Weighted Itemset Mining from Bigdata using Hadoop

Divya.M.G[#], Nandini.K[#], Priyanka.K.T[#], Vandana.B*
[#]*UG Student, *Assistant Professor*
Department of CS&E, RRCE, Visvesvaraya Technological University, INDIA
Divyashobha11@gmail.com, k.nandini95@gmail.com, priyankakt07@gmail.com, vandanacse@yahoo.co.in

***Abstract*—** Data items have been extracted using an empirical data mining technique called frequent itemset mining. In majority of theapplication contexts items are enriched with weights. Pushing an item weights into the itemset extraction process, i.e., mining weighted itemsets rather than traditional itemsets, is an appealing research direction. Although many efficient weighteditemset mining algorithms are available in literature, there isa lack of parallel and distributed solutions which are able to scale towards Big Weighted Data. This Proposed work presents an efficient frequent weighted itemset mining algorithm based on the MapReduce paradigm. It adopts the MapReduce architecture to partition thewhole mining tasks into smaller independent subtasks and uses Hadoop distributed file system to manage distributed data so that it allows the parallel and distributed solution.To demonstrate its actionability and scalability, the proposed algorithm will be tested on a Bigdataset collecting large volume of reviews ofitems. Weights indicate theratings given by users to the purchased items. The mined itemsets represent combinations of items that were frequently bought together with an overall rating above average.

Keywords-*MapReduce, Parallel Computing, hadoop, frequentitemset, Data mining, Distributed Computing, Apriori Algorithm.*

1. INTRODUCTION

The swift growth of data generated and stored has directed us to the new era of Big Data. Nowadays, we are surrounded bydifferent kinds of big data, such ase-commerce platform, sensor data, machine-generated data and social data. Extracting valuable information and insightful knowledge from big data has become an urgent need in many disciplines. In view of this, *big data analytics* [3, 4,] has emerged as a novel topic in recent years. This technology is particularly important to enterprisesand business organizations because it can help them to increase revenues, retain customers and make more intelligent decisions. Due to its high impact in many areas, more and more systems and analytical tools have been developed for big data analytics, such as *Apache Mahout*. And MLlib [5] are notable examples of MapReduce- and Sparkbased scalable machinelearning.

In real-life applications items are unlikely to be equally important within the analyzed data. For example, items purchased at the market have different prices, medical treatments have different urgency levels, and genes are expressed in biological samples with different levels of significance. Hence, an appealing extension of traditional itemset mining algorithms is to push of item relevance weights into the mining process. We propose Parallel Weighted Itemset miner, a new parallel and distributed framework to extract frequent weighted itemsetsfrom potentially very large transactional datasets enriched with item weights. The framework relies on a parallel and distributed-based implementation running on a Hadoop cluster. To make the mining process scalable towards Big Data, most analytical steps performed by the system are mapped to the MapReduce programming paradigm.

The Changes in the data access patterns of applications and the need to scale out to thousands of commodity machines led to the birth of a new class of systems referred to as weighted stores [8, 10, 18] which are now being widely adopted by

various enterprises. In the domain of data analysis, the MapReduce paradigm [17] and its open-source implementation Hadoop [20] has also seen widespread adoption in industry and academia alike. Solutions have also been proposed to improve Hadoop based systems in terms of usability [1, 28] and performance.

2. RELATED WORK

Different papers describes concept of data set mining as follows: (i) Large-scale item set mining (ii) weighted itemset mining. In large scale item set mining, exploratory Frequent itemset and association rule mining are widely data mining techniques which are first introduced in [6]. To scale towards large datasets most significant efforts have been devoted to studying parallel and distributed itemset mining strategies. For example, an Apriori-based [9] approach to mining frequent item sets has been presented in. Since Apriori is known to be less scalable than projection-based and vertical algorithms on complex datasets [10], many attempts to parallelize and distribute different itemset mining strategies have also been made (e.g., [12], [13]). For example, BigFIM is a hybrid algorithm based on MapReduce, which combines principles from both Apriori [9] and Eclat [11]. Solutions relying on FP-Growth like strategies have also been proposed. For example, the authors exploited prefix-tree-like structures to drive theparallel itemset mining process. The mining process entails the following steps: first, a horizontal subset of the data is analyzed and a local FP-Tree is built; then the item set mining process is performed on the local FP-Tree. Finally, the candidate pattern bases from different processing flows are then merged together. In an enhanced strategy for merging

processing flows has been proposed, more recently, the Parallel FP-Growth algorithm [13] parallelizes different instances of the recursive FP-Growth process on distributed machines. The key idea is to partition the computation in such a way that each machine executes an independent group of mining tasks thus reducing the communication between machines. To balance the computation load on different machines the authors proposed to consider the support of singletons, while the works presented and exploited clustering techniques and data sampling to limit the computational complexity of each mining task. An attempt to discover misleading patterns from Big datasets using MapReduce has been made in [14]. The idea is to compare frequent (unweighted) item sets mined at different abstraction levels to highlight potentially critical situations. Unlike all the aforesaid approaches, this paper addresses weighted itemset mining instead of traditional itemset mining. To scale towards towards Big Data, the Proposed framework relies parallel and distributed-based implementation running on a Hadoop cluster, where most mining steps are mapped to the MapReduce programming paradigm. Weighted itemset mining. In the traditional itemset and rule mining tasks items belonging to each transaction of the source dataset are treated equally. To differentiate items based on their relevance within each transaction, in [15] the authors first addressed the issue of mining more informative association rules, i.e., the Weighted Association Rules (WAR). WARs are association rules enriched with weights denoting item significance. Weights were introduced only during the rule generation step after performing the traditional frequent itemset mining process. To improve the efficiency of the mining process, the authors in [16] pushed item weights deep into the itemset mining process by exploiting the anti-monotonicity of the weighted support measure in an Apriori-based itemset mining process [9]. In [18] a FP-Growth-like weighted itemset mining algorithm process is presented.Unlike [15], [16], the algorithm proposed in [18] extracts infrequent (rare) item sets rather than frequent ones. A parallel issue is the extraction of weighted item sets and rules when coping with data not equipped with preassigned weights. For example, to generate appropriate item weights, in [17] the dataset is modeled as a bipartite hub-authority graph and evaluated by means of a well-known indexing strategy.

3. SYSTEM DESIGN

Hadoop Framework

Hadoop framework is allows data storing and running applications on clusters of commodity hardware [28]. It provides massive storage for any kind of data. Hadoop is the parallel programming platform built on Hadoop DISTRIBUTED File Systems (HDFS) for MapReduce computation. HDFS is highly fault-tolerant and is designed to bedeployed on low-cost hardware. HDFS holds very large volume of data and provide easier access. HDFS also makes applications available to parallel processing. HDFS is a part of Apache Hadoop main project.

MapReduce

MapReduce is a programming model and an associatedimplementation for processing and generating large datasets.Users specify a *map* and *reduce* functions, they takes <*key*,*value*>pair as an input and generates intermediate <*key*,*value*>pairs and merges all intermediate values associatedwith the same intermediate key respectively.

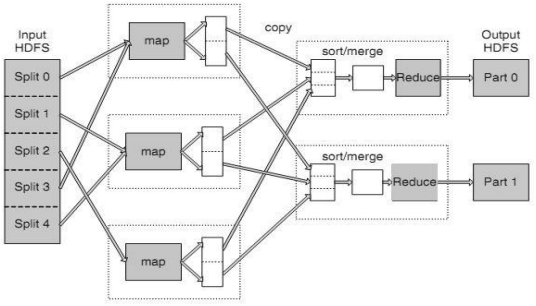

Fig. 1 Hadoop MapReduce Architecture.

Weigted ItemSet Mining System Architecture

This paper presents a scalable frequent weighted itemset mining algorithm based on the MapReduce paradigm. To demonstrate its action ability and scalability, the proposed algorithm will be tested on a Bigdataset collecting large volume of reviews of items. Weights indicate the ratings given by users to the purchased items. The mined itemsets represent combinations of items that were frequently bought together with an overall rating above average.

It integrates a variant of the BigFIM algorithm which is able to successfully cope with data enriched with weights. Furthermore, to allow experts to effectively explore the result of the mining process, the proposed system allows us to rank itemsets by (i) weighted support, (ii) traditional support, and (iii) a mix of the above. While the traditional support indicates the generic degree of interest of the considered combination of items, the weighted support integrated in the proposed framework indicates the average level of interest of the least interesting item within each transaction. The proposed system, running on an Hadoop cluster, overcomes the limitations of state-of-theart approaches in coping with datasets enriched with item weights.

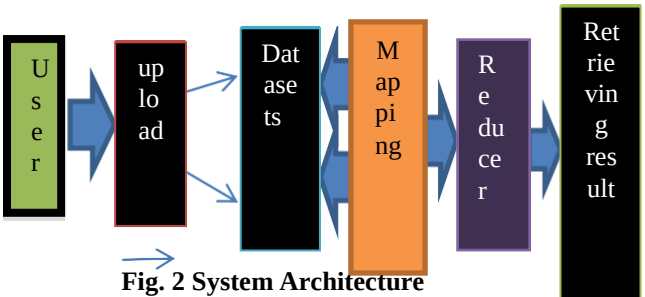

Fig. 2 System Architecture

Modules Split Up

Module 1: Registration module

This module is used for the user to register their login id by providing the minimal information. so that they can login to the website.
Module 2: Sign in module
In this, user can login to the website by registered login id and a valid password. Only the authenticated user can login and use the website.
Module 3: User module
In this module, user extracts itemset which are frequently brought together with an overall rating above average.
Module 4: Admin module
In this module, admin checks the item list, add the items and remove the unwanted items.
Module 5: Insert Dataset module
The large numbers of itemsets will be inserted to the database.

4. PARALLEL WEIGHTED ITEMSET MINING FROM BIGDATA

Parallel Weighted Itemset Miner is a new data mining environment aimed to analyze Big Data equipped with item weights. The main environment blocks are briefly introduced below. A more detailed description is given in the following section.

Data preparation

This step entails preparing data to the subsequent item set mining process. The source data is acquired, stored in a transactional dataset, and equipped with item weights. A transactional datasetis a set of transactions. Each transaction is a set of (not repeated) tems. Depending on the context of analysis, items may represent different concepts (e.g., products, objects, places, stocks). For example, let us consider the dataset reported in Table I. It is an example of (unweighted) transactional dataset consisting of five transactions, each one representing a different customer ofa e-commerce company. For each customer the list of purchased items is known. For instance, customer with id 1 bought items *X*, *Y*, and *Z*. Note that each transaction, which represents a distinct electronic basket, may contain an arbitrary number of items. To consider the relative importance of the items within each transaction during the item set mining process, itemsare enriched with weights. A transactional dataset whose items are equipped with weights is denoted as weightedtransactional dataset.

TABLE I: EXAMPLE OF UNWEIGHTED DATASET: ITEM BOUGHT BY CUSTOMERS

Customer id	Purchased items
1	*X*, *Y* , *Z*
2	*X*, *Y* , *Q*
3	*X*, *Y* , *Z*
4	*X*, *Y* , *Q*
5	*X*, *W*, *Z*

TABLE II: EXAMPLE OF WEIGHTED DATASET: ITEM RATINGS GIVEN BY CUSTOMERS

Customer id	Purchased items and ratings
1	*(X*, 3*)(Y*, 1*) (Z*, 5*)*
2	*(X*, 2*)(Y*, 2*) (Q*, 2*)*
3	*(X*, 4*) (Y*, 2*) (Z*, 5*)*
4	*(X*, 3*) (Y*, 3*) (Q*, 2*)*
5	*(X*, 2*)(W*, 5*)(Z*, 4*)*

item is a pair *_item*, *weight_*, where *weight* is the weight associated with the corresponding *item*. For example, let us consider the weighted transactional dataset reported in Table II. It extends the traditional transactional dataset in Table I by enriching items with the corresponding weights. More specifically, for each customer the rating (from one to five) given to each purchased item is known. For instance customer with id 1 rated item *X* as 3, item *Y* as 1, and item *Z* as 5. The analyzed data are tailored to a weighted transactional data format. Furthermore, if need be, ad hoc preprocessing steps are applied to the raw data to ensure high-quality results. For example, data filtering and discretization areexamples of commonly used preprocessing steps [19]. Data filtering entails discarding the items/transactions that are irrelevant for subsequent analyses. For instance, recalling the previous example, duplicate entries of the same customer basket can be removed because they could bias item set support counts. To ensure the scalability of the knowledge discovery process, the PaWI system performs data filtering as a distributed MapReduce job.

Weighted Item setMining

This step focuses on mining frequent weighted itemsets[15] from the prepared weighted dataset. A *k*-itemset(i.e., an itemset of length *k*) is a set of *k* items. Thetraditional support value of an itemset in a transactional dataset is given by its frequency of occurrence in the source dataset [6]. For example, *{X,Y}* is an itemset indicating the co-occurrence of items *X* and *Y* . If we disregard item weights, this itemset has a support equal to 4 in Table I because it occurs in four out of five transactions, meaning that most of the users purchased items *X* and *Y* together. The goal of this paper is to extend traditional large-scale itemset miners to successfully cope with Big weighted data. Hence, for our purposes, the itemset support measure is extended, similar to [15], to the case of weighted data. As previously done in [18], the weighted support of an itemset*I* in a weighted transactional dataset *T* is defined as a linear combination of the aggregation weights computed on each transaction in *T*. An arbitrary aggregation function*f*(e.g., min, max, average, and mode) can be potentially applied to aggregate item weights within each transaction. The choice of *f* depends on the considered use cases. Hereafter, similar to [18], we will consider *f*=min (i.e., the least weight of any item in *I* is considered), because, as

discussed, the selected patterns are deemed as particularly useful for analyzing real Big datasets. Recalling the running example, let us consider analystswho would like to discover the combinations of items that were frequently bought together with an overall rating above average. To this aim, we may consider item ratings during support computation by weighting item set occurrences within each transaction by the least item rating. For instance, recalling the weighted transactional datasetin table for customer with id 1 between *X* and *Y* the item with least rating is *Y* (rating equal to 1), while for customer with id 5 is *X* (rating equal to 2). Hereafter we will denote as *weighted support* the support of an item set by considering item weights, whereas as *traditional support* the item set support disregarding item weights. For instance, *{X,Y}* has weighted support equal to 1+2+2+3+0=8 and traditional support equal to 4. Given a weighted transactional dataset *D* and an (analyst provided)minimum support threshold *minus*, the PaWIsystem addresses the extraction of all frequent weighted item sets from *D*. To allow comparing weighted item sets with traditional ones, PaWI allows experts to mine traditional item sets as well. As discussedbelow, the support thresholds enforced during weighted and unweighted item set mining are potentially different. The weighted item set mining process relies on a parallel and distributed-based algorithm running on a Hadoop cluster [20]. To make the mining process scalable towards Big Data, the mining steps are mapped to the MapReduceprogramming paradigm. MapReduce [21] is a parallel programming framework providing both a relatively simple programming interface together and a robust computational architecture. MapReduce programs consist of two main steps. In the map step, each mapper processes a distinct chunk of the data and produces key-value pairs. In the reduce step, key-value pairs from different mappers are combined by the framework and fed to reducers as pairs of key and value lists. Reducers generate the final output by processing the key/value lists. To efficiently perform frequent weighted item set mining with MapReducePaWI integrates a variant of the BigFIM algorithm [12] which is able to successfully cope with data enriched with weights. The exploitation of weights is challenging because ad-hoc data structures must be used to efficiently maintain the weights associated with each item and transaction by balancing the impacts on computational and communication costs. The following extensions have been proposed: Distributed transaction splitting. BigFIM relies on two established item set miners: Apriori [9] and Eclat [11]. We modified the BigFIM algorithm to allow both Apriori andEclat to successfully cope with weighted data. More specifically, our algorithm generates an equivalent version of the source dataset that includes only transactions with equally weighted items. Let us assume that the weight of an equivalent transaction *tq*is*w*. Then, the occurrence of any item set in *tq*will be weighted by *w* instead of by 1. Each transaction in the original dataset may correspond to a set of equivalent transactions in the equivalent dataset. A formal definition of the equivalence set of weighted transactions is given in [18]. Note that since two distinct transactions have disjoint equivalent sets the splitting process is straightforwardly parallelizable. Weighted support counting.Since items are equipped with weights, traditional support counting is replaced with weighted support counting, according to Definition 1. Toaccomplishitemsetsupport counting different strategies areadopted according to the algorithm used. Specifically, to perform Apriori-based weighted itemset mining, item set supports are counted by generalizing the word counting problem for documents [21] to weighted item sets, i.e., each mapper receives a disjoint subset of dataset transactions (i.e., the documents) and reports the items/item sets (i.e., the words) for which the weighted support count is performed. A reducer combines all partial weighted support counts and reports only the items/item sets whose weighted support is above the threshold. These frequent weighted item sets are redistributed to all mappers to act as candidates for the next step of breadth-first search [9] and then the procedure is repeated to mine weighted item sets of higher length. To perform Eclat-based weighted itemset mining, each mapper builds the weighted tidlist of the item sets related to a subset of transactions. The weighted tidlist of an arbitrary item *i j* consists of all pairs (*transaction id*, *weight*) such that the transaction related to *transaction id* contains item *i j* with weight. For example, let us assume that a mapper receives the transactions contained in the dataset in Table II. For item *Z* it generates the following weighted tidlist: *{*(cid1,5),(cid3,5),(cid5,4)*}*. The weighted tidlist consists of all pairs (*customer id*, *weight*) for which the transaction related to *customer id* contain item *Z* with weight. For instance, the transaction corresponding the electronic basket of the customer with id 1 contains item *Z* with weight 5. A reducer combines all partial weighted support counts and reports only the items/item sets whose weighted support is above the threshold. Note that the equivalent transaction weights are not stored in the distributed cache, because Bigdatasets may potentially consist of millions of transactions.

Item set Ranking

The manual exploration of all the item sets (weighted or not) mined from Big data is practically unfeasible. Hence, to support the knowledge discovery process experts may wouldlike to access only a subset of most interesting patterns. This step focuses on ranking the mined item sets according to their level of significance in the analyzed data. To filter and rank the mined item sets, the support measure is the most commonly used quality index [6]. To cope with weighted data, for each candidate itemset the PaWIsystem computes both the traditional and weighted support measures. While the traditional support value indicates the observed frequency of occurrence of the considered combination of items in the source dataset, in weighted support counting itemset occurrences are weighted by the least item weight (see Definition 1). To select item sets whose average least item weight is maximal the PaWI system combines the weighted and traditional support measure in a new measure called AW-

sup, i.e., the *Average Weighted support*. The AW-sup measure is defined as the ratio between the weighted emset support and the traditional itemset support. It indicates the average per-transaction weight of the least weighted item. Selecting top interesting item sets based on this measure is potentially
interesting in real applications. For example, let us consider again the example dataset in Table II. According to Definition 1, itemset*{X,Y}* has weighted and traditional support values equal to 8 and 4, respectively. Since transactions represent electronic baskets, the weighted itemset support indicates the overall least item rating computed on the subset of customers who bought both items *X* and *Y* , while the traditional support measure indicates the simple frequency of occurrence of the combinations of items in the electronic basket dataset. The AW-sup value of *{X,Y}* is 2, meaning that, on average, for each electronic basket containing items *X* and *Y* both items have been rated at least 2. Ranking the mined item sets by decreasing AW-sup allows experts to consider first the combinations of items that got maximal average ratings. Note that this statistics cannot be straightforwardly computed based on simple averages, because (i) it considers only the electronic baskets containing both *X* and *Y* , (ii) for each basket it selects the rating of the least weighted item between *X* and *Y* . Item sets not satisfying the traditional support threshold are discarded because they represent combinations of items that rarely occur in the analyzed data. The setting of the minimum weighted support threshold is driven by the average rating of the selected items. More specifically, we are interested in exploring the frequent combinations of items with rating above average, i.e., the item sets whose AW-sup is above a minimumthreshold.

5. IMPLEMENTATION

The below steps gives an overview about weighted item set mining.

Algorithm

Step1: Collect the data sets
Step 2: Select any data from datasets
Step3: Stores datasets with weight (Rank or review)
Step4: Mapping part introduce in mining
Step5: Search parallel items in datasets.
Step6: In reducing part we can get result.

Fig. 3 Sign in module

Fig. 4 Sign up module

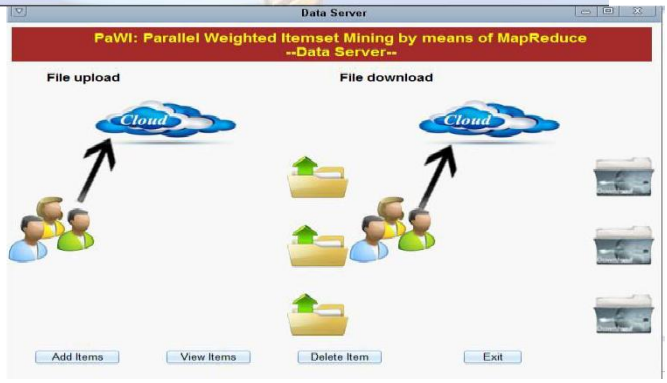

Fig. 5 Cloud module

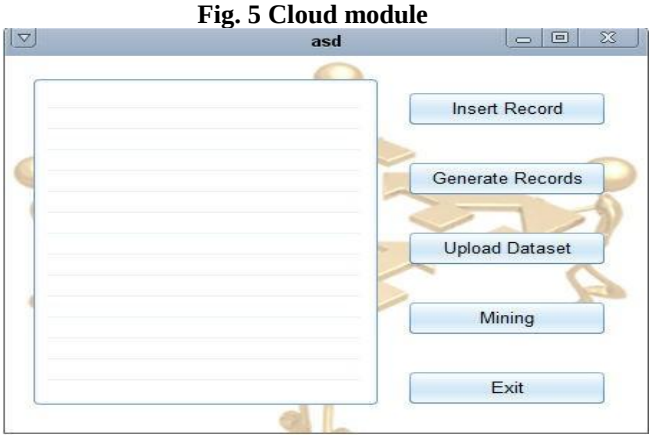

Fig. 6 User module

CONCLUSIONS AND FUTURE WORK

This paper presents a parallel and distributed solution to the problem of extracting frequent itemsets from Big Weighted Datasets. The proposed system, running on a Hadoop cluster, overcomes the limitations of state-of-theart approaches in coping with datasets enriched with item weights. The experiments, performed on a large volume of dataset, confirm the actionability of the mining result. Future works will entail the application of the proposed approach to recommender systems. For example, discovering combinations of items that were frequently bought together with an overall rating above average couldbeusefulfor recommending additional items beyond those already purchased by a given user.

REFERENCES

[1] D. Agrawal, S. Das, and A. El Abbadi, ―Big data and cloud computing: Current state and future opportunities,‖ in *Proceedings of the 14th International Conference onExtending Database Technology*, ser. EDBT/ICDT ‘11. New York, NY, USA: ACM, 2011, pp. 530–533.

[Online].Available:http://doi.acm.org/10.1145/1951365.1951432
[2] H. T. Lin and V. Honavar, ―Learning classifiers from chains of multiple interlinked RDF data stores,‖ in *IEEE InternationalCongress on Big Data, BigData Congress 2013, June 272013-July 2, 2013*, 2013, pp. 94–101. [Online]. Available: http://dx.doi.org/10.1109/BigData.Congress.2013.22
[3] J. Hartog, R. Delvalle, M. Govindaraju, and M. J. Lewis, ―Configuring a mapreduce framework for performance heterogeneous clusters,‖ in *2014 IEEE InternationalCongress on Big Data, Anchorage, AK, USA, June 27- July 2, 2014*, 2014, pp. 120–127.[Online].Available:http://dx.doi.org/10.1109/BigData.Congress.2014.2631
[4] ―The Apache Mahout Project: The Apache Mahout machine learning library. Available: http://mahout.apache.org/ Last access: March 2015,‖ 2015.
[5] ―MLlib: Apache Spark‘s scalable machine learning library. Available: http://spark.apache.org/mllib/ Last access: March 2015.
[6] R. Agrawal, T. Imielinski, and Swami, ―Mining association rules between sets of items in large databases,‖ in *ACMSIGMOD 1993*, 1993, pp. 207–216.
[7] L. Cagliero and P. Garza, ―Itemset generalization with cardinality-based constraints,‖ *Information Sciences*, vol. 244, pp. 161–174, 2013.
[8] E. Baralis, L. Cagliero, S. Jabeen, and A. Fiori, ―Multidocument summarization exploiting frequent itemsets,‖ in *Proceedings of the ACM Symposium on AppliedComputing, SAC 2012, Riva, Trento, Italy, March 26-30, 2012*, 2012, pp. 782–786. [Online].Available: http://doi.acm.org/10.1145/2245276.2245427
[9] R. Agrawal and R. Srikant, ―Fast algorithms for mining association rules in large databases,‖ *in VLDB’94, Proceedingsof 20th International Conference on Very Large DataBases*, J. B. Bocca, M. Jarke, and C. Zaniolo, Eds. Morgan Kaufmann, 1994, pp. 487–499.
[10] J. Han, J. Pei, and Y. Yin, ―Mining frequent patterns without candidate generation,‖ in *Proceedings of the 2000 ACMSIGMOD International Conference on Management of Data*, 2000, pp. 1–12.
[11] J. Neville, D. Jensen, and B. Gallagher, ―Simple estimators for relational bayesian classifiers,‖ in Proceedings of the Third IEEE International Conference on Data Mining, 2003, pp. 609–612.
[12] C. Perlich and F. Provost, ―Distribution-based aggregation for relational learning with identifier attributes,‖ Machine Learning, vol. 62, no. 1-2, pp. 65–105, Feb. 2006.
[13] A. McCallum and K. Nigam, ―A comparison of event models for naive bayes text classification,‖ in AAAI-98 Workshop on Learning for Text Categorization. AAAI Press, 1998, pp. 41–48.
[14] T. Ferguson, ―A bayesian analysis of some nonparametric problems,‖ The annals of statistics, pp. 209–230, 1973.
[15] T. P. Minka, ―Estimating a dirichlet distribution,‖ Tech. Rep., 2012.
[16] R. E. Madsen, D. Kauchak, and C. Elkan, ―Modeling word burstiness using the dirichlet distribution,‖ in Proceedings of the 22nd International Conference on Machine Learning, ser. ICML ‘05. New York, NY, USA: ACM, 2005, pp. 545–552.
[17] R. Gordon, R. Bender, and G. T. Herman, ―Algebraic reconstruction techniques (ART) for three-dimensional electron microscopy and x-ray photography,‖ Journal of Theoretical Biology, vol. 29, no. 3, pp. 471–481, 1970.
[18] M. Nickel, V. Tresp, and H.-P.Kriegel, ―Factorizing YAGO: scalable machine learning for linked data,‖ in Proceedings of the 21st international conference on World Wide Web, ser. WWW ‘12. New York, NY, USA: ACM, 2012, pp. 271–280.
[19] U. Losch, S. Bloehdorn, and A. Rettinger, ―Graph kernels for ¨ RDF data,‖ in The Semantic Web: Research and Applications, ser. Lecture Notes in Computer Science, E. Simperl, P. Cimiano, A. Polleres, O. Corcho, and V. Presutti, Eds. Springer Berlin Heidelberg, 2012, vol. 7295, pp. 134–148.