Project Summary: Document Classifier

Arjun Patel[1], Harshil Prajapati[1], and Bowen Song[1]

[1]Department of Electrical and Computer Engineering, Boston University

March 26, 2018

1 Problem Definition

This project explores machine learning algorithms for document classification. The specific application of the project is sensitive to datasets. The contribution of the project is to compare and contrast algorithm performances based on results and computation efficiency with respect to different types of datasets.

2 Literature Review

Document Classification is a traditional problem where usually a bag-of-words approach is to extract features and are used for supervised classification using Naive Bayes or Support Vector Machine (SVM) algorithms [1].

In classical papers, the order of words is ignored, but the order can be included by using Term FrequencyInverse Document Frequency (tf-idf) scheme where basic identification of sets of words that are discriminative for documents in the collection but reveals little in the way of inter- or intra- document statistical structure [2]. To overcome the short commings of tf-idf, latent semantic indexing was introduced in [3]. It uses a singular value decomposition of the X matrix to identify a linear subspace in the space of tf-idf features that captures most of the variance in collection. This was modified by [4] to fitting the model to data using maximum likelihood or Bayesian methods known as Probabilistic latent semantic indexing (pLSI). Latent Dirichlet allocation (LDA) considers exchangeable representations for documents and words, we need to consider mixture models that capture the exchangeability of both words and documents [5]. Recent study shows use of Convolutional Neural Network (CNN) [6] to get improved results.

Due to the size of documents, different techniques like topological spaces for developing Information Retrieval System (IRS) [7], Ontology-Based Feature Vector Generation [8] have been developed which can effectively reduce the computation while extracting features that aids in classification. Kernel methods (KMs) are an effective alternative to explicit feature extraction. Another way to efficiently extract features is based on finding conditional probability using n-gram models [9] .

More over the traditional supervised learning techniques, research has been done to improve prediction in k-NN by using pyramidal decomposition [10], skip-gram and paragraph vectors-distributed bag of words (PV-DBOW) with multiple discriminant analysis [11], expansion

method and Powerset-label mechanism for the short hierarchical classification using the Support Vector Machine (SVM) classifier [12], Bayes-optimum decision with the maximum margin principle yields kernels for SVMs [13].

3 Proposed Work

The project concerns algorithms including:

- Bag-of-words model with Naive Bayes assumption
- Multi-class logistic regression (also known as maximum entropy classifier)
- Sensing-aware kernel SVM [13].
- Unigram model
- Markov model.

This project also includes stop-words filtering based on stop-words vocabulary list and term frequencyinverse document frequency (tfidf) in attempt to improve performances for each model. The project learns from different datasets; the overview of the datasets is included in later chapters. In addition to discussing algorithm performances, the project also concerns the importance of preserving word ordering within a document with respect to classification accuracy. The project attempts to explore correlations between word vectors generated from Google word2vec algorithm and its occurrences.

3.1 Model and Algorithms

- Naive Bayes Classifier
  The Naive Bayes classifier with bag-of-words model is considering all features as conditionally independent given the class label and evaluated using MPE decision rule. The prediction performance considers to be the baseline for our prediction accuracy.

$$P(Y_j|\boldsymbol{x}_j) \propto P(Y_j)\prod_{i=1}^{d_j} P(w_{i,j}|Y)$$

- Logistic Regression

$$NLL(\theta) = \sum_{j=1}^{n} ln(\sum_{k=1}^{m} e^{\boldsymbol{w}_k^T \boldsymbol{x}_j^{ext}}) - \sum_{k=1}^{m} \boldsymbol{w}_k^T (\sum_{j=1}^{n} 1(y_j = k) x_j^{ext}$$

$$\nabla_{\boldsymbol{w}_y} NLL(\theta) = \sum_{j=1}^{n} \left(\frac{e^{\boldsymbol{w}_y^T \boldsymbol{x}_j^{ext}}}{\sum_{k=1}^{m} e^{\boldsymbol{w}_k{}^T \boldsymbol{x}_j^{ext}}} - 1(y_j = y) \right) \boldsymbol{x}_j^{ext},\ y = 1, \ldots, m.$$

$$f(\theta) := NLL(\theta) + \frac{\lambda}{2} \sum_{k=1}^{m} ||\boldsymbol{w}k||_2^2,\ \lambda > 0$$

$$\nabla_{\boldsymbol{w}_y} f(\theta) = \nabla_{\boldsymbol{w}_y} NLL(\theta) + \lambda \boldsymbol{w}_y, y = 1, \ldots, m.$$

  As MPE/MAP Classifier, the decision is based on MPE rule:

$$h_{MPE}(x) = arg\ max y \in \{1, \ldots, m\} p; (y|\boldsymbol{x}, \theta)$$

- Sensing aware kernel SVM
  Something

- N-gram

  One of the algorithm models we plan to us n-grams, specifically unigram and bigrams. The n-grams model uses the probability of each word appearing. For unigram, it considers each word as independent of all other words. Therefore, for a sentence to appear, it would be the probability of each word appearing multiplied together. For bigram, it takes the probability of the first word appearing multiplied by the probability of the second word appearing given the first word and so. The probability of the word depends on the probability of the all the previous words that appear before it in a sentence. [14].

$$\textit{Unigram:} \quad P(w_{1,n}) = P(w_1).P(w_2) \cdots P(w_n) \tag{1}$$

$$\textit{Bigram:} \quad P(w_{1,n}) = P(w_1).P(w_2 \mid w_1) \cdots P(w_n \mid w_{n-1}) \tag{2}$$

  In the n-gram models, the features can become very large. For example, if there are D features, in unigram feature size would be D but in bi-gram the feature size will be D2. But with every increasing n, the frequencies of each one appearing decreases. Due to this very large, the efficiency of the model is not great and can run into memory problems. One way to make the model more efficient is by setting a threshold frequency of appearance [9].

- TF-IDF
  Another way to do thresholding aside from just word frequency is by determine how important a word is in determining the class of a document. To determine how important a word is we will be using tf-idf weighting. TF-IDF weighting takes into consideration word frequency as well as document frequency. The weight of the word is the frequency of a word multiplied by the inverse document frequency. A word has the highest weight when with a high word frequency and a low document frequency. Using the tf-idf, we can set a threshold, which would decrease the feature size and remove words that dont mean much, for example stop words [15].

3.2 Code and Dataset

We have multiple datasets in which we want to implement these algorithms. Initially we want to start by using the news group data set, due to previous use of the data set. This dataset has 20 class, 11,314 training documents and 7,532 test documents. The dataset is already preprocessed in a form which can be used. It is broken down into document ID, word ID and word count. Once we test the algorithms on the news group data set, we will test it on another dataset, Reuters21578 but will be using the ModApt version, which is a smaller overall dataset from the Reuters21578 dataset [16]. In this dataset there are 5,945 training documents and 2,347 testing documents. This dataset has also been preprocessed. Finally we want to run these algorithms on the dataset obtained from a Kaggle project, Personalized Medicine: Redefining Cancer Treatment [17]. The data has not been preprocessed so we have the raw files. The data we have is currently, a text file of each paper. There 3,320 documents for training and 5,667 documents for testing. There are more test documents due to Kaggle adding machine generated documents, but these machines generated documents can be removed and the test dataset will be down to 987 documents. There is a total of 9 classes in this dataset. But the biggest job with this dataset is all the preprocessing has to be done by us and made into a usable form, like the other datasets.

3.3 Minimum Achieviable Plan

We would like to implement and compare all the algorithms but if we aren't able to complete all the algorithms we will try to implement atleast Naive Bayes, Logistic Regression, Traditional SVM ,N-gram.

4 Conclusion

In this project, we will be exploring different document classification algorithms. We will be using the Nave Bayes, Logistic Regression, SVM (kernel sensing), and n-grams models. Each of these three algorithms will be tested on three datasets: newsgroup dataset, Reuters21578 dataset and Personalized Medicine: Redefining Cancer Treatment (Kaggle Project) dataset. Each of the algorithms will be taking into account TF-IDF to give each word value of importance. At the end we want to be able to compare the performances of each algorithm in terms of accuracy of classification and computational efficiency across all three datasets.

Division of Labor

- Arjun Patel
  - Logistic Regression
  - TF-IDF
- Harshil Prajapati
  - Pre-Processing of Personalized Medicine: Redefining Cancer Treatment Dataset
  - N-gram
- Bowen Song
  - Naive Bayes
  - Sensing Aware Kernal SVM

References

[1] D. S. Sachan, M. Zaheer, and R. Salakhutdinov, "Investigating the working of text classifiers," *arXiv preprint arXiv:1801.06261*, 2018.

[2] P. Maes, "Agents that reduce work and information overload," in *Readings in Human–Computer Interaction*. Elsevier, 1995, pp. 811–821.

[3] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, "Indexing by latent semantic analysis," *Journal of the American society for information science*, vol. 41, no. 6, p. 391, 1990.

[4] T. Hofmann, "Probabilistic latent semantic analysis," in *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*. Morgan Kaufmann Publishers Inc., 1999, pp. 289–296.

[5] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *Journal of machine Learning research*, vol. 3, no. Jan, pp. 993–1022, 2003.

[6] Y. Kim, "Convolutional neural networks for sentence classification," *arXiv preprint arXiv:1408.5882*, 2014.

[7] B. Parlak and A. K. Uysal, "On feature weighting and selection for medical document classification," in *Developments and Advances in Intelligent Systems and Applications*. Springer, 2018, pp. 269–282.

[8] M. K. Elhadad, K. M. Badran, and G. I. Salama, "A novel approach for ontology-based feature vector generation for web text document classification," *International Journal of Software Innovation (IJSI)*, vol. 6, no. 1, pp. 1–10, 2018.

[9] J. Fürnkranz, "A study using n-gram features for text categorization," *Austrian Research Institute for Artifical Intelligence*, vol. 3, no. 1998, pp. 1–10, 1998.

[10] P. Heroux, S. Diana, A. Ribert, and E. Trupin, "Classification method study for automatic form class identification," in *Pattern Recognition, 1998. Proceedings. Fourteenth International Conference on*, vol. 1. IEEE, 1998, pp. 926–928.

[11] P. Lauren, G. Qu, F. Zhang, and A. Lendasse, "Discriminant document embeddings with an extreme learning machine for classifying clinical narratives," *Neurocomputing*, vol. 277, pp. 129–138, 2018.

[12] Z. F. Salih and S. Tiun, "Term expansion and powerlabel set for multi-label hierarchical on short document classification," *International Journal of Applied Engineering Research*, vol. 13, no. 1, pp. 539–544, 2018.

[13] W. Ding, P. Ishwar, V. Saligrama, and W. C. Karl, "Sensing-aware kernel svm," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on.* IEEE, 2014, pp. 2947–2951.

[14] F. Song and W. B. Croft, "A general language model for information retrieval," in *Proceedings of the eighth international conference on Information and knowledge management.* ACM, 1999, pp. 316–321.

[15] H. Schütze, C. D. Manning, and P. Raghavan, *Introduction to information retrieval.* Cambridge University Press, 2008, vol. 39.

[16] "Collection of datasets," http://www.cad.zju.edu.cn/home/dengcai/Data/TextData.html.

[17] "Kaggle data: Personalized medicine: Redefining cancer treatment," https://www.kaggle.com/c/msk-redefining-cancer-treatment.

[18] P. Ishwar, "Lecture notes in learning from data (ec 503)," January - May 2018.