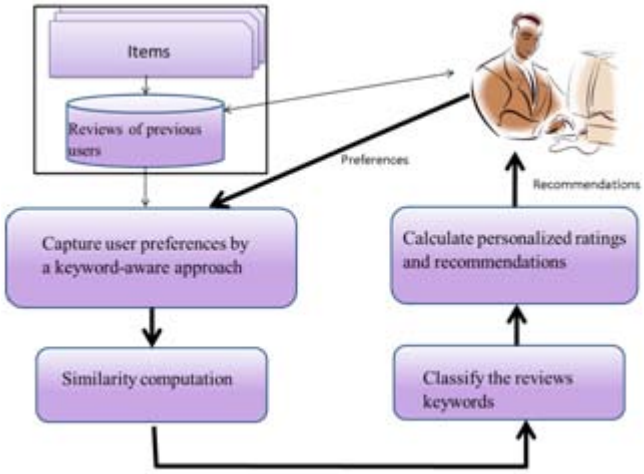
Qizmt, which is a .Net MapReduce framework, thus their system can work for large scale video sites.

Moreover, MapReduce has favourable scalability and efficiency. Hadoop MapReduce is a software framework for easily writing applications which process vast amounts of data (multi-terabyte datasets) in parallel on large clusters (thousands of nodes) of commodity hardware in a reliable, fault tolerant manner. A MapReduce job usually splits the input dataset into independent chunks which are processed by the map tasks in a completely parallel manner. The framework sorts the outputs of the maps, which are then input to the reduce task

3. Proposed Recommendation Method

The project proposesa novel method of personalized recommendation system. In which Keyword-candidate List and Domain Thesaurus are maintained for particular system. Preferences are taken from user. And similar users are searched out by keyword extraction method and similarity calculations. Then the keywords are classified, and weights of reviews of similar users are calculated. Then finally, recommendation list of top-k items is generated.



In proposed method, keywords are used to indicate both of users' preferences and the quality of candidate services. A user based CF algorithm is adopted to generate appropriate recommendations. Proposed system aims at calculating a personalized rating of each candidate service for a user, and then presenting a personalized service recommendation list and recommending the most appropriate services to him/her.

Table 1Summarizes the basic symbols and notations used in next algorithms.

Table 1: The basic symbols and notations

Symbol	*Definition*
K	The keyword candidate list, K={k1, k2, …,kn}
APK	The preference keyword set of the active user
PPK	The preference keyword set of a previous user
sim(APK,PPK)	The similarity between APK and PPK
$\overrightarrow{W_P}$	A preference weight vector
$\overrightarrow{W_{AP}}$	Preference weight vector of the active user
$\overrightarrow{W_{PP}}$	Preference weight vector of a previous user

3.1 Keyword Candidate List and Domain Thesaurus

In our method, two data structures, “keyword candidate list” and “specialized domain thesaurus”, are introduced to help obtain users' preferences.

Keyword candidate list: The keyword candidate list is a set of keywords about users’ preferences and multi-criteria of the candidate services [11], which can be denoted as K= 1 where n is the number of the keywords in the keyword candidate list. An example of a simple keyword candidate list of the hotel reservation system is described in Table 2.

Table 2: sample keyword candidate list

No.	*Keyword*	*No.*	*Keyword*
1.	Service	6.	Transportation
2.	Room	7.	Location
3.	Food	8.	Cleanliness
4.	Shopping	9.	Environment
5.	Value	10.	Fitness

Keywords in the keyword candidate list can be a word or multiple words related with the quality criteria of candidate services. In this method, the preferences of previous users will be extracted from their reviews for candidate services and formalized into a keyword set. Usually, since some of words in reviews cannot exactly match the corresponding keywords in the keyword candidate list which characterize the same aspects as the words. The corresponding keywords should be extracted as well. In KASR [11], specialized domain thesauruses are built to support the keyword extraction, and different domain thesauruses are built for different service domains.

Domain thesaurus: A domain thesaurus is a reference work of the keyword candidate list that lists words grouped together according to the similarity of keyword meaning, including related and contrasting words and antonyms.

3.2 User Preferences/ choices

In this step, the preferences of active users and previous users are formalized into their corresponding preference keyword sets respectively. In this method, an active user refers to a current user needs recommendation.

Preferences of an active user: An active user can give his/her preferences about candidate services by selecting keywords from a keyword candidate list, which reflect the quality criteria of the services he/she is concerned about. The preference keyword set of the active user can be denoted as APK= $\{ak_1, ak_2, .., ak_l\}$ where ak_i $(1 \leq i \leq l)$ is the ith keyword selected from the keyword candidate list by the active user, l is the number of selected keywords.

Preferences of previous users:
The preferences of a previous user for a candidate service are extracted from his/her reviews for the service according to the keyword candidate list and domain thesaurus. And a review of the previous user will be formalized into the preference keyword set of him/her, which can be denoted as PPK= $\{pk_1, pk_2, .., pk_h\}$ where pk_i $(1 \leq i \leq h)$ is the ith

keyword extracted from the review, h is the number of extracted keywords.

The keyword extraction process is described as follows:

a) Preprocess

Firstly, HTML tags and stop words in the reviews snippet collection should be removed to avoid affecting the quality of the keyword extraction in the next stage. And the Porter Stemmer algorithm (keyword stripping) is used to remove the commoner morphological and in flexional endings from words in English. Its main use is as part of a term normalization process that is usually done when setting up Information Retrieval systems.

b) Keyword Extraction

In this phase, each review will be transformed into a corresponding keyword set according to the keyword candidate list and domain thesaurus. If the review contains a word in the domain thesaurus, then the corresponding keyword should be extracted into the preference keyword set of the user. For example, if a review of a previous user for a hotel has the word “spa”, which is corresponding to the keyword ``Fitness'' in the domain thesaurus, then the keyword ``Fitness'' should be contained in the preference keyword set of the previous user. If a keyword appears more than once in a review, the times of repetitions will be recorded. In this method, it is regarded that keywords appearing multiple times are more important. The times of repetitions will be used to calculate the weight of the keyword in preference keyword set in the next step.

c) Classify Keywords

When a keyword is captured, it is classified into positive or negative keyword with respect to the meaning of particular word in sentence. For the classification, naive bayes algorithm will be used. Keyword classification is described later in section IV.

d) Similarity Calculation

The second step is to identify the reviews of previous users who have similar tastes to an active user by finding neighborhoods of the active user based on the similarity of their preferences. Before similarity computation, the reviews unrelated to the active user's preferences will be filtered out by the intersection concept in set theory. If the intersection of the preference keyword sets of the active user and a previous user is an empty set, then the preference keyword set of the previous user will be filtered out.

Two similarity computation methods are introduced in our recommendation method: an approximate similarity computation method and an exact similarity computation method. The approximate similarity computation method is for the case that the weights of the keywords in the preference keyword set are unavailable, while the exact similarity computation method is for the case that the weight of the keywords are available.

3.3 Approximate Similarity Computation

A frequently used method for comparing the similarity and diversity of sample sets, Jaccard coefficient, is applied in the approximate similarity computation. Jaccard coefficient is measurement of asymmetric information on binary (and nonbinary) variables, and it is useful when negative values give no information. The similarity between the preferences of the active user and a previous user based on Jaccard coefficient is described as follows:

$$sim(APK, PPK) = Jaccard(APK, PPK) = \frac{|APK \cap PPK|}{|APK \cup PPK|}$$

Where APK is the preference keyword set of the active user, PPK is the preference keyword set of a previous user. And the weight of the keywords is not considered in this approach.

Algorithm 1, SIM-ASC, illustrates the functionality of the approximate similarity computation method.

Algorithm 1: SIM-ASC (Approximate Similarity Computation)

Input: The preference keyword set of the active user APK
The preference keyword set of a previous user PPK_j
Output: The similarity of APK and PPKj, sim_{ASC}(APK, PPK_j)
1: $sim_{ASC}(APK, PPK_j) = \frac{|APK \cap PPKj|}{|APK \cup PPKj|}$
2: return similarity of APK and PPKj, sim_{ASC}(APK, UPK_j)

3.4 Exact similarity computation:

A cosine based approach is applied in the exact similarity computation, which is similar to the Vector Space Model (VSM) in information retrieval.

Preference weight vector: In this cosine based approach, The preference keyword sets of the active user and previous users will be transformed into n-dimensional weight vectors respectively, namely preference weight vector, which can be denoted as $\overrightarrow{W}_P = [w_1, w_2, .., w_n]$, n is the number of keywords in the keyword candidate list,w_i is the weight of the keyword k_i in the keyword candidate list. If the keyword k_i is not contained in the preference keyword set, then the weight of k_i in the preference weight vector is 0, i.e., w_i=0. The preference weight vectors of the active user and a previous user are noted as $\overrightarrow{W}_{AP}$ and $\overrightarrow{W}_{PP}$ respectively.

In KASR method [11], the Analytic Hierarchy Process (AHP) model decides the weight of the keywords in the preference keyword set of the active user. AHP method is provided by Saaty in 1970s to choose the best satisfied business role for its hierarchy nature. The weight computing based on the AHP model is decided as follows:

Firstly, pairwise comparison matrix in terms of the relative importance between each two keywords is constructed. The pairwise comparison matrix QUOTE $A_m = (a_{ij})A_m = (a_{ij})$, where m must satisfy the following properties, a_{ij} represents the relative importance of two keywords:

a_{ij} = 1, i=j=1,2,3..,m

$a_{ij} = {}^{1}/_{a_{ji}}$, i,j =1,2,3..,m and i ≠ j

$a_{ij} = {}^{a_{ik}}/_{a_{jk}}$, i,j , k =1,2,3..,m and i ≠ j

After checking the consistence of the matrix, then calculate the weight by the following function:

$$w_i = \frac{1}{m}\sum_{j=1}^{m}\frac{a_{ij}}{\sum_{k=1}^{m} a_{kj}}$$

Where a_{ij} is the relative importance between two keywords, m is the number of the keywords in the preference keyword set of the active user. The weight vector of the preference keyword set of a previous user can be decided by the term frequency/inverse document frequency (TF-IDF) measure, which is one of the best known measures for specifying the weight of keywords in Information Retrieval.

In the TF-IDF approach, to calculate the preference weight vector of a previous user u', "all reviews" by user u' should be collected. Here, "all reviews" contain the reviews by user u' for the candidate services and similar services not in the candidate services. The reviews should also be transformed into keyword sets respectively according to the keyword candidate list and the domain thesaurus.

TF, the term frequency of the keyword pk_i in the preference keyword set of user u' is defined as

$$TF = \frac{N_{pk_i}}{\sum_g N_{pk_i}}$$

Where N_{pk_i} is the number of occurrences of the keyword pk_i in all the keyword sets of the reviews commented by the same user u', g is the number of the keywords in the preference keyword set of the user u'. The inverse document frequency (IDF) is obtained by dividing the number of all reviews by the number of reviews containing the keyword pk_i.

$$IDF = \log\frac{|R'|}{|r' : pk_i \in r'|}$$

where |R'| is the total number of the reviews commented by user u', and |r': pki∈ r'| is the number of reviews where keyword pk_i appears. So the TFIDF weight of the keyword pki in the preference keyword set of user u' can be decided by the following function:

$$w_{pk_i} = TF * IDF = \frac{N_{pk_i}}{\sum_g N_{pk_i}} * \log\frac{|R'|}{|r' : pk_i \in r'|}$$

Then the similarity based on the cosine-based approach is defined as follows:

$$sim(APK, PPK) = \cos(\vec{W}_{AP}, \vec{W}_{PP}) = \frac{\vec{W}_{AP} \bullet \vec{W}_{PP}}{\|\vec{W}_{AP}\|_2 \times \|\vec{W}_{PP}\|_2}$$

$$= \frac{\sum_{i=1}^{n} \vec{W}_{AP.i} \times \vec{W}_{PP;i}}{\sqrt{\sum_{i=1}^{n} \vec{W}^2_{AP,i}}\sqrt{\sum_{i=1}^{n} \vec{W}^2_{PP,i}}}$$

Where $\vec{W}_{AP}$and $\vec{W}_{PP}$ are respectively the preference weight vectors of the active user and a previous user. $\vec{W}_{AP.i}$is the ith dimension of $\vec{W}_{AP}$ and represents the weight of the keyword k_i in preference keyword set APK, $\vec{W}_{PP;i}$ is the i-th dimension of $\vec{W}_{PP}$ and represents the weight of the keyword k_i in preference keyword set PPK. Algorithm 2, SIM-ESC, illustrates the functionality of the exact similarity computation method.

Input: The preference keyword set of the active user APK
The preference keyword set of a previous user PPKj
Output: The similarity of APK and PPKj, sim_{ESC}(APK, PPKj)
1. for each keyword ki in the keyword-candidate list
2. if ki ϵ APK then
   Calculate $\vec{W}_{AP,i}$
3. Else
   $\vec{W}_{AP,i} = 0$
4. End If
5. if ki ϵ PPK then
   Calculate $\vec{W}_{PP,i}$
6. Else
   $\vec{W}_{PP,i} = 0$
7. End if
8. End for
9. Calculate sim_{ESC}(APK, UPKj)
10. Return the similarity of APK and PPKj, sim_{ESC}(APK, UPKj)

3.5 Generate Personalized Recommendation List

Based on the similarity of the active user and previous users, further filtering will be conducted. Given a threshold δ, if sim(APK, PPK_j) <δ , the preference keyword set of a previous user PPKj will be filtered out, otherwise PPKj will be retained. The thresholds given in two similarity computation methods are different, which are both empirical values. Once the set of most similar users are found, the personalized ratings of each candidate service for the active user can be calculated. Finally, a personalized service recommendation list will be presented to the user and the service(s) with the highest rating(s) will be recommended to him/her.

Here, a weighted average approach is used to calculate the personalized rating pr of a service for the active user.

$$pr = \bar{r} + k\sum_{PPKi \in \hat{R}} sim(APK, PPKj) \times (rj - \bar{r})$$

Where $k = 1/\sum_{PPKi \in \hat{R}} sim(APK, PPKj)$

Where sim(APK, PPK) is the similarity of the preference keyword set of the active user APK and the preference keyword set of a previous user PPK_j ; multiplier k serves as a normalizing factor; $\hat{R}$denotes the set of the remaining preference keyword sets of previous users after filtering; r_j is the rating of the corresponding review of PPKj, and r is defined as the average ratings of the candidate service.

Repeating the steps above, the personalized ratings of all candidate services for the active user are calculated. Then rank the services by the personalized ratings and present a personalized service recommendation list to him/her. Without loss of generality, it is assumed that the services with higher ratings are more preferable to the user.

4. Results

Expected results will include, most appropriate recommendation list. For simplification of expected results, some assumptions are made.

1) All keywords have same weight.
2) Weight of positive keyword is assumed to be +1.

3) weight of negative keyword is assumed to be -1.
Example: let's consider the following review for mobile phone model.
Preference keyword set of user: light sensor, battery backup, gesture sensor, internet speed, LED display

- **Review:** Overall performance is good..... Quad-core processor makes it superfast. I am amazed to see that this phone give me battery backup of 2 days but it don't have light sensors, gesture sensor and also its internet speed is only 21.1mbps which is slow compare to other smartphones at this price.
- **Keywords captured:** performance, processor, battery backup, light sensors, gesture sensor and internet speed

4.1 Output of existing System

5 keywords are extracted from particular review. And amongst them, 4 keywords matches with active user's preferences. It implies that weight of particular product is +4. So, there are more chances that the product will likely to fall in recommendation list.

4.2 Output of proposed System

The above review consists of 5 keywords. From these 5 keywords, 4 keywords match with user preferences. But by checking the sentences structure, one can easily find out the sense behind keyword use. On this basis, keywords are classified into positive and negative keywords

Classification of keywords:
Positive keywords: performance, processor, battery backup
Negative Keywords: light sensors, gesture sensor, internet speed
The weight of review = (+1) – (3) = (-2)

With this weight factor, there is not any chance that product will be recommended to user. In this way, a more personalized and appropriate recommendation list is generated for user. So, as per new method, output of recommendation system will be more personalized and efficient for user. And by parallelizing algorithm processing, system will be more time efficient, than existing system.

5. Conclusion

The proposed system is more efficient in terms of complexity. And the system gives more accurate results or recommendations to the users. This system is being developed for products based on amazon data set.

References

[1] C. Lynch, "Big Data: How do your data grow?," Nature, vol. 455, no. 7209, pp. 28-29, 2008.
[2] M. Chui, B. Brown, et al Manyika, "Big Data: The next frontier for innovation, competition, and productivity," , 2011
[3] J. Dean, S. Ghemawat, and W. C. Hsieh F. Chang, "Bigtable: A distributed storage system for structured data," ACM Transactions on Computer Systems, vol. 26, no. 2.
[4] X. Zhang, J. Liu, J. Chen W. Dou, "HireSome-II: Towards Privacy-Aware cross Cloud Service Composition for Big Data Applications," IEEE Transactions on Parallel and Distributed Systems, 2013
[5] B. Smith, and J. York G. Linden, "Amazon.com Recommendations: Item-to-Item collaborative Filtering," IEEE Internet Computing, vol. 7, no. 1, pp. 76-80, 2003.
[6] M. Bjelica, "Towards TV Recommender System Experiments with User modeling," IEEE Transactions on Consumer Electronics, vol. 7, no. 1, pp. 1763-1769, 2010.
[7] F. Alvarez, J. Menendez, and O. Baez M. Alduan, "Recommender System for Sport Videos Based on User Audiovisual Consumption," IEEE Transactions on Multimedia, vol. 14, no. 6, pp. 1546-1557, 2013.
[8] Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," Communications of the ACM, vol. 51, no. 1, pp. 107-113, 2005.
[9] A. Tuzhilin and G. Adomavicius, "Toward the Next Generation of Recommender Systems: A Survey of the State of the Art and Possible Extensions," IEEE Transactions on Knowledge and Data Engineering, vol. 17, no. 6, pp. 734-749, 2005.
[10] A. Cheng and W. Hsu Y. Chen, "Travel Recommendation by Mining People Attributes and Travel Group Types From Community Contributed Photos," IEEE Transactions on Multimedia, vol. 25, no. 6, pp. 1283-1295, 2012.
[11] Wanchun Dou, Xuyun Zhang, Jinjun Chen Shunmei Meng, "KASR: A Keyword-Aware Service Recommendation Method on MapReduce for Big Data Applications," IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, vol. 99, no. 2, 2014.
[12] Y. Guo, Y. Liu X. Yang, "Bayesian-inference based recommendation in online social networks," IEEE Transactions on Parallel and Distributed Systems, vol. 24, no. 4, pp. 642-651, 2013.
[13] Z D Zhao and M. S. Shang, "User Based Collaborative Filtering Recommendation Algorithms on Hadoop," In the third Internation-al Workshop on Knowledge Discovery and Data Mining, pp. 478-481, 2010.
[14] M. Hu, H. Singh, D. Rule, M. Berlyant, and Z. Xie Y. Jin, "MySpace Video Recommendation with Map-Reduce on Qizmt," Proeedings of the 2010 IEEE Fourth International Conference on Semantic Computing, pp. 126-133, 2010.