Grenze International Journal of Engineering and Technology, June Issue

Job Profile Upskilling Through Content-based Course Recommendation System using Hadoop MapReduce

Mayank Parida[1], Vikramaditya Singh Saxena[2] and Dr. P Supraja[3]
[1-3]SRM Institute of Science and Technology, Kattankulathur, Chennai
Email: mp8699@srmist.edu.in, vs6735@srmist.edu.in, suprajap@srmist.edu.in

***Abstract*— As technology evolves, job roles become more demanding. Job profile upskilling using courses and certifications is imperative for an individual to be up to date about all-in-demand skills.**
Courses related to job profiles help individuals perform better at their workplace as they work more efficiently and optimally while utilising the least necessary resources.
Much popular e-commerce and over-the-top (OTT) apps now have recommendation algorithms. Typically, recommender systems will either present a user with a list of items from which they may choose or they will make a prediction as to the degree to which the user will like each item on that list. Collaborative filtering and content-based filtering are two popular methods for making suggestions. Hybrid recommendation systems may be created by combining these two methods, taking into account user ratings and item features when making recommendations. Features of the small data set may be analysed using the currently available data analysis software. Still, a big data analysis platform like Hadoop is utilised when dealing with large datasets found in the course and employment profiles. Hadoop is a platform for processing huge data collections in a decentralised manner. To speed up the process of assessing an item's characteristics (keywords of course descriptions and skill criteria for a job profile), Hadoop employs the MapReduce paradigm to execute distributed processing across clusters of computers.Job Profile Upskilling Through Content-Based Course Recommendation System Using Hadoop MapReduce allows us to predict the most relevant courses according to the job profile preference of the user.
Compared to other current recommendation systems, the suggested approach improves reliability and fault tolerance because of its use of course ratings to predict relevance with respect to attributes in job profiles. The suggested approach outperforms state-of-the-art recommendation engines, according to experimental data.

I. INTRODUCTION

Many companies need more skilled workers. Business owners in all sectors are re-evaluating their approaches to employee education because of the pervasive skill gap. Therefore, most modern employee training and development approaches focus on helping workers acquire new skills and improve existing ones. In today's technologically advanced world, a company's competitive edge can be significantly enhanced by investing in its people through upskilling or training and development. Companies, according to experts, need to invest in employee training to mitigate the effects of COVID-19. Upskilling is the process a candidate uses to increase their knowledge, beat out the competition, and advance in their present career path. It equips candidates with new skills to stay relevant and efficient in the ever-evolving business environment. After graduation, the

Grenze ID: 01.GIJET.9.2.73

program’s many job-oriented courses maximize the candidates' potential for success in a competitive business environment. Therefore, many ambitious individuals acquire new skills to boost their performance and advance their careers. Candidates with few or no transferable abilities may need help keeping up with the ever-evolving corporate landscape. On the other hand, acquiring up-to-date knowledge and abilities might assist in safeguarding one's professional future against disruptions and uncertainty. Many working professionals need more clarification and suggestions on how to elevate their job profile and prepare and commit to a different job profile. Many students find it challenging to choose which courses to opt for while searching for an entry-level job profile considering different job profiles have different requirements. The objective is to design a project that recommends top courses that need to be opted for by the student or working professional or working aspirant to suit fit for their current dream job profile. The objective is not just to recommend but also to build a project that makes this recommendation system process faster and more efficient yet still provides optimal results.

Recommender systems identify recommendations for individual users based on past purchases and searches, and on other users’ behavior. Recommender systems personalize each user's experience by surfacing material that is especially pertinent to their observed interests, as opposed to giving each user a generic experience. Sequential recommendation system has performance issues and drawbacks. The sequential algorithm takes more time with the increasing number of users and items. In sequential recommendation system tasks are not used for distributed processing, the choice of algorithm and their performances varies in sequential domain with respect to distributed environment. If the recommendation system implemented correctly can effective and improve the performance of system. We use a Hadoop distributed map reduce based recommendation system using Hadoop distributed cache and collaborative filtering technique. MapReduce framework provides a feature to cache read-only files such as text files, zip files, jar files etc. and create a local copy where MapReduce job is running. Each Data node gets a copy of the file which is sent through Distributed Cache. Performance of the system varies depending upon type of algorithms used as part of the recommendation system, and no of nodes used in the cluster environment.

The paper is organized as follows: II. Literature Survey, III. Hadoop Framework IV. System Architecture, V. Implementation, VI. Results and VII. Conclusion.

II. Literature Survey

Boban Vesin et al. (2012) developed a recommendation system termed PROTUS (PRogramming TUtoring System) that recommended courses to the students. Normally, courses are suggested to students based on their age and area of study, but in this system, ideas from semantic web technology are used. Navigation patterns are obtained from the history of the student, and from that pattern, future recommendations are made [1]. Konstantin Shvachko et al. (2010) conducted a study on the Hadoop distributed File System. The study stated that by distributing the storage and computation across the machines of a cluster, the computational time could be reduced for analysing big data compared to single-node processing [2]. Brian McFee et al. (2012) created a music recommendation system using content similarity learning. It initially used a content-based similarity method, and then a collaborative similarity method was imposed on the results. It avoided the cold start problem and the overhead of The query-to-answer technique [3].Raphaël Morsomme et al. (2019) developed a recommendation system for liberal arts courses. The system recommends courses whose content best matches the student’s academic interests and issues warnings for courses that are too advanced, given the student’s academic background. It suggests suitable preparatory courses in the latter case [4].To help authors decide where they should submit their manuscripts, Donghui Wang et al.(2018) present the Content-based Journals & Conferences Computer science recommender system, Based on a manuscript's abstract, this system suggests appropriate conferences or journals in a priority sequence [5]. David Estrela et al. (2017) developed a collaborative filtering recommendation system that helps students perceive relevant information to those courses and decide which courses to take. This is done based on the user profile and their similarity to other users [6]. Raghad Obeidat et al. (2019) created a collaborative recommendation engine that suggests online courses to students based on similarities in their prior course experience. The suggested method uses data mining techniques to identify trends among the courses[7]. Simon Philip et al. (2014) suggested using a content-based strategy to enable users to navigate through enormous collections of digital objects in digital libraries and discover the relevant ones, By using TF-IDF and cosine similarity to assess a research paper's relevance or resemblance to the user's inquiry[8]. Mooney, R.J., & Roy, L. (1999) utilize a content-based system instead of collaborative filtering methods, which use information about an item to give a recommendation. Utilizing this, they provide a Content-based book recommendation system using learning for text categorization[9].Srs Reddy

et al.(2018) In this paper, the recommendation system has been built on the type of genres that the user might prefer to watch. To do this, a content-based filtering strategy based on category correlation has been used. [10].

III. HADOOP FRAMEWORK

MapReduce Model - The MAP/REDUCE is a frame work which is capable of distributed processing of many terabytes of data on thousands of computing nodes in a HADOOP cluster. Map-reduce has primarily two tasks, which are executed in phase wise. In first phase map task is executed and in second phase reduce task is executed. Below is the figure describing map reduce operations. Input the MAP muddle is a set of data and MAP() functions partitioned the data set into <key, value> pairs. These key/value pairs are now send to the REDUCE module and REDUCE() function combined and aggregates, perform SORT operation based on key and finally send to output node to generate output.
Hadoop Distributed Cache - Distributed Cache is a feature provided by the Map-Reduce framework to cache files (text, archives, jars etc.) required by applications. Applications specify the files, via URLs (hdfs:// or http://) to be cached. Before the slave node performs any tasks for the job, the framework will copy the required files onto that node. The files are copied only once per job. The cache files' change timestamps are tracked by Distributed Cache. Clearly cache files 21 should not be modified by the application or externally while the job is executing. Using Distributed Cache feature in Hadoop map-reduce framework will reduce the latency in Hadoop network as the Hadoop framework uses distributed storage and process huge datasets using HDFS and MapReduce. As a result, when processing large amounts of data, the user writes the programme to retrieve the actual data from the storage. This lowers latency because instead of reading each file from the HDFS one at a time, the actual data can be copied and sent to all data nodes at once.

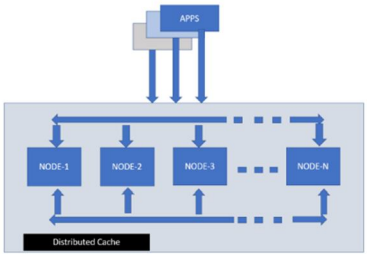

Fig. 3.1 MapReduce Task

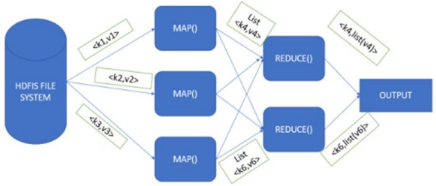

Fig. 3.2 Distributed Cache

Today's world, World Wide Web adapted as an ideal platform for developing applications which uses large volume of data. Data intensive application like Netflix, Amazon, Data Mining, webmail, and online retail application needs to access large data sets in the range from few GB (Giga Bytes) to several TB (Tera Bytes) or even PB (Penta bytes), and processing of huge volume of data set is done in parallel fashions. Google MAP/REDUCE model parallels the data processing on high performance computing environment. Google MAP/REDUCE is a scalable architecture and partitions the large datasets into small tasks and these tasks run on multiple nodes in a large cluster. Accessing large dataset from HDFS file system every time for the computation purpose will degrade the performances due to I/O latency. Hadoop distributed cache mechanism allows data to be stored in cache to avoid reduce the I/O latency and improve the performance of the system. Content Based Filtering algorithms, Cosine are widely used in commercial recommendation systems. Scalability is an issue in Content Based Filtering and is a big problem i.e., when a very large dataset, the computation cost of Content Based Filtering would be very high. There are several scalability architectures available to resolve the problems including cloud solution (SAAS), but we choose Hadoop MAP/REDUCE DISTRIBUTED CAHE base frame works because it is an opensource platform. The Hadoop Map-Reduce framework solves the scalability issue for systems dealing with large data sets. There are multiple algorithms available in CF mechanism for the recommendation system. But performance and execution time of these algorithms in the scalable architecture needs to be measured. We have chosen two such algorithms in HADOOP MAP/REDUCE DISTRIBUTED cache environment for the comparative study purpose in a 3 nodes-based cluster environment.

IV. SYSTEM ARCHITECTURE

The overall architecture of the project begins with the dataset i.e. the csv files being pre-processed to conduct feature extraction in the format of txt file as the Hadoop framework takes txt file as parameter. The Hadoop 3

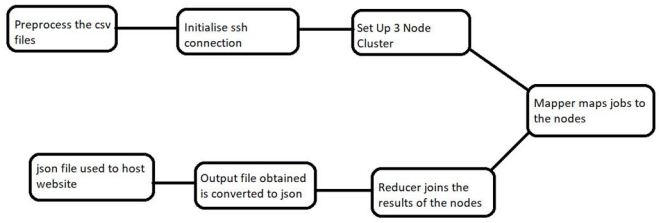

Fig. 4.1 Architecture Diagram

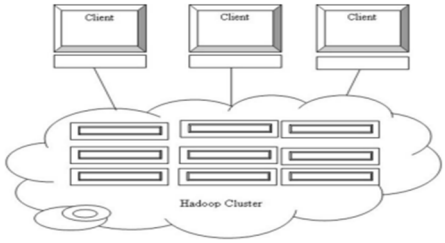

Fig. 4.2 Hadoop 3 Node Cluster

node cluster is set up with password less SSH connection between the nodes. The mapper maps the jobs to the 3 nodes present. The 3 individual nodes work on the jobs provided to them and pass their results to the reducer. The reducer works on compiling the result of the nodes and delivering the actual final result which is of the format txt. This txt file is then parsed to obtain a JSON file. This JSON file is used while designing the website which helps map the courses against the job profiles mentioned.

Feature Extraction - Python would be used to extract and create useful features for the recommendation system. The unwanted columns are discarded, and a few useful columns are included.

The following text preprocessing steps could be followed:

1. Converting everything to lowercase, 2. Removing stop words, 3. Lemmatizing the words.

Hadoop 3 Node Cluster - The main reason to use Hadoop 3 Node Cluster is to increase the performance of the recommendation system. This is achieved by creating three nodes and making them work towards achieving a united goal. Hadoop is an Apache open-source project that enables the development of applications for parallel processing on big data sets dispersed across networked nodes. It comprises the Hadoop Distributed File System (HDFS) that handles scalability and redundancy of data across nodes, and Hadoop YARN, a framework for job scheduling that executes data processing tasks on all nodes. A master node maintains knowledge about the distributed file system, like the inode table on an ext3 filesystem, and schedules resource allocation. This position will be played by Node-master, who will also host two daemons.:

1. The NameNode manages the distributed file system and knows where stored data blocks inside the cluster are.
2. The Resource Manager manages the YARN jobs and schedules and executes processes on worker nodes.

Worker nodes serve as data storage and processing resources for tasks. There will be two daemons hosted by them, Node1 and Node2.:

1. The DataNode manages the physical data stored on the node; it's named, NameNode.
2. The Node Manager manages the execution of tasks on the node.

Hadoop Mapper Reducer - Hadoop Map Reduce is a process that would be followed in the Hadoop 3 Node Cluster where the job/process that needs to be performed is split between these 3 clusters. Each cluster performs the task expected from them. This is the mapper phase. Then comes the reducer phase, where the clusters' results are compiled, and the actual output is obtained for the entire process. This helps us increase performance and give us faster and optimal recommendations of courses for each job profile. MapReduce program executes in three stages: map stage, shuffle stage, and reduce stage. Map stage − The map or mapper's job is to process the input data. Generally, the input data is in the form of a file or directory and is stored in the Hadoop file system (HDFS). The input file is passed to the mapper function line by line. The mapper processes the data and maps several small jobs of data. Reduce stage − This stage combines the Shuffle stage and the Reduce stage. The Reducer's job is to process the data that is received from the mapper. After processing, it produces a new set of outputs, which will be stored in the HDFS. For each job, we calculate cosine similarity with each course based on the term Frequencies calculated in the feature extraction step. Then we multiply the cosine similarities with the course rating so that we make sure to recommend the most suitable highly rated course. The product of cosine similarity and rating is referred to as the similarity that the mapper outputs.

V. IMPLEMENTATION

A. Data Pre-processing

Initially, two datasets will be present, one is the course-related CSV file and the job profile-related CSV file. Here the files are processed such that their features are extracted accurately. The unwanted columns are removed from the CSV files. Stopped words are exempted and followed by the words being lemmatized. The resultant

files undergo feature extraction, and the results are then stored in 2 separate txt files as Hadoop reads only txt files.

B. Hadoop 3 Node Cluster Setup

A 3-node cluster is needed to be set up. This will be achieved using virtual machines. A hypervisor machine of 16 GB is required to set up an efficient three-node cluster. Three virtual machines will be created using oracle virtual box, which has a preferable OS of Unix provided with sufficient memory. The three virtual machines will then have an SSH connection to be set up among them, which would be passwordless. The passwordless connection can be established by generating a public security key that would be present in each of the virtual machines. Finally, a Hadoop Distributed File System would be set up to activate the 3-node cluster.

C. Hadoop MapReduce Integration

The Hadoop MapReduce integration will be done using java. A distributed cache system would be set up such that the nodes do not have to access the course data every time, as it is a constant requirement in all the nodes. The mapp.er would separate the job profiles among the nodes and process the data to provide results per node. The results from each node are then compiled and processed by the reducer to produce the outcome of the entire process.

D. Website Integration

The final output of the Hadoop MapReduce integration produces an output file which is of the format .txt. It isn't feasible to process a .txt file and integrate it with a web application. The output file is hence parsed into JSON using Python code. The JSON file is now hosted as an API and a get request can be made to get the details of all the job profiles and courses. Now that we have the JSON data, we can easily avail which courses to be followed in order to stride towards a particular job profile. The web application is made using React, html, CSS, and firebase.

VI. RESULTS

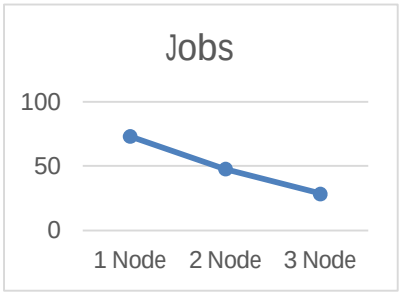

Fig.5.1 Time taken by different cluster

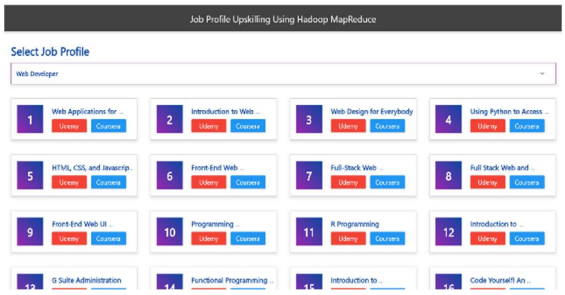

Fig. 5.2 Connected Output to Website

The above chart shows the different execution times for different numbers of clusters and no cluster as well. The above chart clearly shows how the 3 Node Cluster consumes almost 3 times the duration of that of the Single Node Cluster. The 2 Node Cluster performs better than the single node cluster, however falls short of the duration taken by the 3 node cluster. The conclusion drawn here is more the number of clusters, the faster the execution.

VII. CONCLUSION

In this paper, we have achieved many objectives. We not only have found a solution to help working professionals as well as students to pursue their dream next job profiles by doing recommended courses all in one place but also have saved time in all the recommendation processing using Hadoop MapReduce which would have otherwise taken ample amount of time which is not an efficient system. A beautiful UI based Web Application was connected to the output of the recommendation system such that the user can explore the product with ease.

The limitations that are faced while implementing the project is that the input file must be pre-processed in the required format. The evaluation does not include the processing time. Only the execution time and size of the data are considered.

REFERENCES

[1] Vesin, Boban & Ivanovic, Mirjana & Milicevic, Aleksandra & Budimac, Zoran. (2012). Protus 2.0: Ontology-based semantic recommendation in the programming tutoring system. Expert Systems with Applications. 39. 12229–12246. 10.1016/j.eswa.2012.04.052.

[2] Shvachko, Konstantin & Kuang, Hairong & Radia, Sanjay & Chansler, Rob. (2010). The Hadoop Distributed File System. Mass Storage Systems and Technologies (MSST), 2010 IEEE 26th Symposium on 2010. 26. 10.1109/MSST.2010.5496972.

[3] McFee, B., Barrington, L., & Lanckriet, G.R. (2011). Learning Content Similarity for Music Recommendation. IEEE Transactions on Audio, Speech, and Language Processing, 20, 2207-2218

[4] Morsomme, R., & Alferez, S.V. (2019). Content-based Course Recommender System for Liberal Arts Education. Educational Data Mining.R. Nicole, “Title of paper with only first word capitalized”, J. Name Stand. Abbrev., in press.

[5] Donghui Wang, Yanchun Liang, Dong Xu, Xiaoyue Feng, Renchu Guan, A content-based recommender system for computer science publications, Knowledge-Based Systems, Volume 157, 2018, Pages 1-9, ISSN 0950-7051.

[6] Estrela, David & Batista, Sergio & Martinho, Diogo & Marreiros, Goreti. (2017). A Recommendation System for Online Courses. 195-204. 10.1007/978-3-319-56535-4_20.

[7] R. Obeidat, R. Duwairi and A. Al-Aiad, "A Collaborative Recommendation System for Online Courses Recommendations," 2019 International Conference on Deep Learning and Machine Learning in Emerging Applications (Deep-ML), 2019, pp. 49-54, DOI: 10.1109/Deep-ML.2019.00018.

[8] Philip, Simon & Shola, Peter & Abari, Ovye. (2014). Application of Content-Based Approach in Research Paper Recommendation System for a Digital Library. International Journal of Advanced Computer Science and Applications. 5. 10.14569/IJACSA.2014.051006.

[9] Mooney, R.J., & Roy, L. (1999). Content-based book recommending using learning for text categorisation. Digital library.

[10] Reddy, Srs & Nalluri, Sravani & Kunisetti, Subramanyam & Ashok, S & Venkatesh, B. (2018). Content-Based Movie Recommendation System Using Genre Correlation.