Classification of News Articles Using Named Entities with Named Entity Recognition by Neural Network

Nick Latourette and Hugh Cunningham

1. Introduction

Our paper investigates the use of named entities as features for the classification of news articles by topic. Prior work with The McClatchy Company, and in particular *The Sacramento Bee*, informs us that categorization of news articles poses a significant challenge to newspapers interested in determining the interests of their users. Given the large volume of news articles produced (past, present, and future), an automated approach to this work is desirable.

We implement a system for article classification using the named entities contained within the article as the basis for classification. Many algorithms for text classification, naive Bayes classification, for instance, use the words within the text as features for classification. For highly similar categories or topics, like current coverage of civil strife in Syria and Egypt, the words used by articles in either category may be very similar (e.g., words like 'violence' or 'military' may be equally likely to appear in an article about Egypt as they are likely to appear in an article about Syria). We hypothesize that, for such closely-related topics, using named entities for classification will perform better than other approaches. Intuitively, if the named entity 'Assad' appears within an article then it is very likely that that article should be classified as belonging to the category 'Syria'. We use a set of 200 articles created by TIME as our data set: 100 articles categorized as covering either Syria or Egypt. The selection of TIME as a source is based only on the convenient access to a large number of pre-categorized articles there. Based on the relative similarity of the topics of civil strife in Syria and Egypt we select these categories as an appropriate sample for our investigation.

Before utilizing named entities as features for classification, we need to address the non-trivial problem of Named Entity Recognition (NER). We chose to implement our own neural network approach because of its proven success with respect natural language classification problems, its easy parallelizability, and because we were interested in implementing a deep learning algorithm. Note that we borrowed the implementation details of the neural network from PA4 of CS224n. After identifying named entities in the sample, we train a naive Bayes classifier.

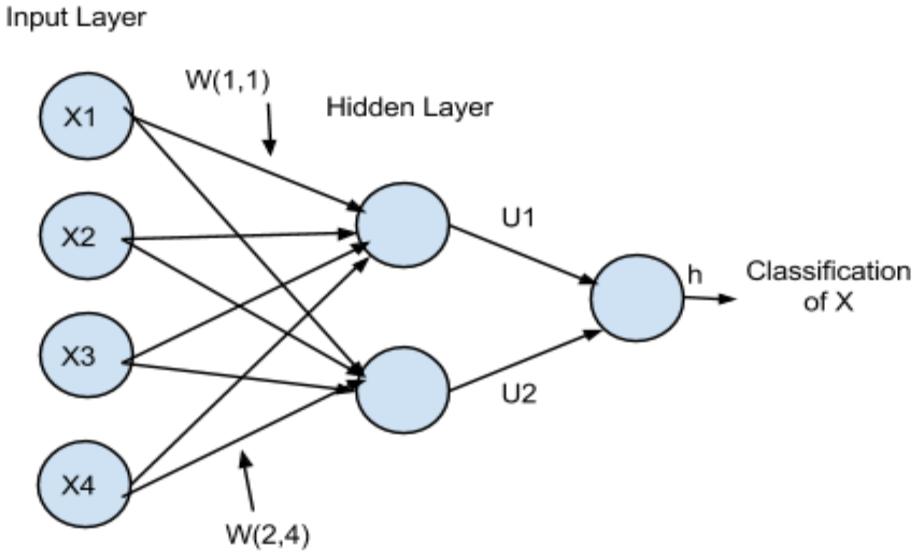
2. Prior Work

Y.C. Gui et al. provide some precedent for the use of named entities in the classification of news articles [1]. Their research is on the use of named entities for classifying news articles within categories hierarchically. Of particular interest is their focus on what they refer to as ‘Close Categories’: highly similar categories within the same hierarchy, e.g., presidential elections in different countries. Gui et al. do not describe the techniques used for NER or extraction of named entities, but found that an SVM trained on named entities outperformed one trained on terms, where ‘terms’ refers to words or phrases within the articles.

Beyond this particular utilization of named entities for classification by Gui et al., there is extensive work on the general problem of Named Entity Recognition as well as on the problem of text classification[2][3].

3. Neural Network

Knowing that the neural network would be the most challenging part of our project in terms of implementation, we decided to start with a single hidden layer and move on from there. Below is an illustration of our neural network, note that W and U are matrices representing linear transformations that are performed on the inputs to the hidden layer and the final classification respectively. In addition to the linear transformations, each node in the hidden layer will also perform a non-linear transformation to its input. Note that if we were to not do this, the whole neural network would just be performing one big linear transformation on the data, which would be considerably less powerful in terms of classifications it could represent.



To represent our words as vectors we will use a dictionary of approximately 100,000 words, where each word is represented by a 50 dimensional vector. Each dimension is a weighted feature trained by an unsupervised learning method that tries to capture a words syntactic and semantic information along with the the context in which the word is normally used[4]. Lets call this dictionary L. To decide whether a word is referring to a person or not we, use the words vector representation from dictionary L, along with the vector representations of the c-1 of the words surrounding that word as input to our neural network. To train our model we used stochastic gradient descent to try and minimize our cost function J, we used the following equations:

$$J = \frac{1}{m}\left[\sum_{i=1}^{m}\left[-y^{(i)} * log(h_i) - (1-y^{(i)}) * log(1-h_i)\right]\right] + \frac{R}{2m}\left[\sum_{j=1}^{nC}\sum_{k=1}^{H} W_{k,j}^2 + \sum_{k=1}^{H} U_k^2\right]$$

$$a = tanh(Wx), \quad h = sigmoid(U^T a)$$

$$\frac{dJ}{dU} = \frac{1}{m}\left[\sum_{i=1}^{m} a_i * (h_i - y_i)\right] + \left(\frac{R}{m}\right) * U$$

$$z = \left[U_1 * (1-a_1^2), \; U_2 * (1-a_2^2), \; U_3 * (1-a_3^2), \; ....\right]^T$$

$$\frac{dJ}{dW} = \left[\frac{1}{m}\sum_{i=1}^{m} z_i * x^i * (h_i - y_i)\right] + \left(\frac{R}{m}\right) * M$$

$$\frac{dJ}{dx_j^{(i)}} = (h_i - y_i)\sum_{k=1}^{H} U_k * (1-a_k^2) * W_{k,j}$$

Training and deciding our parameters for the neural network

Our data set to train and test the neural network consisted of blocks of text where each word in the text is labeled with a 0 or a 1, a one indicating that the word pertains to a person and a zero indicating that the word does not. In total our blocks of text had 200,000 labeled words in them. In order to properly train our model and tune the parameters, which consisted of a learning rate, context size, hidden layer size, the number of iterations of the gradient descent algorithm, and our choice of the regularization constant R, we decided to implement k-fold cross validation, where we chose k to be 20. From the models we tested we chose the best average F1 score whose runtime was below a certain threshold. Our resulting choice was as follows:

Context Size = 5 Learning Rate = 0.001 Iterations = 5
Hidden Network Size = 50 Regularization Constant = 0.001

Average Precision of Model = 0.8200
Average Recall of Model = 0.6205
Average 51 Score of Model = 0.7064

4. Classification

We implement a naive Bayes classifier, limiting the size of the vocabulary to consist only of the named entities identified by the neural network. The performance of this classifier is compared to a typical naive Bayes classifier whose vocabulary consists of all words encountered in the sample articles (common words like ‘a’, ‘as’, etc. excluded). We partition our data randomly into five folds of forty articles each for cross-validation and use each fold as the validation set once for both the named entity naive Bayes classifier and the baseline naive Bayes classifier.

The baseline naive Bayes classifier correctly identified the category of the articles in the validation subset with an average accuracy of 73%. Our own naive Bayes classifier using a vocabulary consisting of named entities performed worse in correctly categorizing articles in the validation set with an average accuracy of only 67%.

5. Conclusion and Future Work

Currently we are using named entity mentions rather naively. If many articles of a topic X mention person Y in our training set, then we are more likely to classify any article that mentions Y as belonging to topic X. However, if our training set does not have any mentions of person Y, then identifying that an article mentions Y is useless during our classification. Several features of our data set may have also contributed to poor performance of our classifier: given a U.S.-centric perspective in many articles, the entities ‘Barack Obama’ and ‘John Kerry’ appear in many articles from both categories so that these entities hinder rather than aid in correct classification. Additionally, many articles contain named entities not mentioned in any other articles in the data set, such as persons interviewed by the author, and the identification of these entities does not aid in classification at all.

The performance of our classifier could be improved by disambiguating named entities to associate them with their real world identities. This would allow us to discard entities not directly associated with either category. Our classification could also be improved by a more robust means of named entity recognition that would identify organizations of nations as named entities rather than only people.

6. References

[1] Gui, Yaocheng, et al. "Hierarchical Text Classification for News Articles Based-on Named Entities." *Advanced Data Mining and Applications*. Springer Berlin Heidelberg, 2012. 318-329.
[2] McCallum, Andrew, and Kamal Nigam. "A comparison of event models for naive bayes text classification." *AAAI-98 workshop on learning for text categorization*. Vol. 752. 1998.
[3] Nadeau, David, and Satoshi Sekine. "A survey of named entity recognition and classification." *Lingvisticae Investigationes* 30.1 (2007): 3-26.
[4] Huang, Eric H., et al. "Improving word representations via global context and multiple word prototypes." *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*. Association for Computational Linguistics, 2012.