**How to Prepare for Placements?**

Kunal Parnami Neerav Jain Nishesh Bharech Vathsala MK

Dept. of Information Science Dept. of Information Science Dept. of Information Science Dept. of Information Science

PESIT PESIT PESIT PESIT

Bangalore, India Bangalore, India Bangalore, India Bangalore, India [[kparnami1@gmail.com](mailto:kparnami1@gmail.com)](mailto:jateenmittal0994@gmail.com)  [ostwalneerav@gmail.com](mailto:%20%20%20%20%20%20%20%20%20%20ostwalneerav@gmail.com) [bharech.nishesh96@gmail.com](mailto:bharech.nishesh96@gmail.com) vathsalamk@pes.edu

*Abstract*— **This work helps students to prepare for the placements. We provide a platform where one can find common and important questions on data structures in Java and also recommend projects based on his/her interest by applying Hierarchical Agglomerative Clustering. The clusters reveal which topics are more similar and which are unique. This knowledge will help students to choose unique projects and also projects based on his/her interest.**

Keywords— Data Structures, Java, Recommender, Content Based Filterring, Collabrative Based Filterring, Hierarchical Agglomerative Clustering, TF, IDF, Euclidean Distance, Content-less words.

# Introduction

Students in final year of Engineering face a very tough time as it marks the beginning of placement season. One has to be totally prepared and be thorough with all the important concepts and courses they have undertaken in the four years of their course. It creates a very chaotic situation as he/she is caught between strengthening his resume and improvising skills. They say you need a strong resume if you have to crack an interview and to build a good resume, you have to have good grades, a good understanding of a number of programming languages and most importantly good projects. And with the increased ideas on the Internet over the years, students find it quite difficult to find the information which is scattered all over the internet.

There are countless number of sites to ease the life of a student and help him prepare to achieve his goals. But hovering over different sites to improve their skills takes more time than understanding a single piece of code. Choosing a project which will increase the weightage of his resume is the most difficult thing to do. Students can’t decide which project to take according to their interest area to work with.

Students, like, dislike or comment on the articles they see on Internet, articles being related to interest area of the students. So, based on the user-data i.e. Student’s historical data on the educational websites, open source forums and on the blogs helps to generate user’s interest-data. Recommending projects based on interest data becomes much easier rather than ideally recommending things to any user.

Recommender Systems are useful alternative to search algorithms since they help users discover items they might not have found by themselves. Recommender Systems typically produce a list of recommendations in one of the two ways – Collaborative filtering and Content-based filtering.

Collaborative filtering approach builds a model from a user’s past behaviour as well as the similar decisions made by other users. This model is used to predict items that the user may have an interest in. Content–Based filtering approach utilize a series of discrete characteristics of an item in order to recommend additional items with the similar properties.

This project is a small effort to help those in need. Data Structures being one of the most important core subject for any IT job, by the help of this project, students can get a fair idea about Data Structures in Java. Our project consists of a set of questions on Data Structures in Java. This set of questions consists of important and common questions. Not only this project will help students excel in Data Structures but also help them build their resume. It also recommends Projects according to their interests and provides with the appropriate research paper of that project. We used content-based filtering to recommend projects. Overall it helps students in academics and improving their profile.

# methodology

## Data Set Extraction

The data set consisted of 2 phases:

Firstly, the data-set for the data structures codes were manually done by us. This data set consists of codes along with the question, the algorithm used and some sample examples.

Secondly, data-set was collected for the different projects. The data set for this project comes from the archive of past CS229(Stanford University) projects and also from the blogs in the open-source communities namely the engpapers.net.

## Processing

The project papers extracted were dealt through a series of pre-processing steps. The project papers were first converted to text file and then to feature vectors. Conversion to feature vectors involved removal of stop-words, lower–casing words and word-stemming. A Filter list was developed to remove all the “Content less words” from each paper for creation of a better and relevant word-frequency vector.

The problem of content-less words was exacerbated by the fact that the texts were project papers with different vocabularies than Standard English texts. For example, words such as “algorithm” and “learning” would be relevant for ordinary topic modeling. But in this case, every project applies some algorithm that attempts to learn something. So these words do not indicate anything particular about the project. Words like these are considered “content-less” in this context and were subsequently added to the stop–words list. As the pre-processing was done feature vectors were created for every project paper extracted and stored for the further usage as explained in Section III.

## Analysis

After the processing was done and all the techniques as discussed in Section III were implemented a user was presented with two different set of results: User was presented the codes of Data Structures written in Java and was recommended a set of projects based on the different area of interest of the user. This analysis was not computed but only done manually and has been talked in section V.

# techniques and implementation

Multiple Techniques were applied in serial on the data-set and recommendations were done. These techniques involved calculation of weights on words, creation of feature vectors and mapping of vectors based on the similarity.

In these techniques, **Item** would refer to content whose attributes are used in the recommender models. These could be movies, documents, book etc. **Attribute** refers to the characteristic of an item. A movie tag, words in a document are examples. So, item in the given context are the project papers and words in the document are the attributes.

## Content – Based Filtering

Content-based filtering, also referred to as cognitive filtering, recommends items based on a comparison between the content of the items and a user profile. The content of each item is represented as a set of descriptors or terms, typically the words that occur in a document. The user profile is represented with the same terms and built up by analyzing the content of items which have been seen by the user.. Concepts of Term Frequency (TF) and Inverse Document Frequency (IDF) are used for a content based recommender. Tf-idf is a numerical statistic that is intended to reflect how important a word is to a document in a collection or a corpus.

TF is simply the frequency of a word in a document. IDF is the inverse of the document frequency among the whole corpus of documents. IDF is incorporated to diminish the weight of terms that occur very frequently in the document set and increases the weight of terms that occur rarely. Logarithmic way is used to calculate the weights so as to dampen the effect of high-frequency words.

(3.1)

Using the above notation for calculation of weights values becomes more comparable as opposed to the original raw term frequency. Fig3.1 depicts a sample of calculated TF values.

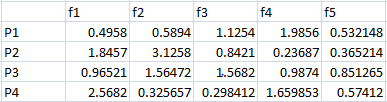


Fig 3.1 A set of data containing dampened TF values for 4 projects

Inverse Document Frequency (IDF) is calculated by taking the logarithmic inverse of the document frequency among the whole corpus of documents.

(3.2)

Fig3.2 depicts a sample of calculated IDF values.

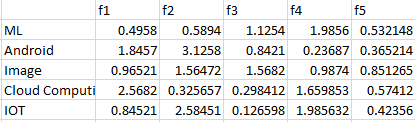


Fig 3.2 A set of data containing idf values for different domains.

Length of these project vectors, user vectors and domain vectors is calculated as the square root of the sum of the squared values. After the above processes, each term vector is normalized as by dividing the vector by the document vector length to get the normalized length. For our work, every item i.e a project paper have a unique feature vector (generated with the help of tf-idf), each containing 10 attributes.

## Similarity

We often want to compare two feature vectors, to measure how similar (or how different) they are. We hope that similar patterns will behave in a similar way. For example if we are performing handwriting recognition, a low distance between digit feature vectors (derived from images) might indicate that they should be given the same label. If we are building a recommender system for an online shop, similar user feature vectors (derived from their purchasing or browsing histories) might indicate users with similar tastes. The distance between two items depends on both the representation used by the feature vectors and on the distance measure used. There are 3 different ways to measure similarity.

* **Euclidean Distance :** The basis of many measures of similarity and dissimilarity is Euclidean distance. The distance between vectors X and Y is defined as follows:

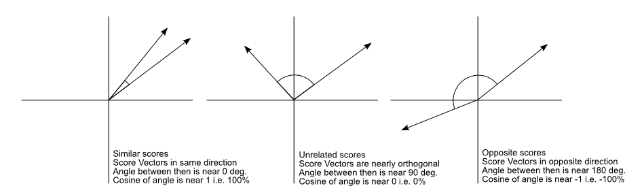
(3.3)

In other words, Euclidean distance is the square root of the sum of squared differences between corresponding elements of the two vectors. Note that the formula treats the values of X and Y seriously: no adjustment is made for differences in scale. Euclidean distance is only appropriate for data measured on the same scale.

* **Cosine Similarity :** The cosine similarity between two vectors (or two documents on the Vector Space) is a measure that calculates the cosine of the angle between them. This metric is a measurement of orientation and not magnitude, it can be seen as a comparison between documents on a normalized space because we’re not taking into the consideration only the magnitude of each word count (tf-idf) of each document, but the angle between the documents.

(3.4)

And that is it, this is the cosine similarity formula. Cosine Similarity will generate a metric that says how related are two documents by looking at the angle instead of magnitude, like in the examples below:



3.3(a) 3.3(b) 3.3 (c)

The Cosine Similarity values for different documents, 1 (same direction), 0 (90 deg.), -1 (opposite directions).

* **Jaccard Similarity :** The Jaccard similarity measures similarity between finite sample sets, and is defined as the cardinality of the intersection of sets divided by the cardinality of the union of the sample sets. Suppose you want to find Jaccard similarity between two sets A and B it is the ratio of cardinality of A ∩ B and A ∪ B.

(3.5)

Of all the above different measures of similarity, Euclidean Distance measure was used to calculate the similarity between the feature vectors with the threshold value of 0.5.

## Clustering

Cluster Analysis or Clustering is the task of grouping a set of objects in such a way that objects in the same group, known as a cluster are more familiar to each other than to those in other groups. Cluster analysis is a iterative process and not an automatic task. Appropriate clustering algorithm and parameter setting like the distance function etc. depend on the individual data-set and intended use of the results. In general, the work done defines two types of clustering techniques: Hierarchical Agglomerative Clustering and k-means clustering.

K-means clustering being a method of vector quantization, aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype for the cluster. Some pros and cons for k-means clustering are:

* Time Complexity: Takes linear time in the number of objects i.e. O (n) where n is the number of data objects, whereas the Hierarchical Clustering is quadratic i.e. O (n^2).
* Shaping of clusters: Favors hyper-spherical clusters or globular. Non-spherical clusters are not advised to work with this clustering.
* Number of Clusters: Needs the number of clusters to be defined whereas the hierarchical clustering doesn’t require number being specified.
* Random initial clustering generally results in inconsistent final clusters as the final results would always vary.

Based on our work, we chose hierarchical clustering namely, HAC (Hierarchical Agglomerative Clustering) as work concentrated more on non-spherical clusters and not on the time complexity.

HAC starts by initializing each sample in its own cluster. On every iteration process, clusters that are most similar are merged together. This is a “bottom-up” approach: each observation starts in its own cluster and pair of clusters are merged as one moves up the hierarchy. Upon successive iterations, threshold for merging the clusters is relaxed. So, more similar clusters are merged first, less similar clusters are merged later and dissimilar clusters are left unmerged. Similarity of the clusters is found using the Euclidean Distance given by:

(3.6)

where xi and yi are the 2 feature vectors. If the result comes out to be greater than 0.5, the vectors are merged together to form a cluster.

# RESULTS & DISCUSSION

The result of our project was to present a platform where a student can prepare for his placements and also can get recommendation as to which project he can choose from various domains mentioned in our overall work namely: Machine Learning, Internet of Things, Big–Data, Cloud Computing, Android Development, Image Processing and Computer Security, to help build him a strong resume.

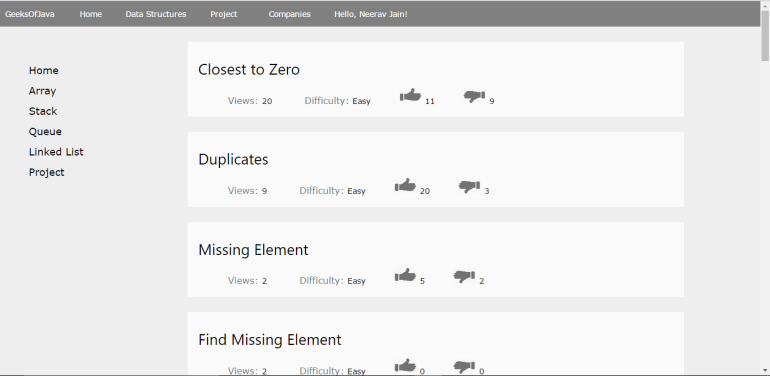


Fig 4.1 The UI of the platform



Fig 4.2 A sample of code in Java

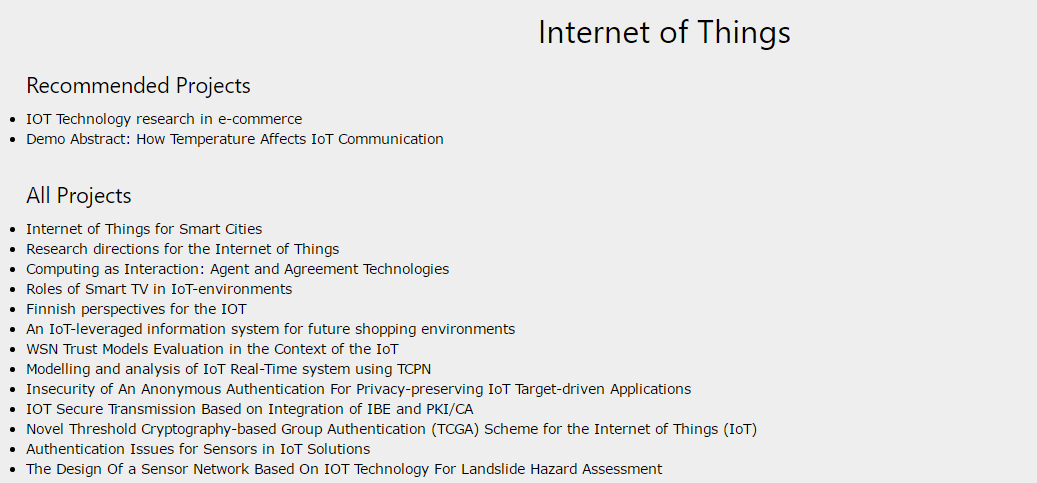


Fig 4.3 A sample of how the projects are recommended

Recommendations were provided to a set of students. Measurement of accuracy was a tougher task as recommendations could be useful and not useful too. The clustering algorithm presented here is able to discover some general patterns in past projects. However, the clusters are not very precise because they are based only on word frequency. The biggest factor that could improve this system is to use better natural language processing (NLP). The simple word frequency approach used here loses information that could be found in sequences of words (N-grams). Named entity recognition (NER) could be used to identify topic keywords and eliminate “contentless” words. Another limitation of the clustering algorithm used was the time complexity as it took a lot of time in separating so many project papers in different clusters.

Another problem encountered in this work is that project topics are becoming more broad as well. For example, robot vision combines robotics and image classiﬁcation. Trading stock based on Twitter messages combines ﬁnance and text analysis. These kinds of projects do not ﬁt neatly into the common topics and introduce a lot of ambiguity in the results.

# V. CONCLUSION AND FUTURE WORKS

This paper provides a common platform for the students to prepare for their placements. It performed unsupervised clustering on the projects. This revealed several common topic areas and as well as some unique projects. Our work confined to be recommending project papers which were domain-specific.

Future work would be to concentrate upon recommending the user with sub-domain specific projects also. We would also like to extend our work in recommending appropriate courses to opt for in the future. And also we would like to have a Q&A section where users can post doubts and can get answers by other peers.

# ACKNOWLEDGMENT

We would like to thank Dr. Shylaja S.S. Professor & Head of Department ISE, PES Institute of Technology for her immense encouragement and support throughout the project. We would extend our gratitude to our guide Prof. Vathsala M K who constantly pushed us forward and helped us alot. We would also like to thank our Information Science Dept. for their kind support.

# REFERENCES

[1] Lops, Pasquale, Marco De Gemmis, and Giovanni Semeraro. "Content-based recommender systems: State of the art and trends." *Recommender systems handbook*. Springer US, 2011. 73-105.

[2] Poon, Andrew. "What Project Should I Choose?."

[3] Resnick, Paul, and Hal R. Varian. "Recommender systems."*Communications of the ACM* 40.3 (1997): 56-58.

[4] Hosken, Benjamin E. "Automated content and collaboration-based system and methods for determining and providing content recommendations." U.S. Patent No. 6,438,579. 20 Aug. 2002.

[5] geeksforgeeks.org