scheduled order for a number of passes until all the parameters of the posterior distributions have converged. This implementation of BCCWords has several advantages in terms of scalability. First, the space complexity is reduced by running local inference in simpler sub-models and merging the results using only a few messages exchanged between the connecting factors. Second, we can parallelize the computation of each sub-model with an evident saving of time complexity. Using just the first of these techniques, we were able to train BCCWords on the 569,786 sentiment judgments of the CrowdFlower dataset in approximately 50 minutes on a standard machine with 3.60 GHz CPU and 16GB RAM.

Shared Task Evaluation

The results of BCCWords compared against the majority voting (MV) baseline on the CrowdFlower dataset are shown in Figure 3. The performance of each method is measured by its absolute accuracy, i.e. the percentage of agreement of the predicted tweet classes with respect to the reference set of gold labels, and its average recall, i.e. the average accuracy of predicted labels in each class. The reference set provided by the organizers includes gold labels provided by expert judges for 100 randomly selected tweets. Specifically, Figure 3a reports the accuracy of the two methods showing that BCCWords outperforms MV by 4.3% (0.92 vs. 0.96). Also, the average recall in Figure 3b shows that BCCWords outperforms MV by 2.5% (0.79 vs. 0.81). An additional experiment with a version of our method that excluded the words model from the learning algorithm, did not show any improvement in accuracy over MV. Thus, this marginal but significant improvement of BCCWords over MV means that the

Word	Negative	Neutral	Positive	Not Related	Don't know
sunny	-5.0018	-4.7870	**-3.2782**	-4.3034	-4.1851
love	-5.9899	-9.6214	**-3.9555**	-4.5247	-6.1868
cold	**-3.5667**	-6.8070	-5.6341	-4.4157	-4.6286
rainy	**-4.3818**	-6.2816	-4.7815	-5.9079	-3.8352
sunshine	-5.5193	-7.0992	-3.6744	**-3.5324**	-4.7221
rain	**-4.4268**	-4.4815	-5.1208	-5.8391	-4.4999

Table 1: Inference results of BCCWords where the values represent the log probability of each word (first column) in each class. The values in bold are the highest log-probabilities of each row.

joint combination of text correlations and sentiment judgments helps improve performance over the baseline method.

Furthermore, two example worker confusion matrices, inferred by our method, are depicted in Figure 3c and Figure 3d. These corresponds to two different worker types. The first type of profile is a worker who typically expresses pessimist judgments, mostly judging the sentiment of weather related tweets as negative or not related (Figure 3c). By contrast, the second type of worker profile reflects 'extremist' workers with more polarized opinions, who mostly vote for the classes negative, positive or not related.

Finally, Table 1 reports the log-probability distributions inferred by the model for a sub-set of six words. It can be seen that the models correctly learn that words like "sunny" and "love" have higher probability in the positive class. In contrast, the words "cold" and "rain" are more likely in the negative class. Crucially, the model learns that more ambiguous words have a higher degree of uncertainty among the classes. For example "sunshine" is almost equally likely to be positive or not related to weather. Similarly, the word "rain" is more likely to be negative or neutral, which substantially differs from the word "rainy" which typically has a more negative sentiment. In this way, the model is able to reflect the information extracted from the text in the predicted tweet sentiment.

Conclusions and Future Work

In this paper, we described BCCWords, a new Bayesian model, which we developed and used to compete in the CrowdScale – Shared Task Challenge 2013. The goal of the challenge was to compute aggregated classification answers for crowdsourced tweet sentiment analysis judgments. Our model does so by combining information from two signals: crowd judgments and text properties, in a principled Bayesian framework. In this way, the model is able to learn individual worker reliability models by means of confusion matrices and also word probability models in each sentiment class from the observed workers' judgments and the text of the tweets. We also described a scalable implementation of our model using the Infer.NET framework, which allows us to train the model on large datasets (i.e. in the order of hundreds of thousands labels). Our preliminary results show that BCCWords applied to the CrowdFlower dataset improves performance over the majority voting baseline. In addition, BCCWords is also able to learn useful latent information such as the workers' confusion matrices and the words' probability distributions for each judgment class.

We envisage that further improvements of our method can be achieved with a thorough tuning of the model hyperparameters and with an intelligent selections of significant words contained the dictionary. In addition, common patterns in workers' behaviors can be exploited for a faster learning over sparse data. We intend to investigate all these directions as part of our future work.

References

Ghahramani, Z., and Kim, H.-C. 2003. Bayesian classifier combination.

Minka, T.; Winn, J.; Guiver, J.; and Knowles, D. 2013. Infer .net 2.5. microsoft research cambridge. *See http://research. microsoft. com/infernet*.

Minka, T. P. 2001. *A family of algorithms for approximate Bayesian inference*. Ph.D. Dissertation, Massachusetts Institute of Technology.

Salton, G., and McGill, M. J. 1983. *Introduction to Moderm Information Retrieval*. McGraw-Hill.