INTERNATIONAL JOUNAL OF: SCIENCE RESEARCH AND TECHNOLOGY

TIMBOU-AFRICA ACADEMIC PUBLICATIONS
INTERNATIONAL JOURNAL, AUGUST, 2021 EDITIONS
VOL. 6 NO. 9 ISSN: 2623-7861

# DESIGN AND REALIZATION OF TOP-N HOTEL RECOMMENDER SYSTEM BASED ON USER REVIEWS KNOWLEDGE GRAPH TECHNIQUE.

***ADERONMU JULIUS; **ADENIYI .D. ADEDAYO; & **MORA HAFSATU**

**Department of Information and Communication Technology, Osun State University, Osogbo, Nigeria.*
***Department of Computer Science, College of Science and Technology, Kaduna Polytechnic, T/Wada. Kaduna, Nigeria.*

**ABSTRACT**

*Travelling to new places for leisure, academic or business has becomes an important parts of the civilized man. However, getting the right information about places of interest at the right time as it relate to hotel accommodation has become a critical task for the online hotel website users (travelers). To overcome this challenge, this work presents the design and realization of a hotel*

## INTRODUCTION

Decision making is a critical task faced by human being on a daily basis, decision as to where to go, what to buy, what book to read, the song to listen to or movies to watch etc. The choice are many but time to explore the choices are limited. Human beings love travelling to new places for leisure, academics or for business etc. However, getting the right information at the right time about the places of interest especially as it relates to hotel accommodation has become a critical task specifically for online hotel website users. To alleviate this problem, this work presents a simple but promising hybride graph based recommender system based on user's implicit feedback. The study presents the design and realization of Top-N-Hotel recommender system based on user's

review, using knowledge graph technique. The stud y aimed at helping travelers in making informed choices about where to go and sometimes when to or not to go. The contributions of this work are as follows:

First, the paper presents the design and realization of a novel Hotel recommender system. Presently, most recommender systems make use of the user-supplied ratings to infer user preference. However, the ratings may not be a true representation of the position of a user because they are limited to some predefined options. What does better in this sense is the review. With the review option, users are given the freedom to express themselves rather than select from a defined rating. This paper places emphasis on user reviews, and provide a

*recommender system based on user's review knowledge graph technique, through implicit feedback of users. The present system is capable of overcoming the problem of rating values misrepresentation through the use of impact user's feedback. The review text passes through sentiment analysis during preprocessing, then user's review polarity is scored. The result of the polarity on the comment are analyzed and used to recommend hotel to the potential user/Visitors/travelers at a given time. The implementation of the designed system is achieved through the use of an in-house developed application using Python programming language with Neo4J an NoSQL database management software at the back-end. Performance comparison between the present polarity on comment technique was carried out with the supplied rating technique, the result shows that the use of review text in the recommendation outperformed the use of ratings. Therefore, the present recommender system is capable of providing useful, consistent, faster, accurate and efficient hotel location recommendation to the user/traveler at any time, online and on real time basis.*

***Keyword:*** *Knowledge Graph, Recommender System, Sentiment Analysis, Top-N, Polarity, Feed-Back*



TIJSRAT
ISSN: 2623-7861

comprehensive and truer position of the user in the survey of hotels and it attempts to use the valuable information in reviews to solve the disparity between the ratings and review. We began with collection of data. To implement the top-N hotel recommender system, we need a data that not only captures the rating scale but includes the text review representing the comments given by users about a particular hotel. This form of unstructured data was found on the Datafiniti's Business Database. We proceeded by looking at the functionality and limitations of existing recommender system. Next we look at the natural language processing i.e sentiment analysis processing technique suitable to derive a more reliable and optimal scale of positive and negative review using a python library called Textblob which is based on Naïve Bayes algorithm.
Second, we present the use of knowledge graph database which is based on graph model that function on the concept of vertices and edges for real time storage, manipulation and retrieval of complex less structured and highly connected data.
Third, we present the construction of a specific hotel recommender system using an experimental application developed in-house with python programming languages at the back end and Neo4J an NoSQL database management system at the front-end. We passed the resulting data of the sentiment analysis into Neo4j, a graph based database application used as our recommender machine. This is queried using the Cypher queries, a Neo4j graph query language to producing a recommendation. To achieve this goal the developed system instead of using the fixed ratings scale of the users, uses the review text which captures the true expression and sentiment of the user. The result shows that our scale of between -0.1 to +1.0 is a better representation of the sentiment of the user than the rating scale of 1 to 5 hence, recommends better. The system, as a result of its entity-relationship design, gives the user the next item, and then the next and so on thereby improving decision making. The present approach has the capability of addressing the identified issues and suggest a direct, simple yet more accurate recommendation.
Finally, We present our experimental results through performance comparison of the present polarity on comment technique with the

supplied rating technique in order to justify the rationale behind the selection of our knowledge based user's review technique. The result of our experiment shows a better performance of our technique over the baseline technique. Therefore, the present Knowledge graph based recommender system is capable of providing transparent, simple, straightforward recommendation to the users consistently.

## RELATED WORK

Several research work have been carried out on recommender system both for research purpose and for applications. The system have been used for item sales recommendation, hotel recommendation, and movie suggestions and so on. Some of this use-case and research papers are reviewed and critiqued in this section. This is done to give a proper understanding of the model proposed in this work and give details of how it intends to address the issues identified. The review is organized according to the following sub headings:

### Overview of Recommender systems

Yu, Ren, Sun, Gu, Sturt, Khandelwal, Norick and Han (2014), proposed a recommendation method based on meta-path which learned paths consisting of node types in a graph. This is done by studying the personalized entity recommendation with implicit user feedback in heterogeneous information networks and diffuse user preferences along different meta-paths to produce latent features for users and items and finally generates personalized recommendation models for different users. Similarly, Ostuni, Di Noia, Di Sciascio and Mirizzi (2013) proposed a system called SPRank (Semantic Path-based Ranking) an hybrid algorithm that computes top-N item recommendation in what is called the learning-to-rank framework where N is the number or items or things. The system proposed by Verma, (2015) features a recommender system that has a knowledge base which comprises entities and relationship connecting them in a manner that each entity is related to a number of other entities via the existing relationship which could be generic. As users click a relationship during system interaction when looking through recommended entities, the information about the

clicked relationships are stored. These user activities (like the count of clicks) is also stored and then it grows overtime and then used to depicts the preferences of the user. *RestRec* is a complete web-based, context-aware personalized hybrid recommendation system for restaurants developed by Pettersen and Tvete (2016), which allows users to sign up and log in and authenticated thereby using their personal detail to create suggestions that are more accurate in future log in. *RestRec* presents a system called the Restaurant data extractor which extracts restaurant data useful for the recommendation. In addition, it has the recommendation engine which uses both the information of the user and the restaurant data to produce recommendation. This part is tagged the heart of the recommendation system. Another key component of RestRec is the Profile learner which keeps track of the user profile and gives the option of possible update of profile by the user. In the system, a new user will have to register and an existing user authenticated and the recommendations will be based on the user profile as supplied. Finally, the user is allowed to provide a feedback.

*RestRec* aims to solve the cold-start problem and also use a number of technology to achieve this and more. For instance, the content-based filtering approach meshed with collaborative filtering is used which calculates the set of restaurants that are most similar to the profile supplied by the user and also recommend restaurant that are most popular among users.

Unlike Pettersen and Tvete, (2016), rather than taking input from users via some designed forms for their profile Dalal, (2016) use the user existing profile from the social media account Facebook to understand their likes, where they have been etc. This is done using the graph API provided by Facebook. User travel experience gotten from the photo information of users which includes time, location, latitude and longitude and the name of the point of interest plus name, email and current location are all used to analyze the user. In this system, the information retrieved are stored and compared with other stored profile and recommendation is given based on profiles having a look-alike taste or interest as that of the target user using the collaborative filtering approach to suggest the destination to the user. This system is

implemented to be used on the Android device that retrieves user data and sends to the web service developed for processing of data and suggesting destination result which is then redirected back to the Android device. This system overall has 4 components including the data collection, Profile analysis, predicting destination and Android application.

(Akinyemi, Amoo, Abimbola, Awoyelu, and Olaniran (2015) proposed a hybrid model that combines content-based, collaborative and demographic filtering on the data for recommendations. The model uses the demographic data to solve the cold-start problem and formulation was done by replacing the ratings in the collaborative recommendation model for purchases. The recommendation is done by category based on the available data. For example, where demographic information is supplied, a demographic suggestion is generated. In the development of the system, web development tools was used which implies that it is a web based application.

A recommender system that focused on collective profiles by Antony, Barrios, Quintero, and Barranquilla, (2017) aims to solve the touristic preferences of a group of tourist and satisfying all group members. That is, the system aims to suggest places of interest to a collection of group members based on their collective profile using the hybrid recommender method. In this proposed system, each member likes, preferences, tourist personality and tourist experience is calculated with characterization instruments, the result of which is the individual touristic preference of the member. The profiles necessary for the collective calculation includes sub-profiles like cultural, bio-ecologic, adventure, urban and sports and also the attitude of members are also measured.

Different approaches have been used by different authors in developing recommender systems be it with a hybrid or conventional strategy. Many of the authors and work reviewed previously focused on the use of hybrid system based on some review ranking. Review ranking from the dataset used in this work was found to be inconsistent with the review text. Hence, in this work, review text was used instead of the rankings. There are several other similarities in the systems even though there are

notable differences in their approach. Similarities include taking care of the cold start problem, use of user profiling options etc. Vast majority of the reviewed work were based on several other approaches but in this project work, the proposed recommender system is based on exploiting the benefits of knowledge graph with the result of sentiment analysis done on the review text as input.

Potentially, graph based recommender systems are great options to illustrating learning based recommendation techniques. Unlike observed trends of recommender systems as seen in the reviewed work, graph based approach deals more on distance-like measures and connection or affiliation between the entities in the knowledge graph. Graph based methods have been proven to have the potential to deal with the cold start problem and to act as the basis of relatedness based recommendation techniques Filzmoser, (2016). In the presentation of the solution, the nodes of the graph represents the different entities exhibiting particular roles in the recommendation system while the edges of the graph are the various relations between the entities.

**Overview of Graph Data Model and Algorithms**

Regardless of the size of the dataset, the architecture of the graph data model allows us to query and compute complex operations on the relationship Cung and Jedidi, (2014). Graphs can be described as a class of data structures that are used in the modelling of relationships (edges) between entities (nodes). Both the edges and nodes may have properties representing core values that describe the entity they are attached to. The major benefit of a graph model is that it is easy to understand and makes it easy to express various scenario. The simplicity of the graph data model does not take anything away from its ability to perform complex task on voluminous dataset. Graph algorithms are applied in the computation metrics for graphs, nodes, or relationships and have proven to be useful in various fields and research areas, this includes but not limited to fields such as biology, transportation, recommendation or social networks and so on Cung and Jedidi, (2014).

After the review of related work as presented above, the contribution of this work has been identified. This proposed work contributes to the

design of a hybrid recommender system which calculates a set of top-N suitable hotels from a list of unstructured review text supplied by previous visitors to the hotels. To achieve this, the unstructured text is passed through text classifier and sentiment analysis which categorizes the review text between the range +1.0(for very positive) and -1.0 (for very negative) values of which is used in the knowledge graph. This approach aims to solve the problem of inconsistency between the ranking value and the exact feeling of a user.

## RESEARCH METHODOLOGY

This section explains in detail the system design and approach. First a description of the data collection and then the dataset used. The preprocessing of the dataset is also explained in detail. In addition, the approach used in preparing the data for the recommender system is explained in detail. Finally, details of the knowledge graph implementation is given. The implementation of the system is largely performed in Python and a graph database application known as Neo4j.

### Data Collection

The dataset used in this project experiment is the "Datafiniti_Hotel_Reviews". This dataset is a list of about 2,000 hotels and 10,000 reviews from Datafiniti's Business Database updated between January 2018 and September 2018. Each hotel listing includes a variation of the term Hotels within the Category field. All fields within this dataset have been flattened, with some omitted, to streamline the data analysis. This version is a sample of a large dataset. The dataset includes the location of the hotel, its name, its rating, and the review. It also includes the review text supplied by users of the facilities, title, username and more. It contains both positive and negative reviews. Of interest is the fact that some rating scale do not particularly match the review text and forms the basis for our choice to use the review text rather than the rating score.

The 'true sentiments' derived from the review text is scored as shown in figure 1, below:

- Negative review rating: -1 to -0.1.

- Positive review rating: 0.1 to 1.

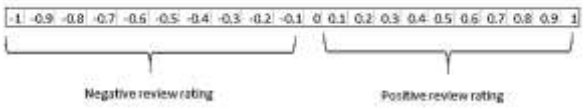


**Figure 1: Positive-Negative review rating scale**

**Dataset Preprocessing**

To get sufficient predictive power, machine learning methods must have a large set of data. A 'preprocess_dataset.py' script is used for this purpose. The 'preprocess_dataset.py' is responsible for taking care of preprocessing the raw dataset. The unedited version of the dataset contained "invalid" hotel reviews, such as "No positive review" or "No negative opinion" - information that do not provide valuable insight when training the classifier. This script also separates the raw dataset into two different files: one for positive reviews and another one for negative reviews. The words in the review includes words that do not impact the classification of the text as positive or negative. These are words referred to as stop-words. Example of these words includes 'i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're", "you've", "you'll", "you'd", 'your', 'yours', 'yourself', 'she', "she's", 'her', 'hers', 'herself', 'it', "it's" and so on.

In addition, the valid words need to be tokenized. Tokenization is the splitting of sentences into its constituent words. This is important because the meaning of text generally depends on the relations of words in that text. The preprocessing procedure is shown in figure 2 below;

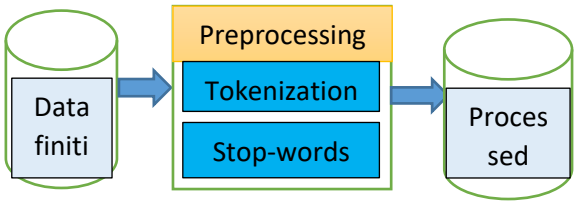


**Figure 2: Raw dataset and preprocessing procedure.**

In figure 2 above, the raw data is passed through initial preprocessing procedures to tokenize the sentences and to remove stop-words. This prepares the dataset for a sentiment analysis.

**Finding the User Sentiment**

The review text rather than numerical rating scales are very significant and are a rich source of information to the recommender engines. User's opinion are usually expressed in the review text and to access them for the use of the recommender engine, a sub-system was implemented to read its implicit features. The degree of positivity or negativity is automatically detected by the subsystem and then provides the recommender engine with some form of polarity score. The polarity score reflects the user's sentiments. In many computer science literature this procedure is referred to as the sentiment analysis or polarity detection where some natural language processing techniques are employed to extract, classify or portray subjective information from the textual data. The task of polarity detection of subjective human opinion is a challenging one. These opinions are usually expressed in complex ways particularly as some are expressed in some rhetorical modes like sarcasm and invariably makes polarity detection a difficult task. Machine learning applied alongside some natural language technique is employed to overcome this challenges. This is depicted as shown in figure 3.

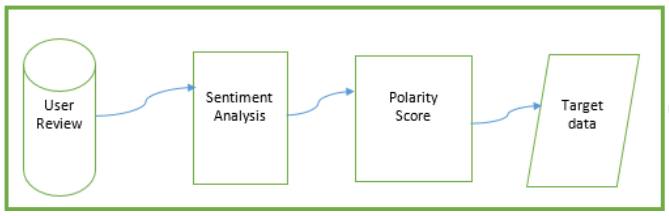


**Figure 3: Sentiment Analysis sub-system**

As was explained in the previous section, the dataset was passed through some NLP module for preprocessing. This is done to tokenize (breaking down streams of text into smallest meaningful components) the review text and to remove the stopwords (words which do not impact or alter the sentiment of a sentence). Thereafter, the preprocessed dataset is passed to the sentiment analysis module. In this module, the polarity of each review is retrieved. The polarity is the score of the degree of positivity or negativity of the review sentence of the user. It is a scale of numeric values ranging from -0.1 to +1.0. The score -0.1 being very negative sentiment or review and +1.0 being very positive

sentiment or review. Binary Naïve Bayes algorithm was used in the sentiment classification. This algorithm classifies text as positive or as negative based on the input supplied. Bayes' theorem is as stated in equation (1). P(*c*|*x*) (1)

In equation (1), P(*c*|*x*) is the probability of hypothesis c given the data *x*. This is referred to as the posterior probability. P(*x*|c) refers to probability of data *x* given that the hypothesis c was true. P(c) is the probability of hypothesis *x* being true (regardless of the data) and is referred to as the prior probability of c and P(*x*) is the probability of the data (regardless of the hypothesis).

A python package based on the Naïve Bayes algorithm was used to get the polarity (sentiment) of the users. It will be recalled that there are about 10,000 reviews of about 2,000 unique hotels, hence, a need to find a score which is a true feeling of users about a hotel.

The mean of several polarity of a hotel is derived. The mean is a better measure of central tendency which is better than just selecting the median.

$$\bar{x} = \frac{\sum x}{n} \qquad (2)$$

**Preparing the Data for Neo4j**

Before our data is ready to be used by the recommender system, it needs to be prepared to standard. A common format of data storage is in rows and columns on flat files. The type similar to what we have in spreadsheets. In a variety of importing and exporting of data to and from relational databases, this spreadsheet format is used, hence it is easy to recover current data this way. Another means of data import to Neo4j is the API. For the purpose of this project work, the Comma Separated Values (CSV) is used. The data from the sentiment analysis is processed to a Unicode Transformation Format- 8bit (UTF-8) saved with specific applicable columns including: Id, DateAdded, DateUpdated, Address, Categories, PrimaryCategories, City, Country, Keys, Latitude, Longitude, Name, PostalCode, Province, SourceURLs, Websites and Polarity.

Polarity, of all the columns captured above, is the new addition derived from the sentiment analysis phase and represents the mean of the


TIJSRAT
ISSN: 2623-7861

sentiments of users about a particular hotel. Saved as 'updated.csv', the dataset is prepared for use by the recommender machine.

**The Neo4j Architecture and the Graph Data Model**

There are certain changes to data that make graph database an important tool in data management and manipulation today. These changes include the increase in volume of data, complexity of the data, structure of the data and interconnectivity of the data. This drift in data properties gave rise to a new group of Database Management System (DBMS) known as the NoSql. One of such databases is the graph database.

Graph database is based on graph model which functions on the concept of vertices and edges. It allows real time storage, manipulation and retrieval of complex, less-structured and highly connected data Bachman, (2013). Neo4j is an open-source graph database which runs on Java Virtual Machine and like some graph DB's, store data as graph. It is a graph database that allows navigating enormous volumes of data easily. Unlike relational databases, data in Neo4j does not adhere to any pre-defined structure Bachman, (2013).

**The Recommender Engine**

Having processed the data for recommendation, the data is imported into the engine with its "LOAD CSV" command. Please note that the CSV must have certain characteristics before the load command to works. These characteristics include:

- Its encoding character is of the form UTF-8.
- The system for which it is run determines the end line termination. For example, On Unix OS it is \n and on Windows it is \r\n.
- "," is the field terminator by default.
- The field terminator character can be altered by applying the option FIELDTERMINATOR which is present in the LOAD CSV command.
- Strings that are quoted are permissible in the CSV file. These quotes are dropped when reading the data.

- Double quote (“”) is the character for quoting strings.

**Algorithm Listing**

The algorithm listing for the present Top-N Hotel recommender system is presented in the listing-1.

**Listing 1:** Algorithm listing for the Recommender system

```
1. Let K be the number of unknown stopwords
2. Begin
3. Let I = 1
4. // section to input and remove Stopwords
5. Do while not (End of Document(i))
                For j= 1 to k
                Retrieve available word from the database i.e. Text(j)
                review_text=str(review)
                ssplit = review_text.split()
                filtered_words = wordinssplit
                if(filtered_words =stopwords.words()
                {
                {
6. // Section to derive mean Polarity
7. I=1
8. blob = TextBlob(str(filtered_words))
9. polartites.append(blob.sentiment.polarity)
10. #data.polarity.loc[i]=blob.sentiment.polarity
11.   i+=1
12. data['polarity']=polartites}
13. Display “ Done Sementic Analysis  Finalizing”
14. unique_reviews_per_hotel=data.groupby('id').mean().reset_ind
    ex()['polarity']
15. backup=data.groupby('id').mean().reset_index()['polarity']
16. data.drop(['polarity'], axis=1)
17. data=data.drop_duplicates('id')
18. datane=data.copy()
19. datane=datane.reset_index()
20. datane=datane.drop(['index'], axis=1)
```



TIJSRAT
ISSN: 2623-7861

21. datane['polarity']=unique_reviews_per_hotel.values
22. End while
23. // Section to perform Recommendation based on Top-N Sentiment.
24. Rank the reviewed-text on scale between -0.1 to +1.0
25. Recommend the first Top-3 Sentiment ranking
26. End

**IMPLEMENTATON AND RESULT**

The design and implementation put together to express the theory of this work as explained in previous section is presented in this section. Two modules are involved in the design. First, the python module for the sentiments analysis for the polarity index and second, the Neo4j database module for the recommender engine.

**Design**

The design of the system is built around the Neo4j an NoSQL database. Neo4j is an open source database that can be download online. The application's inbuilt cypher graph query language is used to query the database presenting a great and interesting visualization of the data.

**Tools and Techniques**

Python programming language and Neo4j 1.1.17 made by Neo4j, Inc. are used in the implementation. In the python module, the dataset is preprocessed and polarity derived producing an output which is supplied as input to the database module. Like most noSql database, the design does not follow any specific module hence makes it a flexible and easy to use applications. However, a good practice is to design the database which solely depends on the common patterns of query.

**Neo4j Core API**

Before the implementation proper, it is important we explain the core API of Neo4j. It is an imperative Application Programming Interface (API) with the ability to provide graph modification and transversal. With the interface of the database comes the capability for node and relationship

creation and retrieval. Nodes and relationships, both implementing the Property Container interface, are represented as Node and Relationship, respectively. Providing access to relationships is The Node while the Relationship gives access to its type and both participating nodes. In the implementation, the Nodes are circular while the relationships are the straight lines linking the circles Bachman, (2013).

**Language and Application**

The implementation of the system is largely performed in Python and a graph database application known as Neo4j. Python programming was used for the data preprocessing and sentiment analysis of the data while Neo4j plus its inbuilt cypher test was used for the recommender engine.

**System Flow**

How the system is constructed is presented in this section. Data collection, Preprocessing of a collection of hotel review including stopword removal, tokenization and classification. Furthermore, the words are classified finding the score of their sentiments. The resulting output is supplied as input into the database application and used for the recommendation of hotels.

**Dataset Sentiment Analysis**

This section gives the detail implementation of the natural language processing.

**Data Processing For Polarity**

In python, a number of libraries were installed to achieve our desired results. They include the numpy for scientific computing, textbob for processing textual data, pandas for data manipulation and analysis, NLTK for natural language processing.

First, we load the data set by reading the data (named as data.csv) using the panda read csv command and header set to "0". By setting to "0", the dataset is read with the first row (row 0) set as the header.

## System Evaluation and Results

### Relationships

There are 3,523 relationships from the two types IS_IN and Ranking. Links between the relationships and nodes.

### Property Keys

For the Hotels, city and category Property Keys created includes the address, categories, city, country, dateAdded, dateUpdated, id, keys, latitude, longitude, name, polarity, postalCode, PrimaryCategories, province, sourceurl, text, and websites. The Primary category has id and text as property keys. Figure 4 is a graphics representation of the sample of the database created

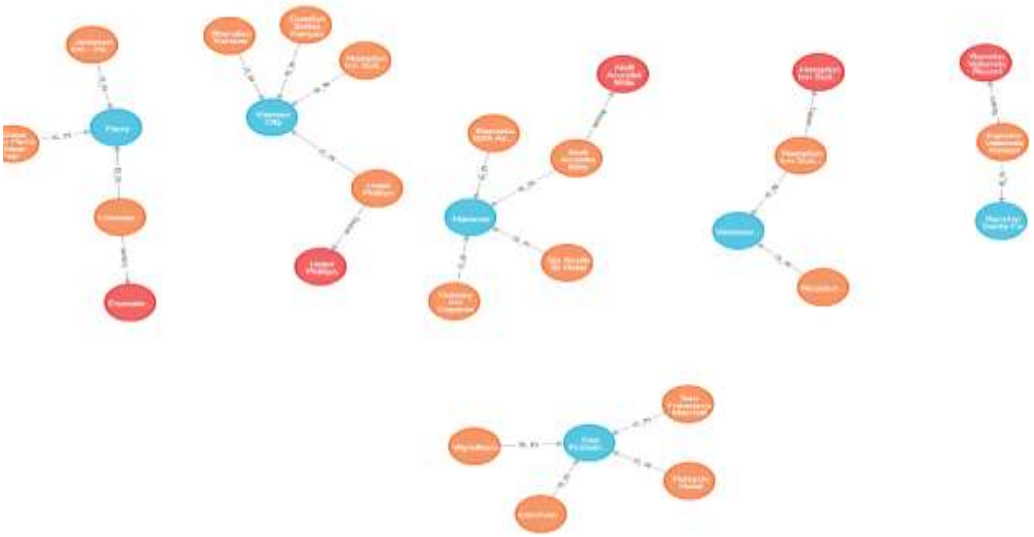

**Figure 4: A sample of the database created**

### Hotel Recommendation

From the graph database and the existing data, recommendation is made by issuing appropriate queries. Figure 5.2 shows a recommendation of the top 3 hotels (Six South St. Hotel, Holiday Inn Express Baltimore and Aloft Arudel Mills) in Hanover, US. Recall that the recommendation is based on the sentiment of the users about the hotels. Figure 5 is a screen shot of the developed python program showing the Top N Recommendation from Sentiments


TIJSRAT
ISSN: 2623-7861

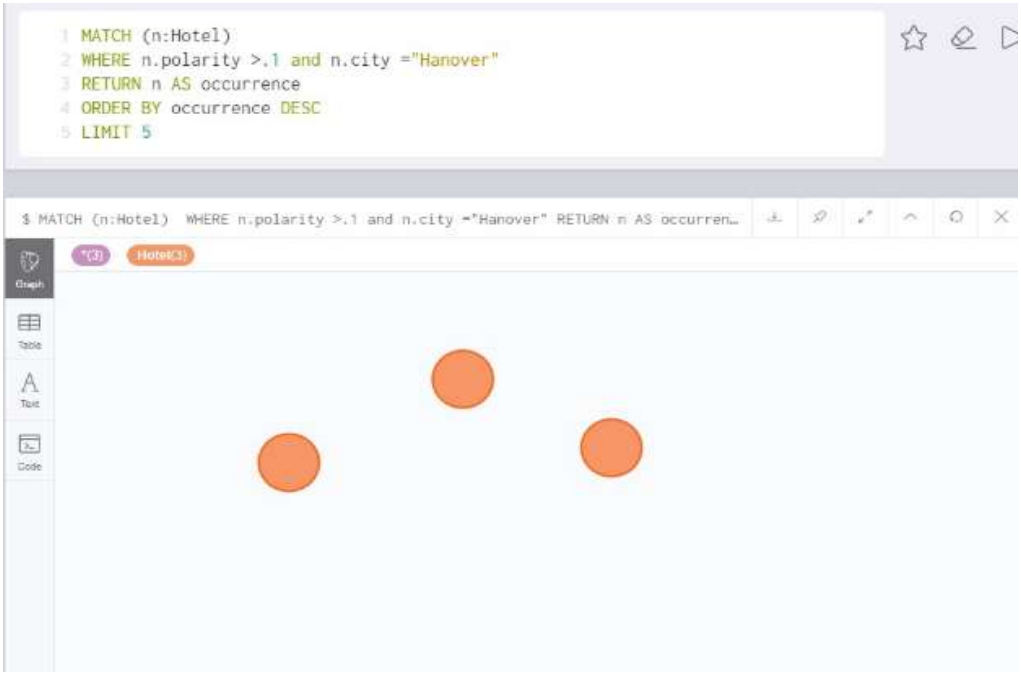


**Figure 5: Top N Recommendation from Sentiments**

**Presentation of Results**

In this section, the result of the implementation is discussed vis-à-vis probable results of using rating scale.

Sentiment Analysis Result


**Figure 6: Review text ratings versus rating scale.**

From figure 6 above, firstly we observe that between 1 and 2, 2 and 3 and so on for the rating scale, there still exists some degree of sentiments as can be seen with the review scale. These scoring cannot be ignored. For example, we cannot affirm that a user who scores on a rate of -1.0 and one who scores same hotel on a rate of -0.7 have the same sentimental position as is the case with rating scale. Hence, our proposed system

captures the actual sentiments of the users and will inevitably produce a better recommendation than when the rating scale is used.

Also worthy of note is the inconsistency noticed in the dataset of the rating score given by some users against the review text they wrote. For example, table 1 below shows a user who was not pleased with the facility from the review text but selected "4" on the rating scale.

**Table 1: Sample rating versus review text before sentiment analysis**

| Rating | Review text |
|---|---|
| 4 | Room in front of the elevator did not have isolation from the noice drunk people make coming drunk from downstairs ball room. Hey were loud and stood chatting right in front of our door. Room 608 |

One would expect such a review to be scored very low. With the implementation of sentiment analysis on the review text, the following result was obtained as shown in table 2.

**Table 2: Sample polarity versus review text after sentiment analysis**

| Polarity | Review text |
|---|---|
| -0.153571429 | Room front elevator isolation noice drunk people make coming drunk downstairs ball room. Hey loud stood chatting right front door. Room 608 |

The result therefore shows that our choice of the use of the sentiment analysis produced a better review rating when compared with the ratings and as a result yields a better recommendation result.

The overall effect of the polarity versus the ratings shows that our knowledge graph (the recommender engine) will suggest a more accurate and reliable opinion of the hotels recommended.

## CONCLUSION

In this paper, we have designed and implemented a recommender system of hotels from user review texts. We reviewed a hand full of

related work and how the system was designed and implemented, many of which used a hybrid recommender system. The system proposed is also a hybrid recommender but takes into consideration two major elements. These are, the use of review text rather than review rating and the use of a graph database generating a Knowledge graph of our dataset. The proposed system focuses on the workability of the process and not on the interface the application will run. The proposed system recommends the Top-N hotels based on the sentiment of reviews collected from the dataset. Two modules were used, first is the sentiment analysis module for generating the polarity and second is the knowledge graph module for recommending the hotels.

## Recommendation for Future Work

The researcher is of the opinion that this work can be taken more further by other scholars, by comparing the result of the present system with other recommender system so as to determine the best technique of solving a problem of this nature in the future.

## REFERENCES

Akinyemi, B. O., Amoo, A. O., Abimbola, R. O., Awoyelu, I. O., and Olaniran, A. T. (2015). An Enhanced Hybrid Item Recommender Model for Nigerian Online Stores. *International Journal of Applied Information Systems*,*10*(1), 31–42. https://doi.org/10.5120/ijais2015451459

Antony, M., Barrios, B., Quintero, C. G., and Barranquilla, M. (2017). *Intelligent Tourist Recommender System Focused on Collective Profiles*. *June*. http://manglar.uninorte.edu.co/bitstream/handle/10584/7623/intelligent.pdf?sequence=1

Bachman, M. (2013). GraphAware: Towards Online Analytical Processing in Graph Databases. *Department of Computing*, *Master*(September), 149. http://graphaware.com/assets/bachman-msc-thesis.pdf

Chen, L., Chen, G., and Wang, F. (2015). Recommender systems based on user reviews: the state of the art. *User Modeling and User-Adapted Interaction*, *25*(2), 99–154. https://doi.org/10.1007/s11257-015-9155-5

Cung, H.-Q., and Jedidi, M. (2014). *Implementing a Recommender system with graph database*. *May*. https://diuf.unifr.ch/main/is/student-projects/thesis/implementing-recommender-system-graph-database

Dalal, A. (2016). *Tourist Destination Recommendation System Based on User Facebook Profile Ankita Dalal Supervisor :*

Filzmoser, P. (2016). *Knowledge Graph based Recommendation Techniques for Email Remarketing*. *9*(3), 514–531.

Miguel, P. De. (2014). *ARLodge : context-aware recommender system based on augmented reality to assist on the accommodation search process*. *June*.

Ostuni, V. C., Di Noia, T., Di Sciascio, E., and Mirizzi, R. (2013). Top-N recommendations from implicit feedback leveraging linked open data. *Proceedings of the 7th ACM Conference on Recommender Systems*, 85–92. https://doi.org/10.1145/2507157.2507172

Pettersen, M., and Tvete, A. K. (2016). *A Hybrid Recommender System for Context-Aware Recommendations of Restaurants*. *June*. https://brage.bibsys.no/xmlui/bitstream/handle/11250/2407641/14967_FULLTEXT.pdf?sequence=1

Verma, R. P. (2015). *Relationship based Entity Recommendation System*.

Yu, X., Ren, X., Sun, Y., Gu, Q., Sturt, B., Khandelwal, U., Norick, B., and Han, J. (2014). Personalized entity recommendation: A Heterogeneous Information Network Approach. *Proceedings of the 7th ACM International Conference on Web Search and Data Mining - WSDM '14*, 283–292. https://doi.org/10.1145/2556195.2556259