Using Context to find Center-Piece Subgraph on Directed Multigraphs for Biomedical Literature

CS 224W Project Milestone

Group ID: 25

Team Members:

Sr. No.	Name	University ID Number	SUNet ID	Stanford Email
1.	Abhik Lahiri	05663100	alahiri	alahiri@stanford.edu
2.	Rifat Reza Joyee	05667070	joyee	joyee@stanford.edu
3.	Pranav Khaitan	05592143	pranavkh	pranavkh@stanford.edu

Abstract: Discovering patterns in RDF graphs, extracted from text has long been an area of interest. There are many instances, where we would like to know that given a set of nodes in an RDF graph, which is the subgraph that best connects this set of nodes. In this project, we aim to solve this problem of finding a Center-Piece Subgraph on Directed Multigraphs (specifically an RDF graph extracted from the text in radiology reports) and include the notion of context in our solution. This is a step towards the goal of the Semantic Web –converting all the information available online into a form that is computable by machines. We implement our ideas on text from radiology reports obtained from the Radiology Department at Stanford University, courtesy Prof. Daniel Rubin with the hope that our analysis shall have immense applications in the biomedical industry and further research in biomedical informatics, and shall also be a step towards automatic diagnosis.

1. INTRODUCTION:

As we mentioned in our project proposal, and as defined by the authors of [1], by extracting a Center-Piece Subgraph (CEPS) from a large graph, we mean to solve the following problem – "Given Q query nodes in a network, and a parameter K, find the node(s) and the resulting subgraph, that have strong connections to at least k of the Q query nodes". Such kind of a query is also called k_softAND query (which generalizes the case of the AND query, where k=|Q| and the OR query, where k=1). Again, as we mentioned in our project proposal, [1] proposes a novel algorithm to solve the above problem of extracting a Center-Piece Subgraph from an undirected weighted graph and is a generalization of the problem solved in [2], where the AND query problem is solved, but for only 2 query nodes. Moreover, as mentioned in our project proposal, [3] discusses the problem of finding a connection subgraph (i.e. CEPS on 2 query nodes - the same problem mentioned in [2]) on Multi-Relational Graphs (RDF graphs constructed by extracting subject-object-predicate triples from text). In [3], the authors use heuristics based on edge weighting mechanisms from the edge semantics suggested by the RDF schema to generate a "good" subgraph that connects the 2 query nodes (where by a "good" subgraph, we mean a subgraph that has a strong connection to our query nodes). However, the problem of extracting a Center-Piece Subgraph, that satisfies the k_softAND query to a graph, for a direct Mutligraph hasn't been tried as yet.

In this project, we try to perform Center-Piece Subgraph on an RDF graph (for the k_softAND query as described in [1]) generated by extracting subject-object-predicate triples from radiology reports. We then also include the concept of *Context* in our analysis of CEPS, and present results to compare how extracting CEPS with a certain *Context* affects the result. The only existing work that uses context is [5].The rest of the report is organized as follows - In Section 2 of this report, we describe in detail the mechanism by which we generate an RDF graph from radiology report text. Section 3 explains the method we used to perform Center-Piece Subgraph on this graph, Section 4 describes in detail what we mean by *Context* in our anaylsis, and how the our results are affected when we extract the CEPS from the RDF, given a certain *Context*. Section 5 presents the results that we have obtained so far and Section 6 presents our conclusion of the work based on our current results and what more can be done in the future. Finally, Section 7 gives a list of the References.

2. MECHANISM TO GENERATE RDF GRAPH:

Before we delve into the description of how we generated an RDF graph from biomedical literature, let us first describe what an RDF graph is. The RDF (Resource Description Framework) is a family of World Wide Web Consortium (W3C) specifications, that is widely used to describe the semantic content of web pages and text corpus in a machine readable form (a form in which programs can read and understand the content in the text). In the RDF model we make statements about text in the form of subject-object-predicate triples (known as triples in RDF terminology) and construct a labeled directed multi-graph from these triples where each subject node is connected to an object node via a directed edge that is labeled with the predicate value. Each phrase that constitutes the subject, object or predicate part of the sentence has a 'head' word and 'modifiers' or 'attributes'. In case of compound entities, we may have an attribute that is in turn the head word for another set of attributes. For example, as mentioned in [6], in the sentence, 'A rare black squirrel has become a regular visitor to a suburban garden', the triplet extracted out of this sentence is squirrel-become-visitor where squirrel has as attributes the adjectives rare, black and the article a; the word become has as attributes its auxiliary has and the object visitor has as attributes the adjective regular, the article a and the noun garden with its attributes: preposition to, article a and adjective suburban. Thus, we shall construct the RDF graph by extracting subject-object-predicate triples from sentences in the text, wherein there shall be nodes that correspond to a subject phrase connected to a node corresponding to the object phrase with a directed edge labeled with the relation between these 2 nodes (the predicate value). Additionally, in order to enable us to effectively perform CEPS on the RDF graph, we also weight each labeled edge with the frequency of occurrence of the corresponding predicate connecting the 2 subject and object head nodes. Now, we shall describe the algorithm used for extracting subject-object-predicate triples from the text in radiology reports and thereby construct an RDF graph from this text.

2.1 Algorithm to Extract S-O-P Triples from the text:

For this purpose, we closely follow the method for triplet extraction from biomedical literature proposed in [4]. However, we observed this method to have its own shortcomings on our text, and we chose to suitably modify it to get the best performance. For extracting subject-object-predicate triples, we used the dependency parse tree in the Stanford Parser to get relations between entities of the sentence. The Stanford dependency scheme contains 50 grammatical relations organized in a hierarchy. In our algorithm, similar to the approach followed in [4], we focus our attention mainly on the argument, conjunct, auxiliary and modifier dependency types. The algorithm for triplet extraction

was encoded in the form of rules. To minimize the number of rules, we used the hierarchy of the dependency types provided by the Stanford Parser. We consider a dependency d to belong to a dependency type C, if d is located under C in the dependency hierarchy. Based on this, we divide the dependencies (or relations) into 4 classes as follows: (1) Class Subject contains the relations {nsubj, csubj, nsubjpass}, (2) Class Complement contains the relations {mark, dobj, iobj, pobj, acomp, attr, ccomp, xcomp, compl, rel} (3) Class Preposition contains the relations {prep, prepc} and (4) Class Subject 2 contains the relation {infmod}. We iterate over all the edges in the dependency parse tree and use the following rules to extract triples:

(1) If a dependency d(w1,w2) is in the class SUBJECT, we mark w1 as the head of a predicate and w2 as the head of a subject.
(2) If a dependency d(w1,w2) is in the class COMPLEMENT and w2 is not a verb, then mark w1 as the head of a predicate and w2 as the head of an object.
(3) If a dependency d(w1,w2) is in the class PREPOSITION and w1 is a verb, then mark w1 as the head of a predicate and w2 as the head of an object.
(4) If a dependency d(w1,w2) is in the class SUBJECT 2, then mark w1 as the head of a subject and w2 as the head of a predicate.

Also, unlike [4], we realize the fact that a sentence may have more than one predicate (for example sentence with conjunctions), and thus more than one triple. So, we constructed our algorithm to extract more than one triple from the sentence (if such a case applies), where a subject-object-predicate triple is a unique pair of subject head and object head connected by the same predicate.

Next, we use the head words in the subject and object to extract the full subject and full object phrase in the subject and object of each triple, by looking at the dependency parse tree and combining siblings of this head word in the dependency parse tree that are also locally close to the head word in the original sentence, to form a complete phrase. Thus, at each iteration over sentences in the text, we extract triples, where for each triple, we extract a subject phrase, an object phrase and a predicate and then add these triples to the RDF graph as follows – one node represents a subject phrase, another an object phrase and there is a directed edge formed from the subject node to the object node that is labeled with the predicate term. While extracting triples from a sentence and dynamically adding them to the RDF graph, we compute a similarity score between the phrase of the current node with the phrase that every other node in the RDF graph represents, and merge it with a node with which it has the highest similarity score and the score is above a certain threshold. Thus, one node may be a node representing more than one phrase, but since all these phrases are similar to each other semantically, in our resulting RDF graph, it is constructed to represent only the first such phrase. The formula that we use to calculate the similarity score between 2 phrases is shown below:

$$similarity(s_i, s_j) = \frac{\sum_{w \in s_i, s_j} log(1 + freq(w))}{\sum_{w_i \in s_i} log(1 + freq(w_i)) + \sum_{w_j \in s_j} log(1 + freq(w_j))}$$

---- (1)

At subsequent iterations, if we encounter the same set of subject-object nodes (where the nodes are labeled by the head words) connected by the same predicate, we increment the weight of this edge.

Figure 1 shown below, gives a graphical view of a small part of the RDF graph (with edge weight information and predicates that are the edge labels) extracted from the radiology reports (150 mammogram reports returned up by the search query "cancer" given to our datastore).

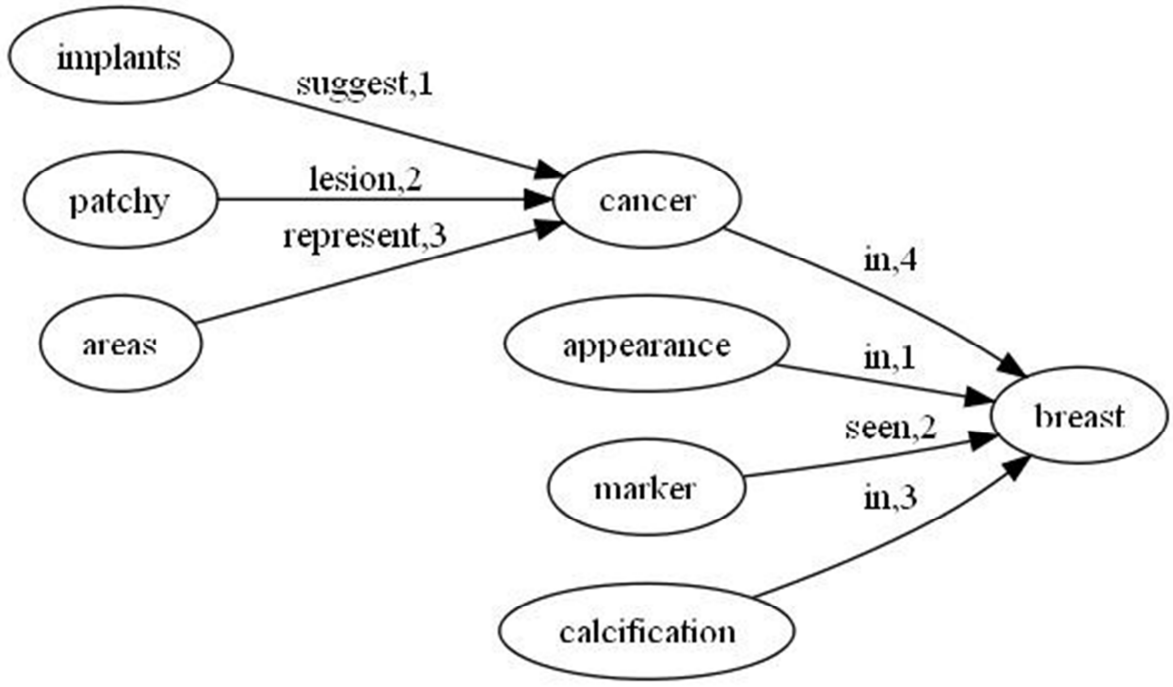

Fig1. Snapshot of a small part (subgraph) of the RDF graph (with edge weight information and predicates that are the edge labels) generated from the text in 25 mammogram reports that were returned as an output to the search query "cancer" to our datastore.

3. CENTER-PIECE SUBGRAPH ON OUR RDF GRAPH:

As mentioned earlier, the problem of extracting a connection subgraph (i.e. CEPS AND query for k=2), has been discussed and solved in [3]. It would seem logical to begin by extending this algorithm for any k. However, we would like to draw attention to the fact that none of the heuristics used for extracting the *Display ρ-graph* (the connection subgraph that is outputted) from the *Candidate ρ-graph* (the graph obtained by pruning the subspace in very large graphs, and from which the *Display ρ-graph* is extracted) are applicable to our case and domain. Specifically, since we do not have knowledge of class hierarchy of words and entities in the text of the radiology reports (as there is no current knowledge bank having information of class hierarchy of biomedical terms, and this is very difficult to construct too), we cannot use the *Class and Property Specificity* heuristic. Also, since we do not have information about what biomedical facts are rarer than others, it is not possible for us to use the *Instance Participation Selectivity* heuristic described in this paper. Also, we do not have any pre-disposed information of what node hops in the graph will give us maximum information gain. Lastly, we cannot even use the SPAN heuristic as we do not have multiple RDFS schema in our case. Our domain is only radiology report text, and biomedical terms and so, we have just one schema (a collection of Class and Property types). Thus, in our specific

case, we can't extend the algorithm described in [3] to perform CEPS for any k. However, our implementation of CEPS on the RDF graph draws inspiration from the CEPS algorithm described in [1] for the case of an undirected graph. In fact, we observed the CEPS algorithm described in [1], with minor modifications, to work well on our graph if we treated each edge as directed. To better appreciate this, notice that the RDF graph in our case consists of subject nodes connected to object nodes with directed edges (weighted and labeled) from the subject node to the object node. Thus, all outgoing edges in the graph are from the subject nodes and all the incoming edges are into the object nodes. With such a structure of the graph, it doesn't seem wise to compute the best paths between two nodes, and to compute the stationary probabilities of arriving at a particular node starting from a given set of query nodes (that are steps towards the computation of the CEPS), by taking into account the directed nature of the edges. Also, since we are only interested in the best subgraph that connects a set of subject/object nodes, it suits for our purpose to consider the edges as undirected for extracting CEPS. The algorithm for CEPS that we ran on our RDF graph is described briefly below (this algorithm is almost completely influenced by the algorithm described in [1], but has a few minor modifications to suit our purpose):

The outline for the algorithm (similar to the algorithm in [1]) is described below:

Algo-box 1: *CEPS*
Input: the weighted graph **W**, the query set *Q*,
K softAND coefficient *k* and the budget *b*
Output: the resulting subgraph *H*
Step 1: Individual Score Calculation. Calculate the goodness score *r*(*i, j*) for a single node *j* wrt a single query node *qi*
Step 2: Combining Individual Scores. Combine the individual score *r*(*i, j*) to get the goodness score *r*(*Q, j*) for a single node *j* wrt the query set *Q*
Step 3: "*EXTRACT*". Extract quickly a connection subgraph *H* with budget *b* maximizing the goodness criteria *g*(*H*)

Here, the k_SoftAND parameter k denotes that we want to find a CEPS such that each node in the resulting subgraph is well connected to at least k of the query nodes. The budget b is a parameter that decides the maximum number of nodes apart from the query set we want the CEPS to have. We shall now briefly describe Steps 1,2 and 3 mentioned in this outline.

Step 1: (Individual score calculation)
In this step, we shall calculate r(i,j), the goodness score of node j, for a single query node q_i. This is the same as $r_{i,j}$, the *steady state* probability that a particle on a Random-Walk with Restarts (RWR) will finally state at node j, starting from query node q_i, where the probability for the particle on RWR hopping from node a to node b is proportional to the edge-weight of the edge connecting these 2 nodes. If we put all the probabilities $r_{i,j}$ into a matrix R, then in matrix-form this can be written as: $\mathbf{R^T} = c\mathbf{R^T x \tilde{W}} + (1-c)\mathbf{E}$, where $\tilde{W}$ is the adjacency matrix appropriately normalized with the degree matrix D, c is the fly-out probability in the RWR and E = [e_i](the axis vectors in a vector space). The weight matrix, here reflects our case of parallel edges between nodes (as is the case in an RDF graph – a multigraph), and we initialize the unnormalized adjacency matrix, W such that W[i,j] is the sum of the weights of all the edges between the

2 nodes i and j. We also realize that while ultimately in Step 3, when we shall compute the *Key Paths* from the query nodes, the edge with the highest weight will be chosen in computing the path.

Step 2: (Combining individual scores).
In this step, we compute r(Q,j,k), the goodness score of node j, with respect to the query set Q and k_softAND parameter k, and can be computed as shown in [1] by the recursive equation:
$r(Q, j, k) = r(Q', j, k-1).\ r(Q, j) + r(Q', j, k)$
where $r(\emptyset, j, 0) = 1 (j = 1, ..., Q)$. and Q' is the set of the first Q-1 query nodes in the query set Q.

Step 3: (*Extract* step)
In this step, given a weight matrix W, k_softAND parameter k and budget b (budget is the max number of nodes in the Center-Piece Subgraph that is not in the query set Q), we extract the Center-Piece Subgraph, H. Before we jump to the algorithm, let us first define a few terms as explained in [1]:

SPECIFIED DOWNHILL NODE - Node u is downhill from node v wrt source q_i ($v \rightarrow d_i, u$) if $r(i, v) > r(i, u)$;
SPECIFIED PREFIX PATH - A specified prefix path $P(i, u)$ is any downhill path that starts from source q_i and ends at node u; that is, $P(i, u) = (u_0, u_1, ..., u_n)$ where $u_0 = q_i$, $u_n = u$, and $u_j \rightarrow d_i, u_{j+1}$;
EXTRACTED GOODNESS - The extracted goodness is the total goodness score of the nodes within the subgraph H: $CF(H) = P_{j \in H}\ r(Q, j)$.
EXTRACTED MATRIX. Cs(i, u) - is the extracted goodness score from source node q_i to node u along the prefix path $P(i, u)$ so that:

1. $P(i, u)$ has exactly s nodes not in the present output graph H
2. $P(i, u)$ extracts the highest goodness score among all such paths that start from qi and end at u.

ACTIVE SOURCE. For *K softAND*, the source node q_i is active w.r.t. destination node pd if $r(i, pd) \geq r^{(k)}(i, pd)$, where $r^{(k)}(i, pd)$ is the *kth* largest value among $r(i, pd)$, $(i = 1, ..., Q)$.
Also, let us define pd, the promising destination node selected by the algorithm at each step, and for this node, the algorithm tries to find the best source-destination path for the query nodes (in case of parallel edges between nodes, the edge with the highest weight is chosen in the path) .

$$pd = argmax_{j \notin H}\ r(Q, j))$$

Thus, our *Extract* algorithm is as shown below:

Algo-box 3: Our *EXTRACT* Algorithm
1. Initialize output graph *H* null
2. Let *len* be the maximum allowable path length
3. While *H* is not big enough
    3.1. Pick up destination node *pd* by the Eq. given above.
    3.2. For each active source node q_i wrt node *pd*
        3.2.1. use Algo-box 4 to discover a key path $P(q_i, pd)$
        3.2.2. add $P(q_i, pd)$ to *H*
4. Output the final *H*

Algo-box 4: Single Key Path Discovery
1. Let *len* be the maximum allowable path length
2. For $j \leftarrow [1, ..., n]$
    2.1. Let $v = u_j$

2.2. For $s \leftarrow [2, ..., len]$

If v is already in the output subgraph

$s' = s$

Else

$s' = s - 1$

Let $C_s(i, v) = max_{u|u->d\_i,v} (C_{s'}(i, u) + r(Q, v))$

3. Output the path maximizing $C_s(i, pd)/s$, where $s \neq 0$. In case of parallel edges, choose the edge with the highest weight.

Thus, we have described the algorithm used for performing CEPS on our RDF graph. In Section 4 below, we shall describe in detail the idea of *Context* as applicable to our case and explain how we included this concept in our analysis

4. INCLUDING CONTEXT IN OUR ANALYSIS

In the creation of our RDF graph, we connect various objects with certain predicates. However, there is always a context to each of these objects. Moreover, after creating an RDF graph from a large corpora of radiology reports, one might want to analyze the graph with a specific context in mind. E.g. for an RDF graph constructed from a large database of mammograms, a person wanting to know how certain query nodes are connected with respect to the context of "breast cancer" wouldn't want to know how these nodes are connected with nodes that represent other deformities or ailments of the breast, or those that talk about fractures of the ribs, etc., and would only like to obtain a Center-Piece Subgraph that gives him/her information about the subgraph that is a best connection to the query nodes, and the nodes of which represent phrases that are semantically related to the context of "breast cancer". This is the same as saying that a Random Walker in Step 1 of our CEPS algorithm, making Random Walks with Restarts (RWR) on our RDF graph starting from each of the query nodes, has a higher probability of hopping to a node that represents a phrase which is semantically related to the context phrase under consideration, and this is exactly what we aim to achieve by modifying our algorithm to incorporate *Context*.

Towards this end, we make a few modifications: Since we are interested in nodes that represent phrases semantically closer in meaning to the context phrase, it is a better idea to compute the similarity score of each node's phrase with respect to phrases that are semantically related to the context phrase rather than computing the similarity score with respect to only the context phrase. Thus, we query 2 popular ontologies of biomedical terms, RadLex and SNOMED with the context phrase and dump the top 10-20 terms that they return (we shall call this the Context Phrase Set). Next, after we have constructed an RDF graph as explained in Section 2, we bias the weights in the adjacency matrix so that nodes that are semantically closer in meaning to the context phrase have higher probability of being visited by a Random Walker on RWR. This is done by carrying out the following computation on the adjacency matrix (or the weight matrix)

$$W[:, j] = W[:, j] + Similarity(n_j) \quad \text{------ (2)}$$

where Similarity(n_j) is the average of the similarity of the phrase represented by node j with respect to the phrases in the Context Phrase Set and is computed as:

$$Similarity(n_j) = \sum_i similarity(p_i . n_j)$$ ------ (3)

where p_i is the i^{th} phrase in the Context Phrase Set and similarity(p_i,n_j) is the similarity score between p_i and n_j, calculated as in Equation (1) in Section 2. Also, W[:,j] in Equation (2) above represents the j^{th} column of the weight matrix, W.

Observe that now, nodes with a higher similarity score with the Context Phrase Set have a higher probability of being visited by the Random Walker. We then normalize this weight matrix so that the sum of all rows equals one before using it to compute the R matrix in the CEPS algorithm as explained in Section 3.

As one might guess, having a certain Context phrase significantly affects the results obtained after performing CEPS. Moreover, given that we construct an RDF graph from a certain class of text material, one may obtain very good results with a certain context phrase, and extremely poor or meaningless ones with a context phrase that has nothing to do with the subject of the text material from which the RDF graph was constructed.

We now present results of our analysis that we obtained, with and without setting a Context phrase. We also compare the 2 scenarios and discuss how having a certain context affects the CEPS extracted. Finally, we also present a case in which we have a Context phrase because of which we are unable to obtain a meaningful CEPS.

5. RESULTS:

We first generated an RDF graph from the text in 100 radiology reports (mammograms). This RDF graph generated from the text in 100 radiology reports consisted of 175 nodes and 472 edges. For the CEPS analysis, the edge weight between two nodes was taken to be the sum of the weights of all the edges between these two nodes.

We ran our algorithm for a number of queries, and we obtained fairly good results. We present below in Fig. 2 and Fig. 3 the result we obtained after running our CEPS analysis on this dataset for query nodes, Q = {“muscle”,”nipple” and “malignancy”}, first without having any context phrase and then with the context phrase = “breast cancer”, both with k=2 and b=7 (here, k and b are the k_SoftAND parameter and budget as mentioned in Section 3 earlier – k denotes that we want our CEPS to have nodes that are well connected to at least k of the query nodes and b is the max. number of the nodes apart from the nodes in the query set we want to have in our extracted CEPS). In the figures below illustrating the CEPS extracted, the nodes represented by squares are the ones belonging to the query set, and the ones represented by an ellipse do not belong to the query set.

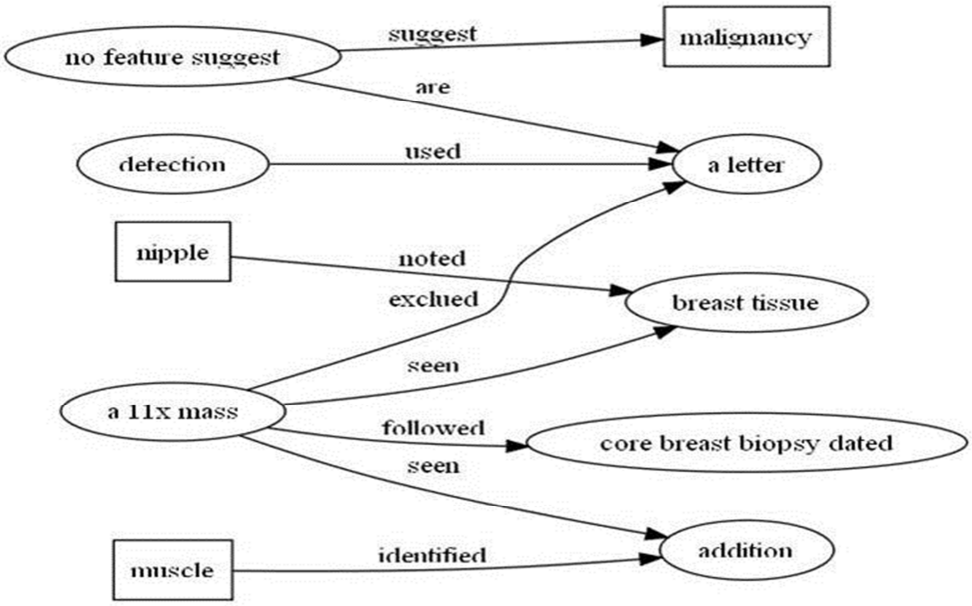

Fig. 2: CEPS extracted from RDF for Q = {"muscle","nipple" and "malignancy"}, k=2 and b=7 and without any context phrase.

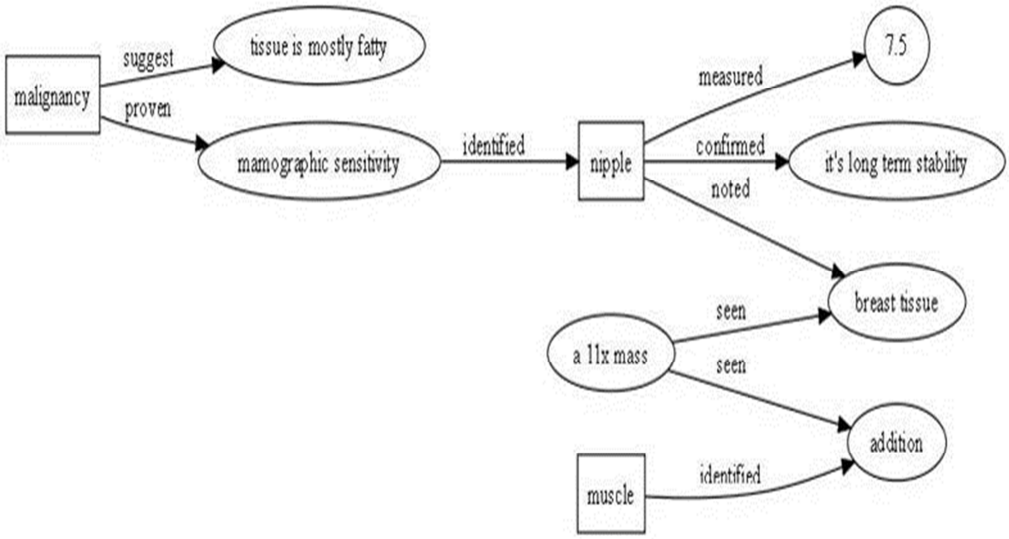

Fig 3. CEPS extracted from RDF for Q = {"muscle","nipple" and "malignancy"}, k=2 and b=7 and with context phrase = "breast cancer"

For this dataset, we tested our algorithm for small values of k and b, and on small query sets since since this RDF graph was extracted from 100 radiology reports, and hence is sparse and has fewer number of nodes and edges.

Observe that introducing “breast cancer” as the context phrase significantly affects the CEPS extracted from the RDF graph. The CEPS extracted with the context phrase “breast cancer” has nodes such as “tissue is mostly fatty”, “mammographic sensitivity” and “its long-term stability” which are semantically more related to breast cancer, and which were not present in the CEPS that was generated without any context. Observe that the predicate links, the connection and paths between nodes, and the subgraph as a whole gives much more information to a radiologist about the connection between the terms “nipple”, “malignancy” and “muscle” from the perspective of breast cancer.

We next also varied our parameter b, where b is the budget that denotes the max. number of nodes we apart from the query set we want our CEPS to have. As expected, when we restrict our budget b to a very small number, we do not get a very meaningful result as restricting the value of b will not allow the Key Path Discovery step of the CEPS algorithm to explore nodes more than a few hops away, and the nodes that are returned as a part of CEPS with such a small b may not be semantically as meaningful if the value of b was larger. We give one such example below in Fig. 4 where we show the CEPS extracted for the query set Q = {“similar study”, “stable appearance lesions quadrant”, “both breasts”}, b=2 and k=1 with the context phrase as “breast cancer” (because of restricting b to 2, for values of k>1, we didn’t get any meaningful result) .

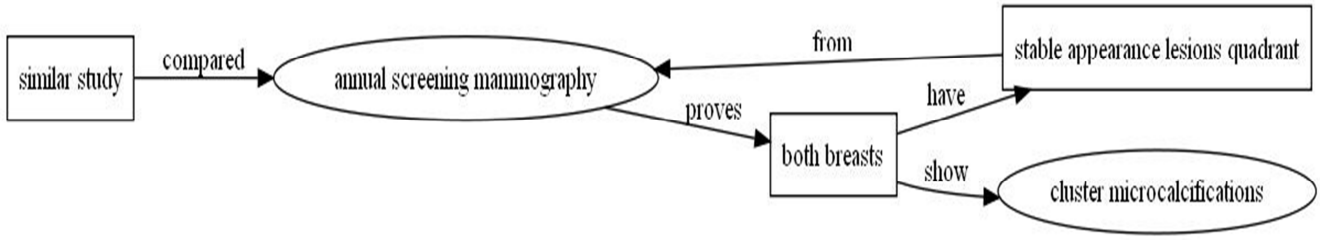

Fig 4. CEPS extracted from RDF for Q = {”similar study”, ”stable appearance lesions quadrant” and “both breasts”}, k=1 and b=2 and with context phrase = “breast cancer”.

We also tried running CEPS with other context phrases. Specifically, we were interested in the results we got when the context phrase was not at all related to the subject matter of the text analyzed. Our dataset consisted of mammograms, so we also analyzed the CEPS generated when the context phrase was “brain tumor”, which is semantically unrelated to the information presented in mammography reports. We show below in Fig. 5, the CEPS extracted for the query set, Q={“core breast biopsy dated” and “craniocaudal”} with k=2 and b=4. Observe that the CEPS extracted is not semantically relevant compared to the CEPS extracted when the context phrase was “breast cancer” as shown in Fig. 3 above. In the limiting case when the context phrase is tangential to the subject matter of the text, we expect CEPS to behave as well as (or sometimes a little worse) compared to the case of no context phrase at all. This is because, most node phrases in the RDF graph have very little similarity with the Context Phrase Set, and there may be some node phrases that have a higher similarity with the Context Phrase Set, but do not want to highlight them in the context of the subject matter of the text used for extracting the RDF graph.

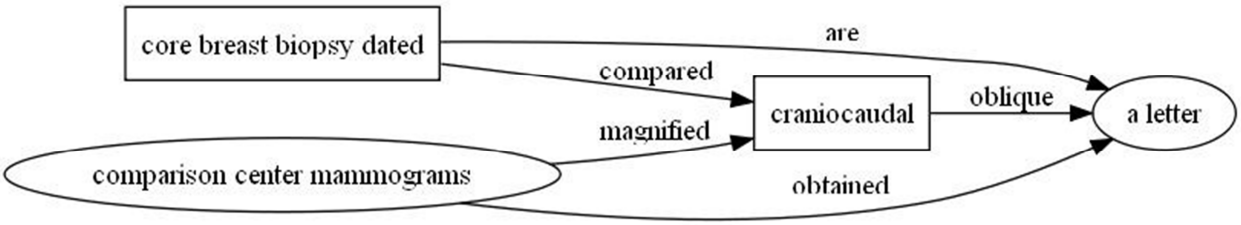

Fig 4. CEPS extracted from RDF for Q = {"core breast biopsy dated" and "craniocaudal" }, k=1 and b=2 and with context phrase = "brain tumor".

6. Conclusion and Future Work:

From the results that we obtained, we conclude that our proposed method does indeed succeed in extracting a meaningful Center-Piece Subgraph that well represents the semantic relation between the query phrases in the text (radiology reports in our case). The results obtained are even better if we have a context phrase that is semantically well aligned with the subject matter of the text from which the RDF graph is generated ("breast cancer" in our case as we extracted text from mammography reports). However, as one can easily observe, the success of this analysis largely depends on the efficacy of the triplet extraction algorithm, and the quality of triples extracted. After examining the RDF graph extracted from the text in the radiology reports and the CEPS computed for various queries, we observed that the rules coded for extracting triples are indeed able to extract correct triples from sentences and produce an RDF graph on which subsequent analysis can be easily performed.

However, there were a number of shortcomings that we faced in our analysis. Firstly, in the text of the radiology reports, there were a number of sentences that were not either grammatically complete, or had special characters, abbreviations, etc. that caused the Stanford Parser to generate a dependency parse tree from which good triples could not be extracted. Specifically, there were a lot of sentences such as "slight nipple retraction is present" that do not have any object. In such cases, one or more null nodes were generated in the triple, and had to be pruned away before the CEPS analysis. Also, there were a number of sentences of the form, "impression: 1. right breast: bi-rads 2, benign. left breast: bi-rads 1, negative. recommend follow up screening mammogram in 12 months.", that are grammatically incorrect but are used immensely in biomedical literature to describe observations. In these cases also there were a number of null nodes thrown up as the dependency parse generated by the Stanford Parser could not produce results that could be used to extract meaningful triples, and thus in a number of cases, an object or a subject would not be detected. Also, many special characters, in words such as 'findings:' - which when stripped in text preprocessing would change the meaning of the sentence - were a roadblock in effective triple extraction. We did a bit of research on this issue and found to our dismay that not much work has been done to handle triplet extraction in these cases. Most works assume a grammatically complete sentence and which do not have many abbreviations, special characters, etc. Thus, in the future we would like to explore this issue more so that we are able to extract better triples from this kind of text and so that our subsequent analysis isn't handicapped by it.

We have a dataset of 41,142 mammogram reports, but when ran our algorithm to construct an RDF graph on this dataset, or even a small fraction of it, it took a very long time to run even on a modern computer, and so for the purpose of our experiments, we had to consent with analyzing a smaller number of reports(50, 100, 150, etc). Thus, in the future, we also plan to work on scaling this algorithm up for a very

large dataset of reports (like the dataset of 41,142 reports that we have) by trying to construct a parallel version of our algorithm and running it on a high performance cluster.

We also plan to evaluate our method in a more concrete manner. Since the output of our analysis is qualitative in nature, we have decided to measure success by showing a number of results from different queries and context phrases to Dr. Daniel Rubin from the Department of Radiology at Stanford and other radiology experts, and obtain a qualitative rating on a scale of 1 to 10 of the results generated for different context phrase, so that we are able to analyze our method better. However, due to a packed schedule, Dr. Daniel Rubin was unable to schedule an appointment with us as yet, and we plan to do this very soon.

7. REFERENCES:

1. Tong, H., Faloutsos, C. Center-Piece Subgraphs: Problem Definition and Fast Solutions. Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, pages 404-413, 2006.

2. Faloustos, C., McCurley, K.S., Tomkins, A. Fast Discovery of Connection Subgraphs. . Proceedings of the 10th ACM SIGKDD international conference on Knowledge discovery and data mining, pages 118-127, 2004.

3. Ramakrishnan, C., Milnor, W.S., Perry, M., Sheth, A.P. Discovering Informative Connection Subgraphs in Multi-relational Graphs. ACM SIGKDD Explorations Newsletter, Volume 7, Issue 2, pages 56-63, December 2005.

4. Ramakrishnan, C.; Mendes, P.N.; da Gama, R.A.T.; Ferreira, G.C.N.; Sheth, A.P. Joint Extraction of Compound Entities and Relationships from Biomedical Literature. International Conference on Web Intelligence and Intelligent Agent Technology, 2008, pages 398-401, Dec. 2008.

5. Cheng, J.; Yiping Ke; Ng, W.; Yu, J.X. Context-Aware Object Connection Discovery in Large Graphs. IEEE 25th International Conference on Data Engineering, 2009, pages 856-867, April 2009.

6. D. Rusu, L. Dali, B. Fortuna, M. Grobelnik, D. Mladenic, Triplet extraction from sentences, SiKDD 2007, October, 2009, Ljubljana, Slovenia.