PERFORMANCE COMPARISON OF VARIOUS CLASSIFICATION ALGORITHMS FOR TEXT CLASSIFICATION

CSC 565 Data Mining - Final Project Report

Due on April 29, 2016 at 11:59pm

Professor Predrag Radivojac CSC 565

INDIANA UNIVERSITY

Anirudh Kamalapuram Muralidhar | Praneet Vizzapu | Santhosh Soundararajan

.

PERFORMANCE COMPARISON OF VARIOUS CLASSIFICATION ALGORITHMS FOR TEXT CLASSIFICATION

Team Members:

Anirudh Kamalapuram Muralidhar anikamal@umail.iu.edu

Santhosh Soundararajan soundars@umail.iu.edu

Praneet Vizzapu pranvizz@umail.iu.edu

Guidance under Professor:

Johan Bollen

Associate Professor

SOIC, Indiana University

Significance

In the past years we have seen a humongous growth in the volume of text documents available on the Internet, digital libraries and news sources. Automatic text classification or document classification, which is the task of assigning unlabeled text documents to pre-assigned classes of documents, is an important task that can help both in organizing data as well as in the text based information retrieval.

With this in the mind, we have been inspired to take upon this project and idea and work on.

Objectives

Text classification is a data mining process where an initial model is trained with the training data and testing of the model is carried out after the model is Sufficiently trained. Based on our research, we are primarily concern in delving into the following questions:

i. How much of train data can be considered Sufficient?
ii. Which are the most robust text classification models and how they compare in terms of amount of training data required to reach optimal performance levels?
iii. How the tendency to overfit behave in different classifications?
These are a set of very interesting research question that can help time critical information retrieval systems to adopt models according to its needs.
There are many algorithms based upon which we can build a model for text classification but in our project

we are going to implement and analyze two highly successful methods: *a) Naive Bayes classifier b) Artificial Neural Networks.*

4.a Background

Text Classification

Text classifications are mainly approached based on machine learning methodologies: a common way is to builds a model by learning, from a set of pre classified documents known as training data, and then characterizing the test data which is the process of evaluation.
So the requirement of information classification retrieval systems, that retrieve texts and classifies it based on a nature and content of the text.

The traditional approach known as knowledge engineering which involve manual labeling of text data, is highly unsuitable for this age of big data, hence we resort to Machine Learning based classifier.

Existing Machine Learning approach to Text Classification:

The Machine Learning approaches at present relies on the availability of an initial corpus $\omega = (d1, ..., d|\omega|) \subset D$ of the documents pre-classified under the set $C = (c_1, ..., c_{|C|})$. That is, the values of the total function $\phi : (D \times C) \to (T, F)$ are known for every pair $\langle D_j, C_i \rangle \in \omega \times C$ A document d_j is a positive example of c_i if $\phi(d_j, c_i) = T$. Given a corpus ω, the above approach define the generality $g\omega(c_i)$of a category c_i as the percentage of documents that belong to c_i, i.e.:

$$g\omega(c_i) = \frac{|(d_j \subset \omega | \phi(d_j, c_i) = T)|}{|\omega|}$$

Indexing: Term Frequency-Inverse Document Frequency

Indexing strategy is one of key procedures in making the classification algorithm learn beased on the term weights and this method of indexing is widely used and is know and term frequency- document frequency indexing.

$$tf - idf(t_k, d_k) = f(t_k, d_k) \times log \frac{|T_r|}{f(T_r * (t_k))}$$

where $f(t_k, d_k)$ denotes the number of times t_k occurs in d_j , and $f(T_r * (t_k))$ denotes the document frequency of term t_k, i.e. the no. of text documents in T_r in which t_k occurs. This particular function means that the more often a particular term occurs in a document, the more it is illustrative of its content. Also the more documents a term occurs in, the less useful it is. Note that this above equation weighs the significance of a term to a document in terms of the numbers of occurrences only, thereby saying that the order in which the terms occur and their syntactic role in a document is of no importance.

Bayes Rule:

$$P(H|E) = \frac{P(H)_{[belief]}\ P(E \mid H)_{[Likelihood]}}{P(E)_{[norm]}}$$

So $P(E \mid H)$ is significantly different from the P(E) because, $P(E \mid H)$ is the conditional probability or the probability measure, given a set of beliefs/prior knowledge.

This idea of calculating the probability of the document being assigned into a certain label based on the prior knowledge of existing training documents labels is our as Naive Bayes approach.

Introduction to Neural Networks

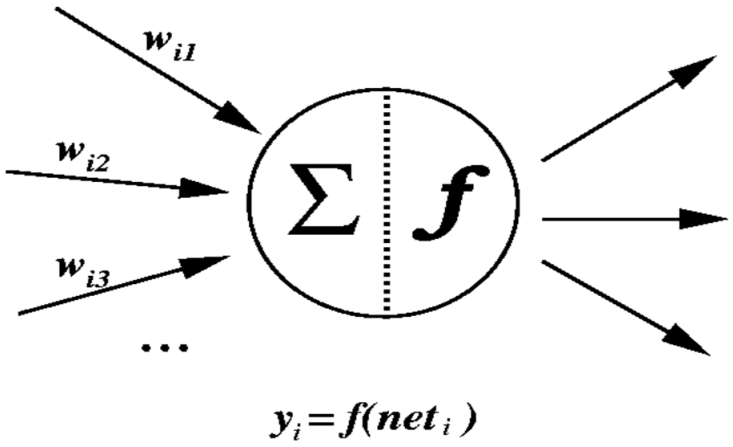
Neural networks are one of the most efficient and the hottest algorithm of the current age. The neural network tries to implement the function of the brain. The human brain consists of billions of neurons and each neuron is connected to other neurons through synapses. This forms a massive parallel information processing system. On the basis of this the neural network algorithm is formed.

A simple artificial neural network is explained below.

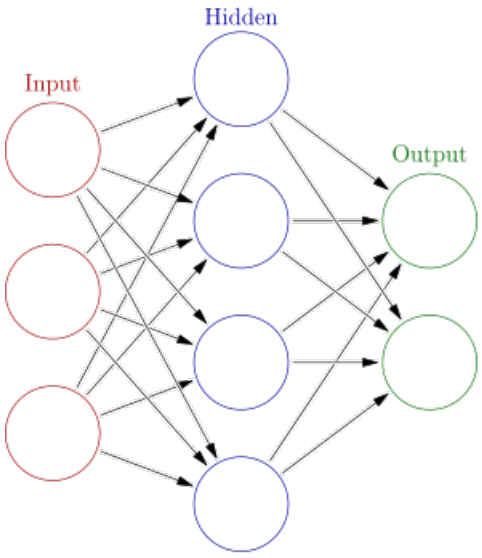
Simple Artificial Neuron,

Each computational element is called a node or unit. It gets its input from an external source or from other neuron models. Here each input is associated with a particular weight and the unit computes some function f for the sum of weights.

$$y_i = f(\sum_j w_i j y_i j)$$



[Image Source: https://www.willamette.edu/ gorr/classes/cs449/ann-overview.html] The function f is often called as an activation function. This basic idea can be extended for the neural networks algorithm, where we will have inputs given to a set of neurons which communicate with neurons in the hidden layers several times which finally results in the output layer. Basic neural network model.



[Image Source: Wiki]

4.b Previous work on this problem

There are many existing works in this area of text classification particularly the Naive Bayes classifiers and Neural networks based classifiers when high accuracy is desired. Most of these models are built based on Apriori algorithm, Association rule, confidence intervals, Nave Bayes classifier and support vector machines. The existing works are listed and sited below

[1] **Text classification with support Vector machines** Learning with many relevant features by Thorsten Joachims, Univeristy of Dortmund, Link: http://goo.gl/KgJ5Qx

This papers explains the text classification done with the help of SVM classification model, where they have achieved an accuracy score of 86%.

[2] **Effective use of word order for text categorization with convolutional neural networks** by Rie Johnson and Tong Zhang, Rutgers University, Link: http://goo.gl/S2VlOE

This paper explains the implementation of text classification with the algorithm convoluted neural network.

[3] **High performance feature selection for text classification** by Monica Rogati and Yiming Yang, Carnegie Mellon University, Link: http://goo.gl/8ON7UL

This papers compares various algorithm such as Naive Bayes, k-nearest neighbor and Support Vector Machine on the text classification.

[4] **A Comparison of Event Models for Naive Bayes Text Classification**

link: [*http* : *//citeseerx.ist.psu.edu*]

This Paper does a similar job to our research which does a comparison of various models on the Naive classification.

4.c How the existing work compares with our idea

The work that we have currently undertaken already exists, but the scope of improvement is still high. So we would like to take this opportunity to build a classifier model based on Naive Bayes and Neural Network. We are trying to replicate the existing work and from there, we wanted to work on improving the accuracy of the classifier and also handle problems such as overfitting of training data.

So to summerize, we showed a bayesian statistical approach and neural networks based approach for building a classification models, in this scenario our work is going to be about building both the models with improved accuracy focused on neural network model and as discussed before we would like to specifically answer few research questions concerning

-Implementing and analysing for the most robust algorithmic model

-We target Naive Bayes and Neural Networks model

- Relation between the amount of Train data and Model Outfitting - Model Outfiting across different models

5a. Format of the data and acquisition

For this project we will using the dataset provided by Carnegie Mellon University, (link in reference). The dataset contains 20,000 documents where we have 1000 documents per newsgroup.

The data downloaded will have 20 folders each with 1000 articles. All the data provided are in only English language.

5b. Description of proposed method and implementation

In this project we would first implement the document classification using the naive bayes approach (multinomial model) and evaluate its performance using cross validation and by calculating other parameters.

Next, we would implement the same experiment with the implementation of convoluted neural network and again evaluate its performance using cross validation and by calculating other parameters.

The input of the network is a document named D which has sequence of words which can be labeled as w1, w2.. wn. Whereas the output of the network will be the class of each document. Here we use $P(k|D,\theta)$ to denote the probability of document D belonging to a particular group, where θ is the parameter that will be used.

Naive Bayes Classifier This is a probabilistic approach that uses prior knowlege into account to predict as a probability measure.

Posterior poisson inference for the binary independance classifier is given by,

$$Pr(Y_1....,Y_n = y_n \mid \theta) = \prod_{i=1}^{n} p(y_i|\theta) \ = c(y_1,....,y_n)\ \theta^{\sum y_i} e^{-n\theta} \ and$$

$$p(\theta) = dgama(\theta,\ a,\ b) = \frac{b^a}{\Gamma(a)} \theta^{a-1} e^{-b\theta}$$

where the a,b are the probability weights associated with the positive outcomes vs the negative outcomes in the Document classification model.

The probability the document known priod to us is going to be represented by a vector Y_i and this uses binary-valued vector notations for documents. Since we know both $Pr(Y_i|\theta)$ *and* $p(\theta)$, we are confident that these two can fetch us the posterior distribution of the $P(\theta|Y_i)$

For the above joint distribution of the classifier to be true, **the prior distribution of** Y_1 *should be i.i.d. Poisson with mean* θ [*i.i.d.* $-$ *conditionally independent and identically distributed*]

And we say that

$$\sum Y_{prior} \textit{ is a sufficient statistic to come to a Poisson Inference}$$

And this predictive distribution does not depend on any unknown quantities. All it works with is the Y_1. If it was depended on any other variable containing uncertainty, we would not be able to use it to make predictions.

Our Approach to Neural Network Model

In this project we combine word with its context to present a word, this is because it can help in obtaining the meaning more precise way.

We define $c_l(w_i)$ as the left context of the word w_i and $c_r(w_i)$ as the right context of the word w_i. Both of these are dense vectors with $|c|$ real valued elements.

The left side context $c_l(w_i)$ is calculated as follows

$$c_l(w_i) = f(W^{(l)}c_l(w_i) + W^{(sl)}e(w_{i-1}))$$

$$c_r(w_i) = f(W^{(r)}c_r(w_i) + W^{(sr)}e(w_{i-1}))$$

Where,

$e(w_{i-1})$ is the word embedding of the word (w_{i-1}) and $c_l(w_{i-1})$ is the left side context of the previous word.

W^l is the matrix that transforms the hidden layer (context) into the next hidden layer.

W^{sl} is the matrix that is used to combine the semantic of the current word with the next word left context.

And f is a non-linear activation function.

In a similar fashion the right side context is also calculated, so finally the word w_i can be represented as

$$x_i = [c_l(w_i); e(w_i); c_r(w_i)]$$

Finally we apply linear transformation with a tanh activation function and send results to the next layer.

$$y_i = tanh(Wx_i + b)$$

Text representation learning
When all the representation of the words are calculated, we apply a max pooling layer.

$$y = max(y_i)$$

The pooling layer converts texts with various lengths into a fixed length vector. With the the information about the entire text is being captured. So finally we would define the output layer as

$$y_out = Wy + b$$

The probability can be finally calculated as

$$p_i = \frac{exp(y_i)}{\sum_{k=1}^{n} exp(y_k)}$$

6. Individual Roles:

Anirudh K M: Will mainly be responsible for developing program scripts for neural network implementation and evaluating the performance criteria. He has been given this role because he has shown good programming skills and some amount of experience with neural networks through sentimental analysis course.

Santhosh S: Will mainly be responsible for developing scripts for naive bayes classifier and developing reports graphically. He has been assigned this role because he has taken courses on bayesian statistics and data visualization which makes him apt to develop scripts for this.

Praneet Vizappu:Will be responsible with developing scripts for cross validation, evaluating the model performance and documentation of the projects. He has been given this role because he has implemented cross validation techniques during his previous projects and has vast experience in documentation.

References:

1. Text classification and Naive Bayes - https://web.stanford.edu/class/cs124/lec/naivebayes.pdf.

2. A comparison of event models for Naive Bayes Text Classification by Andrew McCallum and Kamal Nigam, Carenegie Mellon University- http://www.cs.cmu.edu/ knigam/papers/multinomial-aaaiws98.pdf

3. Naive Bayes in Nutshell - $https://www.cs.cmu.edu/\ tom/10701_sp11/slides/GNB_1 - 25 - 2011.pdf$

4. Web page classification method using neural networks by Ali Selamat, Sigeru Omatu.

5. Recurrent convolutional neural networks for text classification by Siwei Lai, Liheng Xu, Kang Liu and Jau Zhao

6. Text classification with support Vector machines Learning with many relevant features by Thorsten Joachims, Univeristy of Dortmund, $Link : http://goo.gl/KgJ5Qx$

7. Effective use of word order for text categorization with convolutional neural networks by Rie Johnson and Tong Zhang, Rutgers University, Link: http://goo.gl/S2VlOE

8.High performance feature selection for text classification by Monica Rogati and Yiming Yang, Carnegie Mellon University- $http://goo.gl/8ON7UL$