A Generalized Relationship Mining Method for Social Media Text Data

Tuhin Sharma[1] and Durga Toshniwal[1]

[1]Department of Computer Science and Engineering,
[1]Indian Institute of Technology Roorkee
Roorkee - 247667, Uttarakhand, India
{tuhinsharma121,durgatoshniwal}@gmail.com

Abstract. Increasing popularity of Social Media has resulted in the creation of a huge amount of user generated documents. A large number of research works have focused on inferring relationship in certain specific social network domains. Few have considered structured data to establish syntax based relationship. In this work, we develop a two-step syntax based and semantic based relationship mining approach. Here we generalize the concept of relationship mining for all structured as well as unstructured unsupervised text documents from all social network domains. At first, we choose suitable features from individual document and store them in graph structure. Then we establish relationships in the graph generated to obtain Reduced node Social Graph with Relationships (RSGR). Our empirical study on various social media document validates the effectiveness of our approach and suggests its generality in finding relationships irrespective of the type of text documents and the social network domains.

Keywords: Social network analysis, Relationship mining, Concept, Wordnet, Freebase, Social graph, Visualization.

1 Introduction

Web consists of billions of documents, with an increasing rate of growth of 7.3 million pages per day. A document could be a facebook post, a tweet, a blog, a review or even a video. With this rapid growth of information in the web, there is an important need for identifying the relationships between various documents, since all these documents are not useful unless manually read.

In the current scenario, every document is fed individually in social media analysis [1][2]. Here, no relationships exist between documents because every document is considered as a single isolated entity. So we have knowledge of every document individually but not in the presence of other documents. So certain queries cannot be answered like who is the most active Author who talks about data mining, what are the other similar documents the Author is talking about. So, if we can introduce some relationship links or edges we can find out the importance of certain documents as well as we shall be able to give corresponding output according to such queries.

In this work, we develop a hybrid of syntax and semantic based relationship mining approach for all structured as well as unstructured unsupervised text documents from all social network domains. At first, suitable features are selected and extracted from individual documents and are stored in the form of graph structure. In the second step we establish relationships in the context of the graph generated and merge duplicate nodes. The relationships are represented in the form of edges. The strengths or weights of the relationships represented in numerical form are also stored as part of those relationship edges. Thus finally we obtain a Reduced node Social Graph with Relationships (RSGR).

2 Related Work

Earlier research works in relationship mining mainly focus on applying text mining technique on structured data. Bonaventura Coppola et al. propose a machine learning framework to mine relations automatically, where the target objects are structurally organized in a tree[3]. But the approach is syntax based and semantic meaning related to the extracted dimensions is not used.

Current solutions for social relationship mining are generally statistical learning-based approaches[4]. Christopher P. Diehl et al. propose a supervised learning model, which treats links as features and labeled pairs as training data[5]. Wen-bin Tang et al. proposed a Partially-labeled Pairwise Factor Graph Model to infer the type of social ties[6]. But only advisor-advisee, manager-subordinate relations and friendships from mobile network are mined for domain specific data. Tang et al. propose a transfer-based factor graph to incorporate social theories into a semi-supervised learning framework[7]. Lei Tang and Huan Liu propose a clustering-based approach which differentiates latent social dimensions from the social network[8]. But the challenges in social media analysis are to handle the dynamic nature of social network and its multiple entities[9]. Each day, huge number of new members join the social network as well as new connections occur among the existing members. How to efficiently update the relationship model accordingly remains a challenge.

Chi Wang et al. develop a time-constrained probabilistic factor graph model (TPFG) that takes a research publication network as input to find advisor-advisee relationships[10]. But the problem is, how to generalize the approach of relationship mining to enable semi-supervised learning and to cope with multi-typed nodes and links are not done yet.

In this work, we have done the following innovations. We establish relationship between different Authors of different documents on the basis of common interest and user-id. For Domain we establish similarity on the basis of popularity. We have connected different documents on the basis of common topic and reference to other documents. We even connect a document to other Domains apart from its source Domain. The next sections describe the work we have done. In section 3. the proposed work is elaborately discussed. Section 4 represents experimental results along with different possible applications to support the effectiveness of our work.

3 Proposed Work

The flow chart for the proposed work is shown in Fig.1.

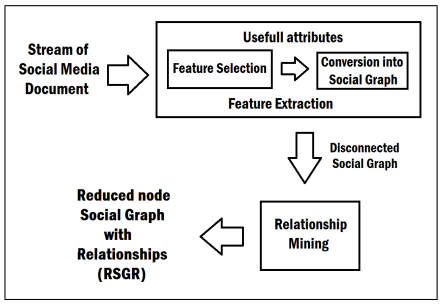


Fig. 1: Proposed process for Relationship Mining

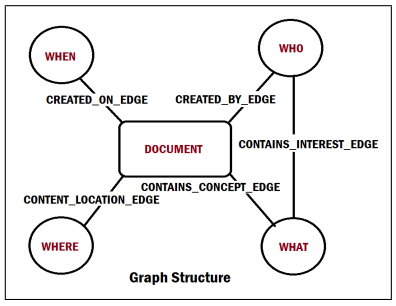


Fig. 2: Each document converted into graph structure

3.1 Feature extraction

Each of the incoming documents can be categorized into 4 types which are Boards (e.g. large blogs, facebook posts etc.), Reviews (e.g. tripadvisor review), Microblogs (e.g. tweets) and Videos (e.g. youtube, dailymotion videos). Each of them has 4 types of information regarding its Author, Timestamp, Domain and Text associated with it. So for individual documents we create nodes corresponding to these 4 types and we segregate corresponding information and store them accordingly in graph structure as shown in Fig.2. We additionally create WHAT nodes to store different concepts or keywords of text documents and to store interest of Authors.

The edges shown are not assigned any weight. For DOCUMENT, WHO, WHERE and WHEN nodes we just read and store the attribute values for respective documents. For the creation of WHAT nodes we perform natural language processing. The significance of these nodes is as follows:-

DOCUMENT node: It keeps track of the actual document. It contains the *Url* of the document, the *Type* of the document, i.e. whether it is a Boards, Microblogs, Reviews, Videos, etc (categorized by us). It also contains *Texthtml, Subjecthtml, Review rating*, etc.

WHO node: It contains information about the Author who has created the document like *Real name, Username, Location, Gender, Profile-url,Author description*, etc.

WHERE node: It contains information about the place where the document has been created like *Domain, Geo-location*, etc. Additionally, we store the Pagerank score. We have used Quantcast dataset[15] for determining the Pagerank score of a particular domain. Pagerank score of the WHERE node is calculated as the difference between the pagerank of the domain and the lowest possible rank in the Quantcast dataset.

WHEN node: It contains information about timestamp like *date of creation* of the document, *date of registration* of the Author etc.

WHAT node: These are basically concepts. Concepts are important keywords which solely determine the meaning of a document or the interest of an Author. The concepts are fetched from unstructured fields which can be either the *Texthtml* and *Subjecthtml* or the *Author description*. We do not use classical TF-IDF measure while mining concepts as it is more appropriate when we need to eliminate frequently occurring stopwords. But we deal with documents irrespective of their category. So it may frequently happen that an important concept (not a stopword) occurs in every or most of the documents. In these cases, if we take TF-IDF measure then that concept will be lost and we shall encounter error. So, we use Apache Opennlp library[14] for mining concepts. From corresponding fields, plain text data is extracted using HTML parser. The resulting plain text is tagged using opennlp POS tagger[14]. Then the plain text is tokenized and stopwords and punctuations are removed from those respective tokens. Here we store the top 10 concepts out of which at most 3 concepts will be proper nouns according to the frequency of occurrence in respective documents. If there exist two different concepts having the same term frequency then first we select the concept which occurred first in the text and then we select the other concept. We are not eliminating any verbs or adverbs because even they can correspond to a concept. For example, *I love to* ***swim (verb)*** *for long time.* All the concepts are stemmed using Porter stemmer library included in [14]. Then the stemmed and the actual concepts are stored in the corresponding WHAT nodes. The WHAT nodes mined from *Texthtml* or *Subjecthtml* are connected with the DOCUMENT node and those mined from *Author description* are connected with the corresponding WHO node as shown in Fig.2. So, in this step we create atomost 10 WHAT nodes for every document and each of them contains a single concept along with its stemmed form.

3.2 Relationship Mining

Depending on the type of nodes different approach is followed to establish relationships and to remove duplicates.

Approach for WHO nodes For establishing relationships between WHO nodes 2 types of relations are considered as shown in Fig.3. They are as follows:

Explicit relationship. It represents whether one Author is a friend or follower of another Author in social media like Facebook and Twitter. To mine this relation facebook api and twitter api have been used. It is called explicit as it is explicitly defined by the social network provider. We connect such WHO nodes via EXPLICIT_RELATIONSHIP_EDGE(ERE).

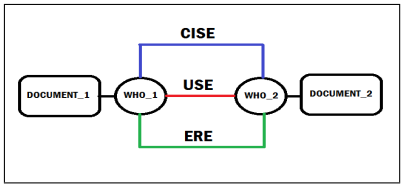


Fig. 3: Relationship between WHOs

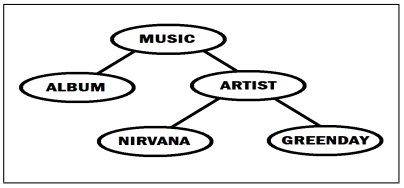


Fig. 4: MUSIC domain's Ontology tree

Common interest relationship It represents whether two Authors have any similar interests. To mine this relation the WHAT nodes associated with one WHO node are compared to those associated with the other WHO node using Wordnet[16] and Freebase[17]. The reason for using the Wordnet dictionary is to figure out the similarity between two WHAT nodes on the basis of synonym and hyponym-hypernym relations. The reason for the use of Freebase is to figure out the similarity between WHAT nodes which belong to the same category. If the concepts are Proper nouns then we cannot find synonyms. For example, let WHAT_1 and WHAT_2 represent the interests *Nirvana* and *Greenday* respectively. Here, there exists a relationship between WHAT_1 and WHAT_2 because both the interests are related to the music category. So to detect this type of relationship we use Freebase. For WHAT_1 as well as for WHAT_2 we shall get an ontology path starting from the root node to the desired node in the ontology tree of Freebase. Now, if these two paths share more common nodes then they will be more similar.

Algorithm 1 Algorithm for creating CISE

INPUT: Disconnected social graph
OUTPUT: Social graph with CISE

for all pair of WHO nodes **do**
  for all pair of WHAT nodes containing Proper noun concepts i.e. one WHAT node associated with one WHO node and other WHAT node associated with the other one **do**
    Calculate the Wu-Palmer similarity (WPS) value according to Freebase
  end for
  Say,there are such m pairs of Proper noun concepts
  for all WHAT nodes containing concepts other than Proper noun associated with any one of the WHO nodes **do**
    synonyms from Wordnet are fetched for each actual concept and stored in respective WHAT nodes and associated with those actual concepts
    for each of the WHAT nodes associated with the other WHO node **do**
      Compare actual concept against each of the actual concepts from the other set and against the synonyms associated using string comparison
      Compare stemmed concept against each of the stemmed concepts from the other set using string comparison
    end for
  end for
  Say,there are such n matched concepts which are not Proper nouns
  $match\_value = \sum_{i=1}^{m} WPS_i + n$
  $CISV = match\_value/avg\_length.$
  if $CISV > (THRESHOLD\_CISV\_CONNECT = 0.5)$ **then**
    Connect the two WHO nodes by a CISE and store $CISV$ in it
  end if
end for

For example, let us consider the ontology tree for the Music domain as per Fig.4. Here, there exists a relationship between *Album* and *Artist* as well as between *Nirvana* and *Greenday*. But *Nirvana* and *Greenday* are more similar as

compared to *Album* and *Artist*. It is because *Nirvana* and *Greenday* appear at greater depths in the ontology tree as compared to that of *Album* and *Artist*. So additionally we have to consider the depth in the ontology tree also. For this, we use Wu-Palmer similarity measure[11] while considering the similarity based on Freebase. The algorithm for calculating common interest similarity value ($CISV$) and creating COMMON_INTEREST_SIMILARITY_EDGE (CISE) is depicted in Algorithm.1. Here, avg_length is the total number of WHAT nodes associated with those two WHO nodes divided by 2.

User-id similarity relationship. It represents whether two Authors are similar or not on the basis of their user-ids. To mine this relation 5 fields i.e. *Username, Gender, Author description, Location* and *Original name* are compared. Generally when a person gets registered in two different social sites he usually chooses Usernames which starts with a matching subsequence of characters (eg. *Alex Maxwell* is present in facebook as *alexm* and in twitter as *alexmax*). To capture this type of similarity both the number of characters matched and the length of starting subsequence matched are to be considered. So, for *Username* Jaro-Winkler distance[12] is used. For *Gender, Location* and *Original name* Jaccard coefficient[13] is calculated to find out the similarity. For *Author description* we use $CISV$. The priorities of the similarity values corresponding to these 5 fields in descending order are as follows:-

Gender>Location>Author description>Original name>Username.

Algorithm 2 Algorithm for creating USE

INPUT: Disconnected social graph
OUTPUT: Social graph with USE

for all pair of WHO nodes **do**
  Calculate $S_i \forall\ 1 \leq i \leq m$
  $W_i = \frac{m-i+1}{\sum_{i=1}^{m} i}$
  $USV = \sum_{i=1}^{m} W_i S_i$
  if $USV > (THRESHOLD\_USE\_CONNECT = 0.8)$ **then**
    Connect the two WHO nodes by a USE and store USV in it
  end if
end for

The reason for such priority ordering is very simple. If two WHO nodes have different gender then irrespective of other fields they cannot be the same person. If gender is same for two WHO nodes but they belong to two different countries then they cannot be the same person and so on. So, the similarity due to *Gender* should get highest weightage and that due to *Username* should get the lowest weightage when we use those similarity values to calculate the user-id similarity value. As we are considering the intra social network relationship as well as inter social network relationship so it is quite often that we find some attributes are applicable to certain documents and some are not. As for example, for a *facebook* document we shall find an attribute named *number of friends*, but in case of *tripadvisor* document we cannot find such attribute. So, while calculating the similarity we shall consider only those attributes for which the values are present in both of the documents.

Let, A_i represents the i^{th} attribute and there are m number of attributes for which values are present in both WHO nodes. The priorities in descending order are as shown in Eq.1.

$$A_1 > A_2 > ..... > A_m[1 \leq m \leq 5] \quad (1)$$

For each of the attributes we get respective the similarity values (S_i) and we assign weights W_i to those S_i to calculate userid similarity value (USV). The process to calculate USV and to establish USERID_SIMILARITY_EDGE (USE) is described in Algorithm.2.

Approach for WHERE nodes For establishing relationships between WHERE nodes we create PAGERANK_SCORE_SIMILARITY_EDGE (PSSE) based on pagerank score as shown in Fig.5. It is described in Algorithm.3.

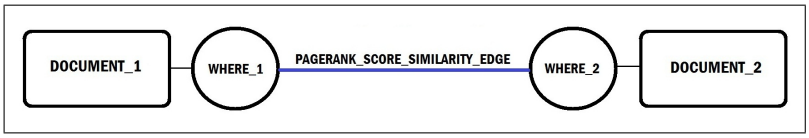


Fig. 5: Relationship between WHERE nodes

Algorithm 3 Algorithm for creating PSSE

INPUT: Disconnected social graph
OUTPUT: Social graph with PSSE

```
for all pair of WHERE nodes do
    Calculate diff as the difference between pagerank scores of those 2 WHERE nodes
    if diff > (THRESHOLD_PSSE_CONNECT = 50) then
        Connect the two WHERE nodes by a PSSE
    end if
end for
```

Approach for DOCUMENT nodes For establishing relationships between DOCUMENT nodes 2 types of relations are considered as shown in Fig.6. They are as follows:

Document Concept similarity relationship We check if two documents have any similar concepts. To mine this relation Algorithm 1 is followed except we consider DOCUMENT nodes instead of WHO nodes and we calculate document concept similarity value (DCSV) and store them in DOCUMENT_CONCEPT_SIMILARITY_EDGE (DCSE) connecting the two DOCUMENT nodes.

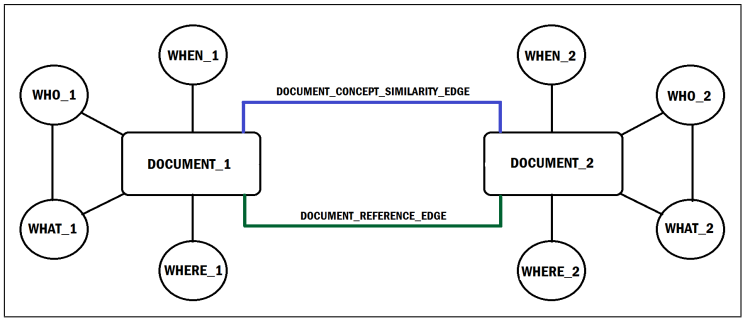


Fig. 6: Relationship between DOCUMENT nodes

Document Reference relationship It represents if a DOCUMENT is referred by another DOCUMENT. To find this relationship we parse the *Texthtml* using html parser and find the links to other document if present in it. If we find any such document then we connect those documents using DOCUMENT_REFERENCE_EDGE (DRE).

Relationship between DOCUMENT node and WHERE node We establish 1 type of relation between DOCUMENT and WHERE node other than CONTENT_LOCATION_EDGE as shown in Fig.7.

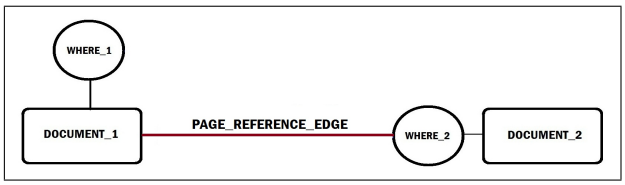


Fig. 7: Relationship between DOCUMENT and WHERE nodes

This relationship represents the domain or the WHERE node from which the documents referred in the current document comes from. We analyze the *Texthtml* of the DOCUMENT and find out different domain names and connect those corresponding WHERE nodes with the DOCUMENT node via PAGE_REFERENCE_EDGE (PRE). So this type of edge shows the other domains a particular document is connected to apart from the source domain.

After all these steps, finally we get the Reduced node Social Graph with Relationships (RSGR) as shown in Fig.8.

4 Experimental Results

In this section we show various experiments that support the effectiveness and the accuracy of our approach.

4.1 Experimental Setup

Data Sets. We use real life data from various social sites, especially from facebook, twitter, tripadvisor, amazon, youtube, dailymotion, hotel.com etc. It consists of $60,370$ unsupervised documents. After the feature extraction step the resulting graph database consists of $8,01,350$ nodes. For the purpose of experimental analysis we take a random sample of 600 documents (resulted into 8229 nodes) consisting of 150 documents from each category, i.e. Boards, Microblogs, Reviews and Videos. These 4 types of categories are associated with the documents by us to explain the experimental results. To test the accuracy of the different types of discovered relationships, we adopt two data sets. In the first one (D_1), those selected 600 documents are manually labeled by looking into the actual document in which the best concepts of the *Texthtml* and the *Author description* are selected. In the other one (D_2), the same process of labeling is adopted, but only top 10 concepts are chosen for the documents. In both the cases, the labeling i.e. selecting the concepts is done by 10 unbiased persons. These persons are not expert in any domain and as the data used also do not belong to any specific category, we assume the labeling as the ground truth.

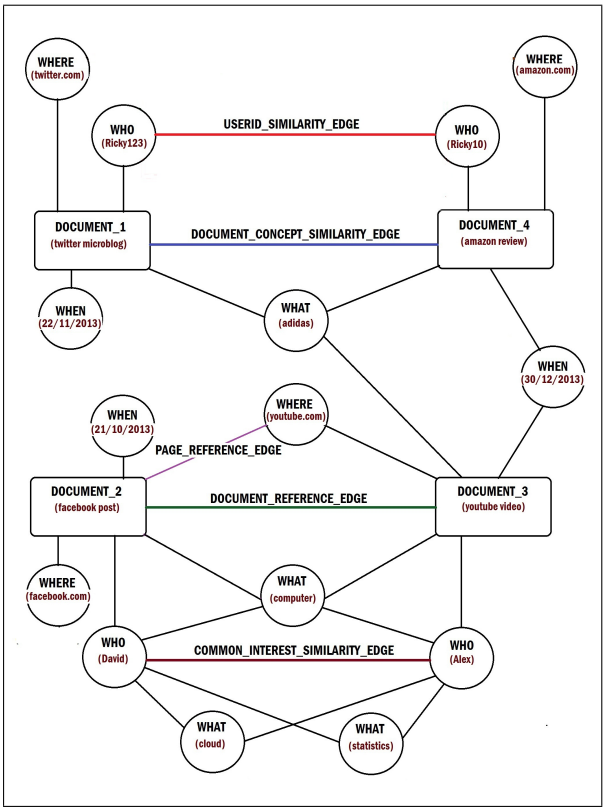


Fig. 8: Reduced node Social Graph with Relationships (RSGR)

Method At first we try to find out the behavior of accuracy and the redundancy of our algorithm for finding the concepts with respect to D_1 and D_2. The accuracy and redundancy is calculated as follows:-

Let us consider a document for which, A represents the set of concepts that we have got applying our algorithm and B represents the set of concepts that we have got from the labeled dataset. $N(A)$ and $N(B)$ represent the number of concepts in the sets A and B respectively. Then the accuracy and redundancy are calculated as per Eq.2 and Eq.3.

$$Accuracy = \frac{N\left(A \cap B\right)}{N\left(B\right)} \tag{2}$$

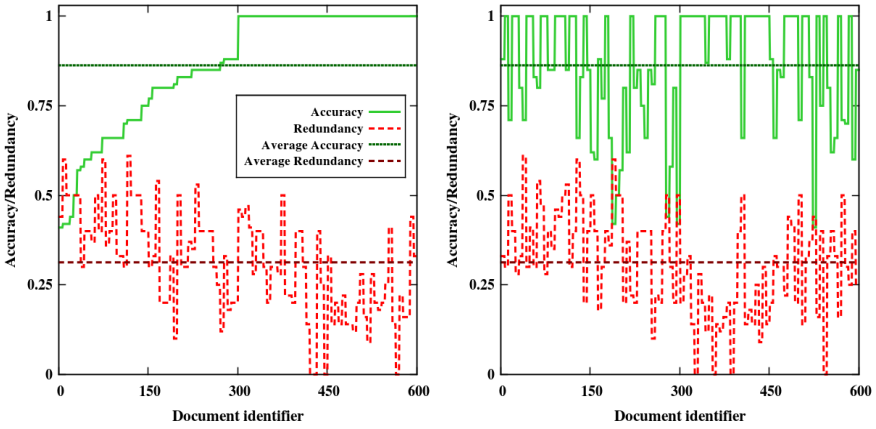
$$Redundancy = \frac{N\left(A - B\right)}{N\left(A\right)} \tag{3}$$

After that we try to compare the relationship edges that we get by applying our algorithm on the concepts that we have found in the feature extraction step with the relationship edges obtained by applying our algorithm on the concepts that we get from the labeled dataset D_1 and D_2. The accuracy and redundancy for the generated relationship edges are calculated as per Eq.2 and Eq.3 where

A represents the set of edges generated by our algorithm using our algorithm generated concepts and B represents the set of edges generated by our algorithm using concepts as per D_1 or D_2.

4.2 Accuracy and Redundancy of Concepts.

The accuracy corresponding to each of the documents are sorted in increasing order and both the corresponding accuracy and the redundancy with respect to D_1 are plotted against the respective documents in Fig.9(a). The same thing is done with respect to D_2 and the corresponding result is plotted in Fig.10(a). In Fig.9(b) and in Fig.10(b) the documents are sorted according to their respective category, where indices from (1-150), (151-300), (301-450) and (451-600) represent document types Reviews, Boards, Microblogs and Videos respectively.



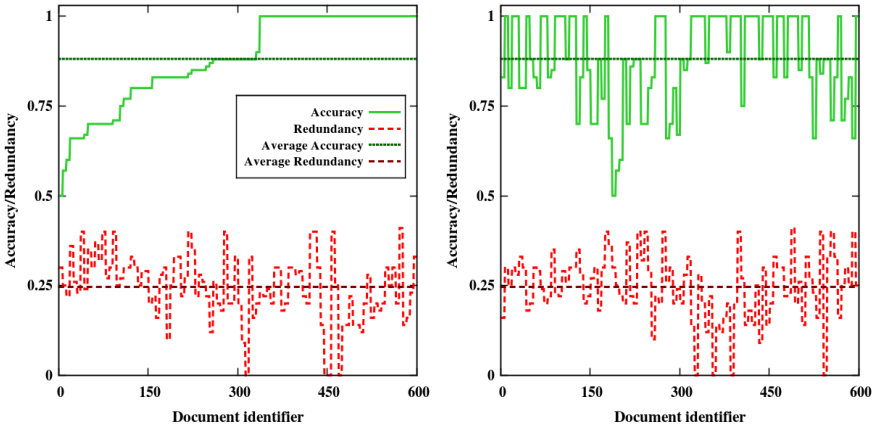
(a) (b)

Fig. 9: Accuracy vs. Redundancy for D_1 (a) documents are sorted according to their Accuracy; (b) documents are sorted according to their Types.

In Fig.9(a) and Fig.9(b) the *average accuracy* is 0.868 and *average redundancy* is 0.327 (Corresponding to D_1). In Fig.9(a) we see that for high accuracy, the encountered redundancy is pretty much low, which is around 0.28 and for low accuracy the corresponding redundancy is much higher. Now if we look at Fig.9(b) we see that for Reviews the encountered redundancy is higher than mean redundancy, but still we get much better accuracy. In case of Boards, the redundancy is similarly higher than the mean but the accuracy is very poor. For Microblogs and Videos the accuracy is very good and encountered redundancy is much lower than the mean redundancy. Actually in case of Microblogs and Videos, the text portion is short so after removing the stopwords, punctuations and after counting frequency of the resultant concepts we get almost accurate results. But in case of Reviews and Boards, the text portion is much big so we are encountering redundancy more than the *average redundancy* and getting accuracy less than the *average accuracy*.

In Fig.10(a) and Fig.10(b) the *average accuracy* is 0.891 and *average redundancy* is 0.246 (Corresponding to D_2). In this case we see that the redundancy

does not vary much and more or less follows the average redundancy value. In Fig.10(b), in case of Boards we see that the accuracy is improved and the redundancy is much lower. So from this, we can infer that if we restrict the user to choose only top 10 concepts, then they are much similar as those obtained by applying our algorithm. This justifies the effectiveness of our approach to find out the concepts or to create WHAT nodes.



(a) (b)

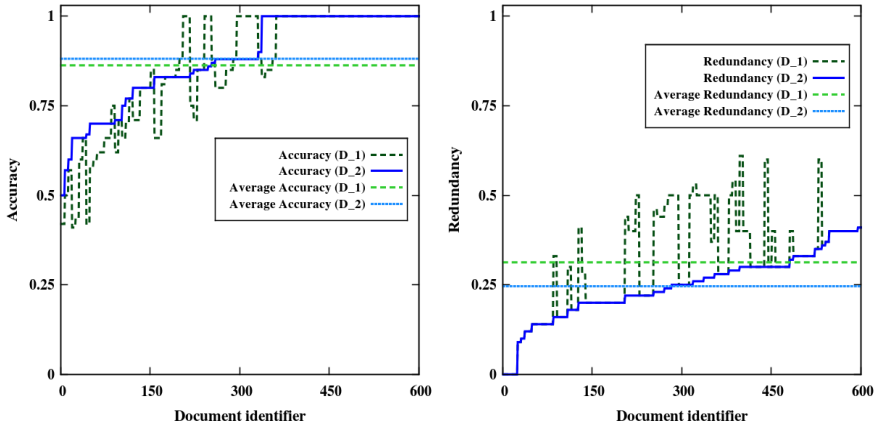
Fig. 10: Accuracy vs. Redundancy for D_2 (a) documents are sorted according to their Accuracy; (b) documents are sorted according to their Types.

In Fig.11(a) the accuracy and in Fig.11(b) the redundancy corresponding to D_1 and D_2 are plotted against the respective documents. We see that the accuracy and the redundancy corresponding to D_2 is much better than those corresponding to D_1. The reason is, in case of D_1 all possible best concepts are chosen. So sometimes there may be more than 10 concepts in case of D_1. But in case of D_2 there exists at most 10 concepts. In our algorithm, we also calculate the top 10 concepts. So the probability of getting better accuracy and redundancy is higher while compared to D_2 than D_1.

In Fig.11(a) and in Fig.11(b) for some documents the accuracy is lower and the redundancy is higher corresponding to D_2 than those corresponding to D_1. The reason is, for those documents, the concepts that we get using our algorithm are of medium importance. For example:-

Let for a document the set of concepts corresponding to D_1 is {*coke, football, basket, drink, match, ground, India, Pakistan, exciting, television, Messi, Ronaldo*} and corresponding to D_2 it is {*coke, football, basket, drink, match, ground, India, Pakistan, exciting, television*}. If a concept occurs before another then it is more important than the other. So, *coke* is more important than *football*. According to our algorithm we get concepts {*coke, football, basket, drink, match, ground, amazing, India, Ronaldo, Messi*}. So, the accuracy corresponding to D_1 becomes $9/12 = 0.75$ which is greater than the accuracy corresponding to D_2 which is $7/10 = 0.7$. Also the redundancy corresponding to D_1 becomes $1/10 = 0.10$ which is lower compared to that of D_2 which is $3/10 = 0.3$. So

we see that $Messi, Ronaldo, amazing$ are not top concepts rather concepts with medium importance. So, we get this type of result.



(a) (b)

Fig. 11: (a) Accuracy corresponding to D_1 vs. Accuracy corresponding to D_2; (b) Redundancy corresponding to D_1 vs. Redundancy correspondingto D_2.

4.3 Accuracy and Redundancy of Relationship Edges

The frequency distribution of edges is shown in Fig.12. For each edge type we have got 3 edge counts. The left most of them corresponds to the edge count when the concepts generated by our algorithm are used (Say C_1). The other two of them corresponds to the edge count when the concepts as per D_1 and D_2 are used respectively (Say C_2 and C_3). We can see that for edge type corresponding to userid (USE), pagerank (PSSE), document reference (DRE) and page reference (PRE) all the 3 counts are same. For edge type CISE corresponding to common interest we see that the difference between C_1 and C_2 as well as C_1 and C_3 are not much. It is because for *Author Description* we get almost same set of concepts either applying our algorithm or using human perception as it consists of limited words. But for edge type DCSE corresponding to document concept similarity we see that the difference between C_1 and C_2 as well as C_1 and C_3 are greater than or equal to 10. It is because for some documents having large text field we lose some concepts and we are not getting relationship edges corresponding to those concepts.

Table 1: Accuracy and Redundancy of the edges

Basis of edges	Edge count			Accuracy(A)			Redundancy(R)		
	C_1	C_2	C_3	w.r.t D_1	w.r.t D_2	Average value	w.r.t D_1	w.r.t D_2	Average value
USE	4	4	4	1	1	1	0	0	0
CISE	56	65	62	0.83	0.87	0.85	0.035	0.035	0.035
PSSE	24	24	24	1	1	1	0	0	0
DRE	63	63	63	1	1	1	0	0	0
DCSE	85	113	101	0.707	0.801	0.754	0.058	0.047	0.052
PRE	58	58	58	1	1	1	0	0	0

Table 1 shows the types of relationship edges and their corresponding accuracies and redundancies. We see that the accuracy and the redundancy of edges corresponding to PSSE, DRE and PRE are 1 and 0 respectively. The reason is, in those cases we simply establish the syntactical relationship and it does not depend on the concepts a document contains. So there is no possibility of committing error or any type of redundancy in case of those 3 types of relationship edges. We can see that 4 pairs of Authors or WHO nodes are similar in terms of userid. We get this result for both of the two datasets. The accuracy of CISE is very high as that compared to DCSE. The reason is in case of Authors interest we get the concepts more accurately. So, the *average weighted accuracy* and the *average weighted redundancy* of our algorithm for relationship mining are $\frac{4\times1+56\times0.85+24\times1+63\times1+85\times0.801+58\times1}{4+56+24+63+85+58} = 0.9127$ and $\frac{4\times0+56\times0.035+24\times0+63\times0+85\times0.052+58\times0}{4+56+24+63+85+58} = 0.022$. So, we see that for syntactical relationship edges, we get very good result and for semantic relationship edges, we encounter some error as well as redundancy. Yet, the best part is we may not generate all possible relationship edges, but we get negligible false relationship edges because the redundancy value is very low.

4.4 Scalibility

Let, n be the size of the RSGR generated. As we are comparing every node with each other of a particular type so the running time is $\mathcal{O}(n^2)$. After that documents come in small chunks relative to the RSGR database. Let, the size of these chunks of documents be m $[m << n]$. But now the nodes stored in the RSGR database are not compared against each other rather the nodes obtained from these small chunks of data of size m are compared against the RSGR database. So now the cost of adding these new chunks of documents is $\mathcal{O}(mn)$. Now due to addition of new documents as the size of the RSGR increases, at some point m becomes much smaller as compared to n. So, now the cost of adding new documents becomes $\mathcal{O}(mn) \approx \mathcal{O}(n)[\because m << n]$.

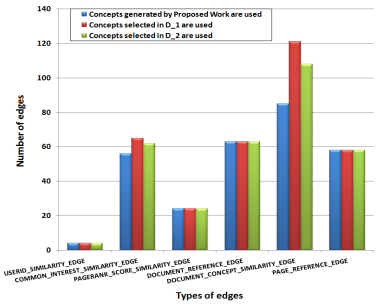


Fig. 12: Frequency of Edges

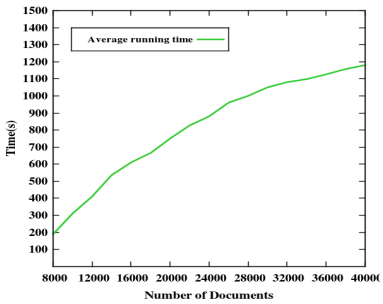
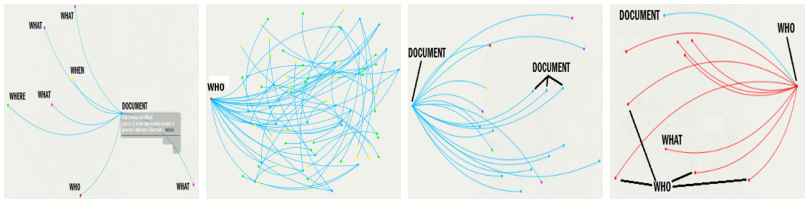


Fig. 13: Average Computation Time

So, in Fig 13 we are adding 2000 documents at a time to the RSGR with an initial size of 8000 documents. Here, we see that it takes significant time to build the RSGR database when the number of documents is low. But, once The RSGR is built the cost of adding different documents to it becomes very less. So, we can claim that RSGR is scalable.

4.5 Applications

We use TitanGraph[18] to store the incoming documents in graph structure as discussed and we maintain the graph database and all its relationships. Whenever new documents arrive we establish new connections among themselves as well as with the existing documents. So the RSGR serves as a source of knowledge. The established relationships can benefit many applications like query regarding highly active Authors, the most popular topics discussed, similar documents, similar Authors, etc. We use SigmaJs[19] to visualize the RSGR generated. Some parts of the RSGR are shown in Fig.14.



(a) (b) (c) (d)

Fig. 14: (a)An isolated Document; (b)Most active Author; (c)Document concept similarity edge; (d)Common interest similarity edge

We use different colors representing different types of nodes and relationships. Red, Blue, Green, Yellow, Pink colors represent WHO, DOCUMENT, WHERE, WHEN and WHAT nodes respectively. A single isolated document is shown in Fig.14(a). Fig.14(b) represents the most active Author and different documents created by him. Relationship edges are also colored. Blue represents the document concept similarity edges associated with DOCUMENT and Red represents Common interest relationship edges associated with WHO nodes as shown in Fig.14(c) and Fig.14(d).

5 Conclusion

We have studied mining of different types of relationships from real life online documents to discover syntactical as well as semantic relationships. We propose a two stage framework to mine relationships step by step until we get hierarchies of Authors, Domains, Time and as well as Concepts. We have used Wordnet and Freebase for this purpose along with different types of api and Quantcast data and to the best of our knowledge it is the first attempt to use them for mining social relationship. We propose a Reduced node Social Graph with Relationships (RSGR) model to establish and store different types of relationships of different social media documents. In this graph we can keep track of relationships of existing documents along with new documents which are constantly added irrespective of their source domain without encountering any redundancy.

Based on the result, we observe for Microblogs and Videos especially for the documents with limited sized text, our approach works well. Interesting problems related to our approach could be how to get a more effective topic mining model irrespective of the size of the document. Another scope is we can use this RSGR to find the credibility or trust of different documents. Clearly much more can be exploited from the knowledge base that we have generated.

Acknowledgment

This research is partially done in IBM's Extreme Blue Lab with Kiran Subbaraman. We thank the anonymous reviewer for labeling the documents.

References

1. Taboada, Maite, Julian Brooke, Milan Tofiloski, Kimberly Voll, and Manfred Stede. *Lexicon-based methods for sentiment analysis*, Computational linguistics 37, no. 2 pp: 267-307. (2011)
2. Pang, Bo, Lillian Lee, and Shivakumar Vaithyanathan. *Thumbs up?: sentiment classification using machine learning techniques*, In Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10, pp. 79-86. Association for Computational Linguistics, (2002)
3. B. Coppola, A. Moschitti, and D.Pighin. *Generalized framework for syntax-based relationship mining*, In ICDM, pages 153-162, (2008)
4. L. Getoor and B. Taskar. *Introduction to Statistical Relational Learning (Adaptive Computation and Machine Learning).* The MIT Press, (2007)
5. C. P. Diehl, G. Namata, and L. Getoor. *Relationship identifcation for social network discovery.* In AAAI'07, pages 546-552. AAAI Press, (2007)
6. Tang W B, Zhuang H L, Tang J. *Learning to infer social ties in large networks.* In Proc. ECML/PKDD 2011, Athens, Greece, September 5-9, pp.381-397, (2011)
7. Tang J, Lou T C, Kleinberg J. *Inferring social ties across heterogeneous networks.* In Proc. the 5th ACM Int. Conference on Web Search and Data Mining (WSDM 2012), Seattle, Washington, February 8-12, pages 743-752, (2012)
8. Lei Tang, Huan Liu. *Relational learning via latent social dimensions.* KDD '09 Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM New York, NY, USA, Pages 817-826, (2009)
9. L. Tang, H. Liu, J. Zhang, and Z. Nazeri. *Community evolution in dynamic multimode networks.* In KDD'08: Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery anddata mining, pages 677-685, (2008)
10. Chi Wang, Jiawei Han, Yuntao Jia, Jie Tang, Duo Zhang, Yintao Yu, Jingyi Guo. *Mining advisor-advisee relationships from research publication networks.* KDD 2010: 203-212. (2010)
11. Wu, Z., Palmer, M. *Verb semantics and lexical selection.* Proceedings of the 32nd An-nual Meeting of the Associations for Computational Linguistics. pages 133-138, (1994)
12. Winkler, W. E. *String Comparator Metrics and Enhanced Decision Rules in the Fellegi-Sunter Model of Record Linkage.* Proceedings of the Section on Survey Research Methods (American Statistical Association): pages 354-359, (1990)
13. Jaccard, P. *Distribution de la flore alpine dans le bassin des Dranses et dans quelques rgions voisines.* Bulletin de la Socit Vaudoise des Sciences Naturelles 37, pages 241-272, (1901)
14. http://opennlp.apache.org/
15. https://www.quantcast.com/top-sites
16. http://wordnet.princeton.edu/
17. http://www.freebase.com/
18. http://thinkaurelius.github.io/titan/
19. http://sigmajs.org/