International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 20 (2018) pp. 14793-14798

Effective way of Ranking Universities based on Online InfoBase Using Analytical Hierarchy Process

Pragatheeswaran Sridharan[1], Thirunavukkarasu Varatharajan[2], Arun Amaithi Rajan[3]

Department of Computer Science and Engineering, College of Engineering Guindy, Anna University, Chennai, India.

Abstract

This paper proposes a ranking system that ranks universities based on their website contents (online InfoBase). This approach uses text mining technique and machine learning technique for classifying documents of each webpage into multiple categories and used analytical hierarchy process (AHP) for ranking. The proposed system outputs ranks of universities. Nowadays it becomes inevitable for every institute to represent quality content in their websites. This applies to universities also. Taking note on this, our system ranked those universities according to their content exposure. Crawling has been done in a focused manner where only the website's specified contents were downloaded. Then each webpage has been Categorized using statistical analysis. Then the universities are ranked using the Analytical Hierarchy Process (AHP) based on the importance that a university has given to the features for ranking. This measure can be used to see the quality of the contents in the websites of universities. Our results has been validated by comparing our results with other existing university ranking system's results.

Keywords: Ranking system, Online InfoBase, AHP and Classification.

INTRODUCTION

Text mining is the process of deriving high-quality information from text. Text mining usually involves the process of structuring the input text (usually parsing, along with the addition of some derived linguistic features and the removal of others, and subsequent insertion into a database), deriving patterns within the structured data, and finally evaluation and interpretation of the output. High quality in text mining usually refers to some combination of relevance, novelty, and interestingness. Typical text mining tasks include text classification, text clustering, concept/entity extraction, production of granular taxonomies, sentiment analysis, document summarization, and entity relation modelling (i.e., learning relations between named entities). Text classification is a subfield of text mining that categorize text files based on specified features. A typical application is to scan a set of documents written in a natural language and either model the document set for predictive classification purposes or populate a database or search index with the information extracted. The intellectual classification of documents has mostly been the province of library science, while the algorithmic classification of documents is mainly in information science and computer science. Many approaches have been described for Text categorization. A few of them are expectation maximization, naive bayes classifier, tf–idf, instantaneously trained neural networks, latent semantic indexing, support vector machines, artificial neural network, k–nearest neighbour algorithms. In this paper, we proposed a system to rank universities based on classifying web contents using linear regression method. There exist many ranking system each produces different kind of results so we decided to overcome this by proposing a new system which considers almost all attributes for ranking an university. Our intention is to help universities to identify potential areas of progress with respect to specific academic performance indicators. Similar to other ranking systems, this system is neither exhaustive nor definitive, and is open to new ideas and improvements. The current ranking system will be continuously upgraded based on ongoing research and the constructive feedback of universities. Many ranking systems produces different results, but we need a precised ranking system based on their website contents. Ranking system based on web contents are very few and hence this system would be a good contribution.

The rest of this paper is organized as follows: Section 2 describes text classification techniques, Section 3 discusses proposed system, Section 4 presents result and discussion and Section 5 finally gives conclusion and future works.

LITERATURE SURVEY

This section gives a survey of the possible approaches to text classification techniques. We want to extract web contents from all university websites and classify into specified categories. Web contents has to be classified correctly based on feature set. Thus, this survey helped us to analyse the various existing approaches to text classification and decide the one which would best cater to our needs.

Naive Bayes Classifier

The naive bayes classifier is based on bayes theorem with independence assumptions between predictors. A naive bayesian model is easy to build, with no complicated iterative parameter estimation which makes it particularly useful for very large datasets. Despite its simplicity, the naive bayesian classifier often does surprisingly well and is widely used because it often outperforms more sophisticated classification methods.

Bayes theorem provides a way of calculating the posterior probability, $P(c|x)$, from $P(c), P(x)$, and $P(x|c)$ in Equation 1. Naive bayes classifier assume that the effect of the value of

a predictor (x) on a given class (c) is independent of the values of other predictors. This assumption is called class conditional independence [4].

$$P(c|x) = \frac{P(x|c)P(c)}{P(x)} \quad ..(1)$$

$$P(c|X) = P(x_1|c) \times P(x_2|c) \ldots . \times P(x_n|c) \times P(c)$$

Where,

- $P(c|x)$ is the posterior probability of class (target) given predictor (attribute).
- $P(c)$ is the prior probability of class.
- $P(x|c)$ is the likelihood which is the probability of predictor given class.
- $P(x)$ is the prior probability of predictor.

Tf-idf Technique

In information retrieval, term frequency and inverse document frequency is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus It is often used as weighting factor in information retrieval, text mining, and user modelling. The tf-idf value increases proportionally to the number of times a word appears in the document, but is offset by the frequency of the word in the corpus, which helps to adjust for the fact that some words appear more frequently in general. Nowadays, tf-idf is one of the most popular term-weighting schemes. For instance, 83% of text-based recommender systems in the domain of digital libraries use tf-idf.

Variations of the tf-idf weighting scheme are often used by search engines as a central tool in scoring and ranking a document's relevance given a user query. Tf-idf can be successfully used for stop-words filtering in various subject fields including text summarization and classification.

One of the simplest ranking functions is computed by summing the tf-idf for each query term. All tf-idf formulae are given below from Equation 2 to 4 [2].

$$tf(t,d) = \frac{f_d(t)}{max\ f_d(w)\ Where\ w \in d} ..(2)$$

$$idf(t,D) = log_e(\frac{|D|}{|\{d \in w: t \in d\}|}) \quad ..(3)$$

$$tf - idf(t,d,D) = tf(t,d) \times idf(t,D) \quad ..(4)$$

Where,

- $f_d(t)$: $frequency\ of\ term\ t\ in\ document\ d$
- $D :=\ corpus\ of\ documents$

Term Frequency

In the case of the term frequency $tf(t,d)$, the simplest choice is to use the raw frequency of a term in a document, i.e. the number of times that term t occurs in document d. If we denote the raw frequency of t by $f(t,d)$, then the simple tf scheme is $td(t,d) = f(t,d)$.

Boolean "frequencies": $tf(t,d) = 1$ if t occurs in d and 0 otherwise; logarithmically scaled frequency: $tf(t,d) = 1 + log\ f(t,d)$, or zero if $f(t,d)$ is zero; augmented frequency, to prevent a bias towards longer documents, e.g. raw frequency divided by the maximum raw frequency of any term in the document.

Inverse Document Frequency

The inverse document frequency is a measure of how much information the word provides, that is, whether the term is common or rare across all documents. It is the logarithmically scaled inverse fraction of the documents that contain the word, obtained by dividing the total number of documents by the number of documents containing the term, and then taking the logarithm of that quotient.

Linear Regression Technique

A linear classifier achieves this by making a classification decision based on the value of a linear combination of the characteristics. An object's characteristics are also known as feature values and are typically presented to the machine in a vector called a feature vector. Such classifiers work well for practical problems such as document classification, and more generally for problems with many variables (features), reaching accuracy levels comparable to non-linear classifiers while taking less time to train and use.

Linear Regression Definition

If the input feature vector to the classifier is a real vector $\vec{x}$, then the output score is

$$y = f(\vec{x}, \vec{w}) = f(\sum_i x_i . w_i) \quad ..(5)$$

where $\vec{w}$ is a real vector of weights and f is a function that converts the dot product of the two vectors into the desired output. (In other words, $\vec{w}$ is a one-form or linear functional mapping $\vec{x}$ onto R.) The weight vector $\vec{w}$ is learned from a set of labeled training samples. Often f is a simple function that maps all values above a certain threshold to the first class and all other values to the second class. A more complex f might give the probability that an item belongs to a certain class [1]. For a two-class classification problem, one can visualize the operation of a linear classifier as splitting a high-dimensional input space with a hyperplane: all points on one side of the hyperplane are classified as "yes", while the others are classified as "no".

A linear classifier is often used in situations where the speed of classification is an issue, since it is often the fastest classifier, especially when $\vec{x}$ is sparse. Also, linear classifiers often work very well when the number of dimensions in $\vec{x}$ is large, as in document classification, where each element in $\vec{x}$ is typically the number of occurrences of a word in a document (see document-term matrix). In such cases, the classifier should be well-regularized.

The naive bayes algorithm affords fast, highly scalable model building and scoring. It scales linearly with the number of predictors and rows. The build process for naive bayes is parallelized. Naive bayes can be used for both binary and multiclass classification problems. The disadvantage is that the naive bayes classifier makes a very strong assumption on the shape of your data distribution, i.e. any two features are independent given the output class. Another problem happens due to data scarcity. For any possible value of a feature, you need to estimate a likelihood value by a frequentist approach. This can result in probabilities going towards 0 or 1, which in turn leads to numerical instabilities and worse results [4]. A third problem arises for continuous features. It is common to use a binning procedure to make them discrete, but if you are not careful you can throw away a lot of information. Tf-Idf technique is easy to compute, you have some basic metric to extract the most descriptive terms in a document, you can easily compute the similarity between 2 documents using it. Tf-idf is based on the bag-of-words (BoW) model, therefore it does not capture position in text, semantics, co-occurrences in different documents, etc. For this reason, Tf-Idf is only useful as a lexical level feature cannot capture semantics (e.g. as compared to topic models, word embedding) [2]. Linear regression technique also give rise to the analysis of feature selection. Now on whether it linear classifiers are fast to train and test, it is half true. For problems that are inherently linearly classifiable or that are well featurized (e.g. features are already linearized using human wisdom), linear classifiers are certainly simpler and faster to train and test. Note that there are tons of such seemingly simple problems in practice [5]. Among these classifiers linear regression technique is very fast to train and test, so for weighting metrics we have used tf-idf technique and to categorize we have chosen linear regression.

PROPOSED SYSTEM

Our proposed system has four phases which are

1. Data Extraction Phase
2. Learning Phase
3. Contribution Computing Phase
4. Ranking Phase

A new model for ranking the universities has been devised. The system aims at ranking universities based on their website contents (online InfoBase). We have taken four metrics for ranking such as Course diversity, Research, Quality, Knowledge. These metrics have been analyzed for ranking. Seed url has been given as the input to the focused crawler [7] which crawls all pages from the website and store it in database in data extraction phase.

In learning phase, each document in every university has been labeled using standard set of words related to the metrics for ranking. We process each document against standard set of related words in each metrics then we assign the document under the label of the metrics which has the maximum match with the standard set of related words of that metrics. Tf-idf values will be computed for each document in every university.

We have created a matrix with vocabulary of words as rows and documents as columns with their corresponding tf-idf values. The matrix will be fed into training module. Trained weights are obtained. This has been stored in vector format.

The documents of universities consider for ranking will be given as a test data. Then we will classify the documents based upon trained weights. Contribution for each metrics will be computed. Computed values of the contribution of universities for each metric will be stored in database. This will be given as input to the AHP which will rank the universities [3]. The proposed block diagram is given Figure 1. In this section, we have explained all important modules of this system.

Focused Crawler

Seed url will be given as input to the focused crawler. Focused crawler will recursively find the hyperlinks and stores this in an intermediate file. Each hyperlink will be checked whether it is part of our university content or not.

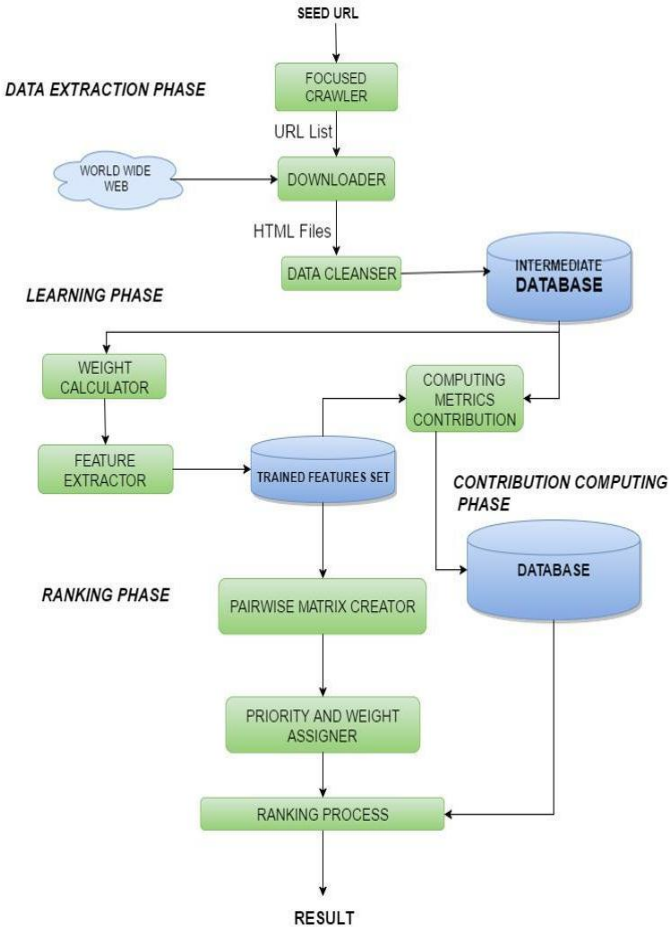

Figure 1: Block Diagram of Proposed System

Downloader and Data Cleanser

It fetches each hyperlink from the intermediate file and downloads the contents from world wide web and store that in database. It will download a file from the hyperlink only when

it contains readable contents. For example it will not download files from the hyperlink which contains jpg, jpeg file formats and it download files from hyperlinks which contains text, html formats. These web pages will be stored in separate folder under its identity. These documents are going to be processed in following modules.

Weight Calculator

This module computes term frequency and inverse document frequency for each terms in cleansed documents after eliminating stop words (of,an,etc..) using the formulae 2 to 4 after labelling process. Term frequency which measures how frequently a term occurs in a document. Since every document is different in length, it is possible that a term would appear much more times in long documents than shorter ones. Thus, the term frequency is often divided by the document length (aka. the total number of terms in the document) as a way of normalization.

Inverse document frequency which measures how important a term is. While computing TF, all terms are considered equally important. However it is known that certain terms may appear a lot of times but have little importance. Thus we need to weigh down the frequent terms while scale up the rare ones, by computing idf. This will be stored in text file with term and its corresponding tf-idf values.

Feature Extraction

Stored tf-idf value for each document in training set will be converted into matrix format. This matrix will be given as input to this module. This module will find link weight for each edge between each word in a document and metrics. This will be stored in vector format.That vector consists four columns with that metrics for ranking.

D<Course_Diversity, Research, Quality, Knowledge>

This vector cannot be changed which is set as the trained weight to compare with testing data.

Computing Metrics Contribution

Test data which has to be ranked will be given as input to this module. It will classify the documents based upon trained weights. Contribution for each metrics will be computed by finding how many documents related to that specific metric and how much the document is related to that specific metric. Computed values of the contribution of universities for each metric will be stored in database. By using these computed values we rank universities in the following modules.

Priority Assigner

In this module we assign a value for each metric considered and also assign a value how a metric is stronger or weaker than the other metrics. We obtained these values by survey over other standard ranking systems. After this we get weight for every metric by normalizing the data in the pairwise matrix. Priority for each metric will be calculated by following 3 steps

- Sum the elements in each column.
- Divide each value by its column total.
- Computer row average.

From this we will get priority for each metrics. Weights for each metric will be obtained by repeating the above steps again [6].

Ranking Process

A pairwise matrix has to be created for each metric by comparing the value contributed by each university to that metric. Row average has to be calculated then row average is assigned as the value scored by the university for that metric. Then the metric score of the university is to be multiplied with weight of the metric calculated earlier. Finally we sum up the results for each university and sort them in descending order. Hence the Ranks of the universities will be obtained. The AHP algorithm is given below. It outputs ranks of the universities.

```
repeat
repeat
data = get data from db for metric
table[ ][ ]=data[metirc1]/data[metirc2]
until count_of(metric2)
until count_of(metric 1)
repeat
sum_column(i,pairwise_matrix)
update_column(pairwise_matrix)
until (count_of(metrics))
repeat
priority(pairwise_matrix)
weight_assigner(pairwise_matrix)
until (count_of(metrics))
repeat
repeat
data = get data from db
compare_and_assign(data,weight_of_metric)
until count_of(metric)
until count_of(university)
return rank_univ(weight_assigned,univ)
```

RESULT AND DISCUSSION

In this section we have discussed the results of the system implemented and performance analysis. We have given

screenshots of metrics contribution calculation and ranking phase. In the following Figure 2 metrics contribution of Anna University has been shown. Here the contribution to each metrics which are Research, Course diversity, Quality, Knowledge has been evaluated.

```
ANNA UNIVERSITY:
Research Contribution      : 8.99
Course Diversity           : 6.19
Quality                    : 4.87
Knowledge Contribution     : 3.55
```

Figure 2: Metrics Contribution of AU

Input to the ranking phase from the database is given Table I below which shows metrics contribution of universities has to be ranked.

Table I: Database with test data

U_id	Research	Diversity	Quality	Know-ledge
IITM	18.86	27.52	10.73	11.96
AU	8.99	6.19	4.87	3.55
VIT	5.95	9.07	2.09	1.41
IITD	97.59	47.65	11.99	6.03
IISC	8.32	14.55	6.32	8.47

Ranks of given universities based on AHP is shown in the following Figure 3.

	Weight	iitd	iitm	iisc	AnnaUniversity	vit
University Rank	100.0%	48.3%	28.6%	10.9%	7.7%	4.6%
Research	53.9%	30.1%	13.8%	3.3%	4.7%	2.0%
Diversity	27.9%	14.2%	7.7%	3.4%	1.0%	1.6%
Knowledge	9.8%	0.3%	4.5%	3.0%	1.4%	0.6%
Quality	8.4%	3.6%	2.6%	1.1%	0.7%	0.3%

Figure 3: Ranks of Universities (Output of AHP)

Performance Evaluation

We have taken three existing ranking systems for evaluation of our system. Table II explains the deviation of our ranking system from existing ranking systems considered.

Table II Comparison with existing ranking systems

Univ	QS	Uni Rank	Ranking web	Our System
IITD	2	2	2	1
IITM	3	1	1	2
IISC	1	4	3	3
AU	4	5	4	4
VIT	5	3	5	5

We obtain the deviation (60%) shown in Figure 4 because, In QS World university ranking system they take a survey for each university over a certain interval of time and then rank them. We have not taken any survey details into account for ranking the university. We are considering only web contents of the universities for ranking.

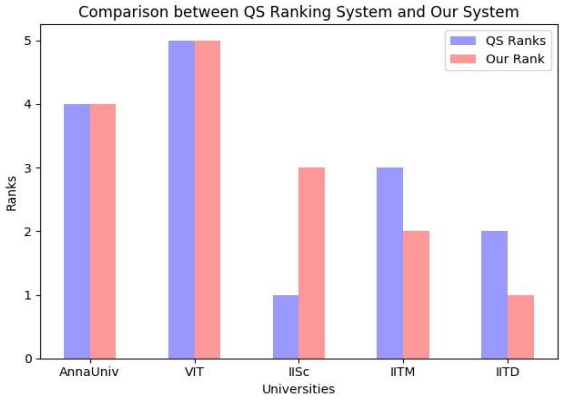

Figure 4: Comparison with QS ranking system

We obtain the deviation shown in figure 5 because, Unirank system has considered different set of metrics for evaluation which leads us to this maximum deviation. They are also taking survey in some periodical intervals.

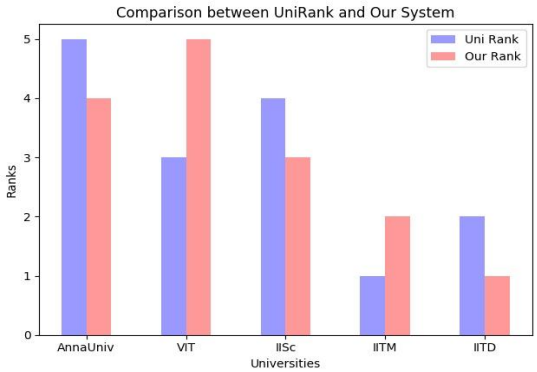

Figure 5: Comparison with Unirank

This ranking web system is more similar to our ranking system, so we got only a slight deviation. We got only 40% deviation from this ranking system which is shown in the following Figure 6.

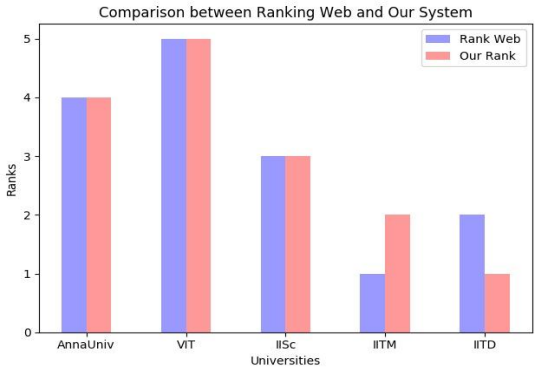

Figure 6: Comparison with Ranking web

CONCLUSION AND FUTURE WORKS

This is a ranking system which rank the universities based on their online infobase using AHP. A new model for ranking the universities has been proposed. Metrics which are taken for ranking can be changed to enhance the ranking. Any classifier algorithm with higher efficiency and less time complexity than this linear regression technique can be implemented to improve the ranking.We could make the system to update ranks automatically while website contents get updates.

REFERENCES

[1] B.Kaushik and S.Naithani., "A comprehensive study of text mining approach.", International Journal of Computer Science and Network Security, vol. 16, pp. 69–76, 2016.

[2] S.Mikacb B.Trstenjak and D.Donkoc., "Knn with tf-idf based framework for text categorization.",vol.69, pp. 1356–1364, 2014.

[3] Febronica Faustina and Tharun Balaji., "Evaluation of universities in chennai city,india using analytical hierarchy process", International Conference on Electrical,Electronics and Optimization Techniques., pp. 112–116, 2016.

[4] Febronica Faustina and Tharun Balaji., "Evaluation of universities in chennai city,india using analytical hierarchy process", International Conference on Electrical,Electronics and Optimization Techniques., pp. 112–116, 2016.

[5] Gurpreet S.Lehal Vishal Gupta., "A survey of text mining techniques and applications", Journal of Emerging Technologies in Web Intelligence, vol. 1, pp. 60–76, 2009.

[6] P Kousalya G Mahendar Reddy S Supraja and V Shyam Prasad., "Analytical Hierarchy Process Approach – An application engineering education", Mathematica Aeterna, Vol. 2, no. 10, 861 – 878, 2012.

[7] Pooja Rohilla and Ochin Sharma., "Web content Mining: An Implementation on Social websites", International Journal of Advanced Research in Computer and Communication Engineering, Vol. 4, no. 7, 108 – 111, 2015.