Single Document Keyphrase Extraction Using Label Information

Sumit Negi
IBM Research
Delhi, India
sumitneg@in.ibm.com

Abstract

Keyphrases have found wide ranging application in NLP and IR tasks such as document summarization, indexing, labeling, clustering and classification. In this paper we pose the problem of extracting *label specific* keyphrases from a document which has document level metadata associated with it namely *labels* or *tags* (i.e. multi-labeled document). Unlike other, supervised or unsupervised, methods for keyphrase extraction our proposed methods utilizes both the document's text and label information for the task of extracting *label specific* keyphrases. We propose two models for this purpose both of which model the problem of extracting label specific keyphrases as a random walk on the document's text graph. We evaluate and report the quality of the extracted keyphrases on a popular multi-label text corpus.

1 Introduction

The use of graphs to model and solve various problems arising in Natural Language Processing have lately become very popular. Graph theoretical methods or graph based approaches have been successfully applied for a varied set of NLP tasks such as Word Sense Disambiguation, Text Summarization, Topic detection etc. One of the earliest and most prominent work in this area has been the TextRank (Mihalcea and Tarau, 2004) method - an unsupervised graph-based ranking model for extracting keyphrases and "*key*" sentences from natural language text. This unsupervised method extracts prominent terms, phrases and sentences from text. The TextRank models the text as a graph where, depending on the end application, text units of various sizes and characteristics can be added as vertices e.g. open class words, collocations, sentences etc. Similarly, based on the application, connections can be drawn between these vertices e.g. lexical or semantic relation, contextual overlap etc. To identify "central" or "key" text units in this text graph, TextRank runs the *PageRank* algorithm on this constructed graph. The ranking over vertices (text units), which indicates their centrality and importance, is obtained by finding the stationary distribution of the random walk on the text graph.

In this paper, we consider the problem of extracting *label specific* keyphrases from a document which has document level metadata associated with it namely *labels* (i.e. multi-labeled document). To elaborate, consider a document as shown in Figure 1. This document has been assigned to two categories as indicated by the labels "*Air Pollution*" and "*Plant Physiology*". Running TextRank on this article yields top ranked key-phrases such as "*calibrated instrument*", "*polluting gases*", "*industrial development*" etc. These keyphrases, though central to the article, are not specific to any of the *labels* that have been assigned to the article. For instance, one would associate keyphrases such as "*carbon monoxide* ", "*air pollutants*" to be more relevant to the "*Air Pollution*" label and keyphrases such as "*stomatal movement*", "*cell defense*" to be more closely associated with the "*Plant Physiology*" label. The objective of this paper is to explore extensions to TextRank for extracting label-specific keyphrases from a multi-labeled document. Such label-specific keyphrases can be useful for a number of practical applications namely: highlighting such terms within the body of a document could provide a label-specific (topic-focussed) view of the document thus facilitating fast browsing and reading of the document, such key terms could also be useful for generating topic-driven or label-specific summaries and in multifaceted search.

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1468–1476, Dublin, Ireland, August 23-29 2014.

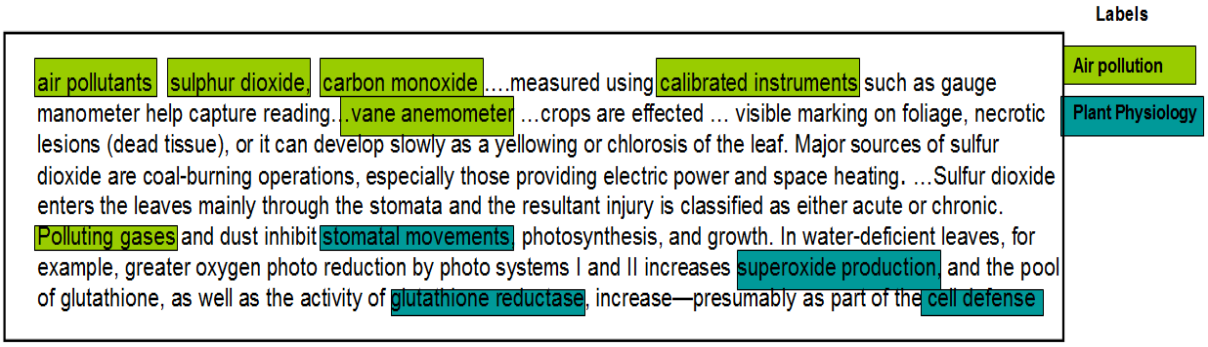

Figure 1: Label specific keyphrases (best viewed in color). Note that there could be keyphrases that are common to both labels. Due to space restrictions only a snippet of the document is shown.

The rest of the paper is organized as following. We discuss related work and provide an overview of our approach in Section 2. Details of the proposed method is discussed in Section 3 followed by evaluation in Section 4. Future work and conclusion is presented in Section 5.

2 Related Work

The methods for keyphrase (or keyword) extraction can be roughly categorized into either *unsupervised* or *supervised*. *Unsupervised methods* usually involve assigning a saliency score to each candidate phrase by considering various features. Popular work in this area include the use of point-wise KL-divergence between multiple language models for scoring both *phrase-ness* and *informativeness* of candidate phrases (Tomokiyo and Hurst, 2003), use of TF-IDF weighting (A. Hulth, 2003) etc. *Supervised machine learning algorithms* have been proposed to classify a candidate phrase into either keyphrase or not using features such as the frequency of occurrence, POS information, and location of the phrase in the document. All the above methods only make use the document text for generating keyphrases and cannot be used (as-is) for generating label-specific keyphrases.

One possible method for extracting label-specific keyphrases from a document could be based on post-processing the output of the TextRank algorithm in the following way (1) Identify a set of *label specific features* f_l^{cand} (unigram terms) that are strongly correlated with the *label*. This could be done by applying feature selection methods (Forman, 2003), (Forman, 2003) on a multi-label text corpus (we discuss this step in more detail in a later section). For instance, $f_{air\_pollution}^{cand}$={"*pollutant*","*gases*",...} (2) Run the TextRank algorithm on the document d to generate a list of keyphrases $keyphrase_d$ (3) *Filter* the resultant list $keyphrase_d$ based on lexical or semantic match with the label specific features f_l^{cand} to generate $keyphrase_d^l$ or label-l specific keyphrase for document d.

This approach suffers from the following limitations (a) The keyphrase list generated in Step (2) i.e. $keyphrase_d$ might be dominated by keyphrases which have little to do with label l. Post processing this list (Step 3) using f_l^{cand} might result in only very few keyphrases in $keyphrase_d^l$. (b) The label specific features f_l^{cand}, which are derived from corpus level statistics[1], might not be the best indicator of the *keyphrase-ness* of a term in the document. (c) Moreover, consider a scenario where a document is associated with more than one label. Consider the previous example where the document is associated with two labels "*Air Pollution*" and "*Plant Physiology*". When extracting keyphrases specific to the label/category "*Air Pollution*" from document d one would expect that the extracted keyphrases are *closer* to the *Air Pollution* label/category and *distant* from other labels associated with document d i.e. "*Plant Physiology*". It is not evident how this can be modeled in this approach. In this paper we propose an approach that models the problem of finding label-specific keyphrases in a document as a random walk on the document's text-graph. Two approaches are proposed namely *PTR: Personalized TextRank* and *TRDMS: TextRank using Ranking on Data Manifolds with Sinks*.

[1]Using feature selection methods

PTR: Personalized TextRank : In this setting the PageRank algorithm, which is the underpinning of the *TextRank* keyphrase extraction algorithm, is replaced with the personalized page rank (Haveliwala, 2002) algorithm. By using the label specific features f_l^{cand} as the *personalization vector* we are able to bias the walk on the underlying text graph towards terms relevant to the label. We discuss this approach in more detail in Section 3.3. Even though using a label specific *transport* or *personalization vector* helps bias the walk towards terms specific to that label, terms relevant to labels other than l continue to influence the walk. The *Personalized TextRank* method offers no elegant solution which would penalize terms unrelated to l while simultaneously preferring terms relevant to label l.

To achieve both these goals in one model we propose the *TRDMS: TextRank using Ranking on Data Manifolds with Sinks* approach. We model the problem of identifying label specific keyphrases in a given document as a random walk over the document's *weighted* text graph with *sink* and *query* nodes[2]. Ranking on data manifolds was first proposed by (Zhou et al., 2004) and has been used for multi-document summarization (Wan et al., 2007), image retrieval (He et al., 2004) etc. An intuitive description of the ranking algorithm is described as follows. A weighted network is constructed first, where nodes represent all the data and query points, and an edge is put between two nodes if they are "*close*". *Query* nodes are then initialized with a positive ranking score, while the nodes to be ranked are assigned a zero initial score. All the nodes, except the *sink* nodes, then propagate their ranking scores to their neighbor via the weighted network. The propagation process is repeated until a global state is achieved, and all the nodes except the query nodes are ranked according to their final scores. Manifold ranking gives high rank to nodes that are close to the query nodes on the manifold (*relevance*) and that have strong centrality (*importance*). Sink nodes, whose ranking is fixed to the minimum (zero) during the ranking process, do not spread any ranking score to their neighbors thus penalizing the nodes that are connected to them. To use this method for extracting label-(l) specific keyphrases , f_l^{cand} are modeled as *query* nodes while features associated with labels other than l are modeled as *sink* nodes. This approach is inspired by the work done by (Cheng et al., 2011) for query recommendation and update summarization. Section 3.4 discusses this method in more detail. To summarize, to the best of our knowledge we are the first to propose the problem of extracting *label* specific keyphrases from a multi-labeled document. Our modifications to TextRank for achieving this task are novel. Moreover, our idea of of using *Ranking on Data Manifolds* on the document-level text graphs for extracting label specific keyphrases is a new contribution.

3 Generating Label Specific Keyphrases

3.1 Notation

In this section we introduce notations which we use throughout the paper. Let D represent a multi-label document corpus and $\Im$ be the set of all possible labels which could be associated with documents in D. A document from this corpus is denoted by d and the set of labels associated with document d is denoted by ℓ, where $d \in D$ and $\ell \subseteq \Im$. The text graph for document d is denoted by G_d and M denotes the number of vertices in G_d . We describe how this text graph is constructed in Section 3.2. Features specific to label l, which are extracted from the corpus D, are represented as f_l^{cand}, where $l \in \Im$. Section 3.5 describes how these *label specific features* are extracted from a multi-label document corpus.

3.2 Building the Text Graph

For a given document d the text graph G_d is built in the following way. All open-class, unigram tokens occurring in d are treated as vertices. Two vertices are connected if their corresponding lexical units co-occur within a window of maximum N words, where N is set to 10 for all our experiments. As indicated by (Mihalcea and Tarau, 2004) co-occurrence links express relations between syntactic elements and represent cohesion indicators for a given text. Note that the methods described in Section 3.3 and Section 3.4 provide a score/rank for each vertex (unigram term) in the graph. To generate keyphrases (n-grams) from these candidate terms the following post-processing is performed on the top ranked terms. Vertices are sorted in reverse order of their score and the top K vertices in the ranking are retained

[2]Nodes correspond to terms in a text graph

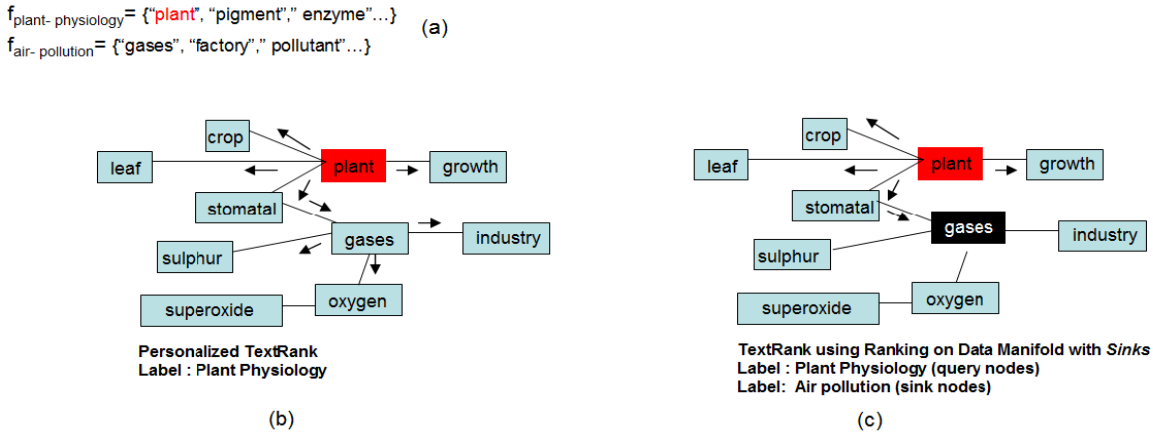

Figure 2: (a) Label specific features f_l^{cand} (b) *Personalized TextRank* - walk biased towards terms related to $f_{plant\_physiology}^{cand}$ (shown in red color). (c) *TextRank using Ranking on Data Manifold with Sinks*: walk biased towards terms related to $f_{plant\_physiology}^{cand}$, while simultaneously penalizing terms that are related to $f_{air\_pollution}^{cand}$. The sink points, which are shown in black color, are vertices whose ranking scores are fixed at the minimum score (zero in our case) during the ranking process. Hence, the sink points will never spread any ranking score to their neighbors. Arrows indicate diffusion of ranking scores (Figure best viewed in color)

for post-processing. Let this ranked list be represented as <T_K>. During post-processing, all terms selected as potential keywords are marked in the text, and sequence of adjacent keywords are collapsed into a multi-word keyphrase. For example, in the text *calibrated instruments are used to measure*, if the unigram terms *calibrated* and *instruments* are selected as potential/candidate terms by the PTR or TRDMS method, since they are adjacent they are collapsed into one single keyphrase "*calibrated instruments*". This heuristic is implemented as a function which is referred as $kphrase_{gen}$(<T_K>,d). This function takes as input the ranked term list <T_K> and the document text d and returns the collapsed set of keyphrases. A similar approach was adopted in the *TextRank* (Mihalcea and Tarau, 2004) work.

3.3 *PTR*: Personalized TextRank

For extracting label-l specific keyphrases from document d we modify the TextRank (Mihalcea and Tarau, 2004) algorithm. We replace the *PageRank* algorithm used in the *TextRank* method with the *Personalized Page Rank* (Haveliwala, 2002) algorithm. PageRank gives a stationary distribution of a random walk which, at each step, with a certain probability ϵ jumps to a random node, and with probability 1-ϵ follows a randomly chosen outgoing edge from the current node. More formally, let G_d denotes the text graph of document d with M vertices where d_i denotes the out degree of node w_i, then $p = \epsilon Lp + (1\text{-}\epsilon)v$. Where p is the page rank vector, L is a $M \times M$ transition probability matrix with L_{ji}=$\frac{1}{d_i}$. In the page rank equation v is a stochastic normalized vector whose element values are all $\frac{1}{M}$. This assigns equal probabilities to all nodes in the graph in case of random jumps. In the *personalized* page rank formulation the vector v can be non-uniform and can assign stronger probabilities to certain kind of nodes effectively biasing the PageRank vector. In the *PTR* approach v is modeled to capture the evidence that is available for label l in document d. Doing so biases the walk towards terms that are more specific to label l in the document. This is achieved by considering vertices (terms) that are common between the label l feature vector i.e. f_l^{cand} and the text graph for document d i.e. G_d. More precisely, for a label l associated with a document d, let V_d^l denote the intersection of the set V_d with f_l^{cand}, i.e. $V_d^l = V_d \cap f_l^{cand}$, where V_d denote the vertex set for the text graph G_d[3] and $l \in \ell$. In this way V_d^l indicates the *evidence* we have for label l in the text graph G_d. To illustrate this point consider Figure 2. The label specific features for label $Plant\ Physiology$ is shown in Figure 2 (a) denoted as $f_{plant-physiology}^{cand}$. The term colored in red

[3] G_d is the text graph built for document d using the method outlined in Section 3.2.

indicates the term that is common between $f^{cand}_{plant-physiology}$ and G_d i.e. $V^{plant-physiology}_d$

Having identified the nodes (V^l_d) which should be allocated stronger probabilities in v the next step is to devise a mechanism to determine these probabilities. We experiment with four approaches. In the first approach, referred to as $seed\_nodes\_only$, we allocate all the probability mass in v uniformly to the nodes in V^l_d, all other nodes i.e. nodes $\notin V^l_d$ are assigned zero probability. In the second approach, referred to as the $seed\_and\_eta$ approach, we keep aside a small fraction η of the probability mass, which is distributed uniformly to all the nodes $\notin V^l_d$, the rest of the probability mass i.e. 1-η[4] is uniformly distributed to all nodes $\in V^l_d$. The third approach, referred to as $non\_uniform\_seed\_only$, is similar to the $seed\_nodes\_only$ approach except that in this case the probability mass in v is not allocated uniformly to the nodes in V^l_d. Probability mass is allocated to the nodes in proportion to their importance, as indicated by the weights allocated to the feature in f^{cand}_l by the feature selection method used. As we discuss in Section 3.5 the feature selection methods, which are used for generating label specific feature f^{cand}_l, compute weights for individual features in f^{cand}_l. These weights (e.g mutual information score, t-score) indicate the strength of association between the feature and the label. In the $non\_uniform\_seed\_only$ approach we allocate probability mass to nodes in V^l_d in proportion to their *feature weights*. Finally, in the $non\_uniform\_eta$ approach we distribute the probability mass i.e. 1-η amongst the V^l_d in proportion to their *feature weights*. The left probability mass of η is distributed uniformly amongst other nodes $\notin V^l_d$. Performance of these different configurations are evaluated in Section 4.1.

One shortcoming of the *PTR* approach is that it does not provides a clean mechanism to integrate features from labels other than l which are associated with the document d. The motivation of doing so is to on one hand bias the walk on the text graph towards terms in f^{cand}_l while simultaneously penalizing terms which are in F_{cand} = $\cup_{k \neq l \text{ and } k \in \ell} f^{cand}_k$[5]. As shown in Figure 2 (b) not incorporating this information results in a leakage of scores (indicated using arrows) to nodes not relevant to label l (e.g. *gases*, *sulphur* etc) . In the next section we describe the ***TRDMS*** or ***TextRank using Ranking on Data Manifold with Sinks*** approach which allows us to simultaneously consider both f^{cand}_l and F_{cand} in the same model.

3.4 *TRDMS*: TextRank using Ranking on Data Manifold with Sinks

Algorithm 1: Algorithm for generating label-l specific keyphrases for document d

Data: Document d, label-l specific unigram features f^{cand}_l, unigram features for label categories other than l represented as F_{cand} = $\cup_{k \neq l and\ k \in \ell} f^{cand}_k$
Result: label-l specific keyphrases from document d
1. Build a Text Graph G_d for document d as discussed in Section 3.2. Let w_i indicate the vertices in G_d ;
2. Construct an *affinity matrix* A, where A_{ij} = *sim*(w_i,w_j) if there is an edge linking w_i, w_j in G_d. *sim*(w_i,w_j) indicates similarity between vertices w_i, w_j ;
3. Symmetrically, normalize A as S = $D^{-1/2}AD^{-1/2}$. D is a diagonal matrix matrix with its (*i*,*i*)-element equal to the sum of the *i*-th row A;
4. **while** *(!*converge*(p))* **do**
   Iterate $p(t+1)$ = $\alpha SIp(t)$ + (1-α)y ;
   /* where 0 <α <1 and I is an indicator diagonal matrix with it's (i,i)-element equal to 0 if $w_i \in V^{\neg l}_d$ and 1 otherwise.*/
end
5. Sort the vertices $w_q \in V_q$ in descending order of their scores $p[q]$. Let this ranked list be represented as <T_K>;
6. $kphrase^l_d$ = $kphrase_{gen}$(<T_K>,d) , where $kphrase^l_d$ is the label-l specific keyphrase list for document d;
7. return $kphrase^l_d$;

In this section we describe the *TextRank using Ranking on Data Manifold with Sinks* approach that allows us to simultaneously consider both f^{cand}_l and F_{cand} when extracting label l specific keyphrases from document's d text graph. For ease of exposition we repeat a few notations and introduce some new ones. Let V_d denote the vertex set for the text graph G_d. Vertices for the text graph G_d are represented by w_i where $i \in$ [1..M], M is the number of vertices i.e. M=$|V_d|$. As introduce earlier, V^l_d denotes the

[4]Please note v is a stochastic normalized vector whose elements sum to 1. In our experiments we set η=0.2
[5]Where ℓ indicates the label set associated with document d

intersection of the set V_d with f_l^{cand}, i.e. $V_d^l = V_d \cap f_l^{cand}$. V_d^l indicates the *evidence* we have for label l in the text graph G_d, where $l \in \ell$. These vertices are also referred to as *query nodes* in the ranking on data manifold literature. Let $V_d^{\neg l}$ denote the intersection of the set V_d with F_{cand}, where $F_{cand} = \cup_{k \neq l \text{ and } k \in \ell} f_k^{cand}$ i.e. all the unigram features associated with label categories other than l[6]. These vertices are also referred to as *sink nodes* in the ranking on data manifold literature. All other vertices are indicated by V_d^q, where $V_d^q = V_d \setminus (V_d^{\neg l} \cup V_d^l)$ denote the set of points to be ranked. Let $p{:}V \rightarrow \Re$ denote the ranking function which assigns a ranking score p_i to each vertex w_i in G_d. One can view p as a vector i.e. $p = [p_1,....,p_M]$. A binary vector $y = [y_1,....,y_M]$ is defined in which $y_i = 1$ if $w_i \in V_d^l$ otherwise $y_i = 0$.

Algorithm 1 gives a detailed outline of the *TRDMS* method. This algorithm is based on the algorithm proposed by (Cheng et al., 2011) for ranking on data manifold with sink points. To generate label-l specific keyphrase for document d the algorithm considers document d, label-l specific unigram features f_l^{cand}, and unigram features for labels other than l represented as F_{cand}. It begins by first building a text graph G_d. After this an affinity matrix A is constructed. This is shown in Step 2. The affinity matrix A, which captures the similarity between vertices (terms in the text graph) w_i and w_j, is built using WordNet. We use the popular WordNet::Similarity (Pedersen et al., 2004) package which measures the semantic similarity and relatedness between a pairs of concepts. After symmetrically normalizing A (Step 3) and initializing the *query* and *sink* nodes the scores are propagated till convergence (Step 4). The routine *converge(p)* checks for convergence by comparing the value of p between two consecutive iterations. If there is little or no change in p the routine return *true*. To generate n-gram keyphrases we follow the approach described in Section 3.2. In Step 6 of Algorithm 1 the $kphrase_{gen}$[7] routine is invoked. In order to choose top-k, label-l specific keyphrases for document d one can select the first k elements of the $kphrase_d^l$ list.

3.5 Generating label specific features from a multi-label corpus

As discussed in previous sections the label specific features f_l^{cand} play an important role in the overall ranking process. When searching for label-l specific keyphrases, the unigram features f_l^{cand} helps bias the walk on the document's text graph towards terms that are relevant and central to label l. We also saw that by considering F_{cand} i.e. unigram features belonging to label categories other than l[8] as *sink* nodes prevents *leakage* of the ranking score to terms not relevant or central to l. We show through experiments in Section 4 that this improves the quality of label-l specific keyphrases extracted from document d. In order to generate *label specific features* from a multi-label corpus D we adopt the *problem transformation* approach commonly used in multi-label learning. In this approach the multi-label corpus D is transformed into $\mid \Im \mid$ single-label data sets, where $\Im$ is the set of labels associated with corpus D. Post this transformation any single-*label* feature selection method can be used to extract label l specific features from these single-label data sets. For our setup we experiment with unigram features selected using mutual information and chi-squared based feature selection methods.

4 Experiment

In order to assess the quality of the label-specific keyphrases generated by our system we conduct a manual evaluation of the generated output. Details of this evaluation are provided in Section 4.1. For our experiments we use a subset of the multi-label corpus EUR-Lex[9]. The EUR-Lex text collection is a collection of documents about European Union law. It contains many different types of documents, including treaties, legislation, case-law and legislative proposals, which are labeled with EUROVOC descriptors. A document in this data-set could be associated with multiple EUROVOC descriptors[10]. The data set that was downloaded contained 16k documents documents and 3,993 EUROVOC descriptors.

[6] We do not assume that $f_l^{cand} \cap F_{cand} = \emptyset$

[7] Details of this routine are provided in Section 3.2

[8] In cases where the document is associated with more than one label or category

[9] http://www.ke.tu-darmstadt.de/resources/eurlex

[10] We treat these as labels

Method	*Precision*$^{\text{avg}}$	*Recall*$^{\text{avg}}$	*F-measure*$^{\text{avg}}$
TPP_{baseline}	0.163	0.194	0.177
$PTR_{\text{seed\_nodes\_only}}$	0.169	0.213	0.188
$PTR_{\text{seed\_and\_eta}}$	0.199	0.223	0.210
$PTR_{\text{non\_uniform\_seed\_only}}$	0.203	0.231	0.216
$PTR_{\text{non\_uniform\_eta}}$	0.237	0.257	0.247
TRDMS	0.397	0.387	0.392

Table 1: Keyphrase Extraction Results

We removed labels that were under represented[11] in this data set. We refer to this data set as the $EUR-Lex_{filtered}$ data set. We randomly selected 100 documents from the $EUR-Lex_{filtered}$ data set. Two criteria were considered when selecting these documents (a) Each document should be associated with at least 2 but not more than 3 labels (b) The size of the *evidence* set i.e. $|V_d^l|$ where $V_d^l = V_d \cap f_{cand}^l$ is at least 10% of $|V_d|$, where V_d represents the vertex set of the text graph associated with d. The resulting data set is referred to as the $EUR-Lex_{filtered}^{keyphrase}$ data set. The reason for enforcing these two criteria is the following. Ensuring that a document in $EUR-Lex_{filtered}^{keyphrase}$ has at least 2 labels allows us to experiment with *sink nodes* i.e. F_{cand}. As we discuss in Section 4.1 for each *label* associated with a document, a human evaluator was asked to generates a label specific list of keyphrases. For example, if a document is associated with 3 labels, three label specific keyphrase list had to be generated by the human evaluator. Allowing documents with more than 3 labels makes this process tedious. The reason for putting restriction (b) when building the $EUR-Lex_{filtered}^{keyphrase}$ is explained in Section 4.1.1. For generating label-l specific features we use the approach described in Section 3.5. For our experiments mutual information based feature selection method was used with a feature size of 250 i.e. $|f_l^{cand}| = 250$.

4.1 Label-specific Keyphrase Evaluation

Two graduate students were asked to manually extract label-specific keyphrases for each document in the $EUR-Lex_{filtered}^{keyphrase}$ data set. At most 10 keyphrases could be assigned to each document-label pair. This results in a total of 1721 keyphrases. The Kappa statistics for measuring inter-agreement among the annotation was 0.81. Any annotation conflicts between the two subjects was resolved by a third graduate student. For evaluation, the automatically extracted label-specific keyphrases for a given document were compared with the manually extracted/annotated keyphrases. Before comparing the keyphrase, the words in the keyphrase were converted to their corresponding base form using word stemming. We calculate three evaluation metrics namely Precision, Recall and F-measure for each document-label pair. Precision (P) = $\frac{count_{correct}}{count_{system}}$, Recall (R) = $\frac{count_{correct}}{count_{human}}$ and F-measure (F) = $\frac{2PR}{P+R}$, where $count_{correct}$ is the total number of correct keyphrases extracted by our method, $count_{system}$ is the total number of automatically extracted keyphrases and $count_{human}$ is the total number of keyphrases labeled by the human annotators. These metrics are calculated for each document-label pair in the $EUR-Lex_{filtered}^{keyphrase}$ data set and then averaged to obtain $Precision^{avg}$, $Recall^{avg}$ and $F-measure^{avg}$. These results are shown in Table 1

We compare the performance of our system against the *TextRank with Post-Processing:* $TPP_{baseline}$ baseline which was explained in Section 2. Briefly, in this setup to identify label-l specific keyphrases in document d, we run *TextRank* on document d and filter the generated keyphrase list based on f_l^{cand} i.e. label l specific features. In all setups the document text graph is built in the same fashion i.e. N = 10 and co-occurrence relationship is used to draw edges between nodes in the text graph. For generating the affinity matrix A, which is used in the *TRDMS* method, the *res* semantic similarity method is used[12]. To reiterate, when generating label-l specific keyphrases for document d the *PTR* method only uses f_l^{cand}, whereas the *TRDMS* method uses both f_l^{cand} (as *query* nodes) and $F_{cand} = \cup_{k \neq l \text{ and } k \in \ell} f_k^{cand}$ (where

[11] Any label which occurred less than 10% times in the data set was removed. The documents associated with these labels were also removed from the data set

[12] We experimented with other semantic similarity measures such as *lin* and *jcn*. The *res* measure gave us the best results

ℓ is the set of labels associated with document d) i.e. all the unigram features associated with label categories other than l (as *sink* nodes). One can observe from Table 1 that for *PTR* the *non_uniform_eta* configuration gives the best result. Overall the *TRDMS* approach significantly outperforms all *PTR* configurations and our baseline. This validates our belief that one can significantly improve the quality of extracted keyphrase by not only considering label-l specific features i.e. f_l^{cand} but also features associated with label categories other than l. When we analyzed the performance of *TRDMS* at the document level we observed that the keyphrase extraction metrics for documents which had *strongly correlated labels* e.g. "*tariff_quota*" and "*import_license*" was 9-11% lower than the reported average scores. On the contrary, keyphrase extraction metrics for documents which had labels that had no or weak correlation e.g. "*aid_contract*" and "*import_license*" was 3-5% higher than the reported average scores. One reason for this could be the substantial overlap between f_l^{cand} and F_{cand} for highly correlated labels. This large overlap results in the *query* nodes being considered as *sink* nodes which negatively impacts the score propagation in the underlying text graph.

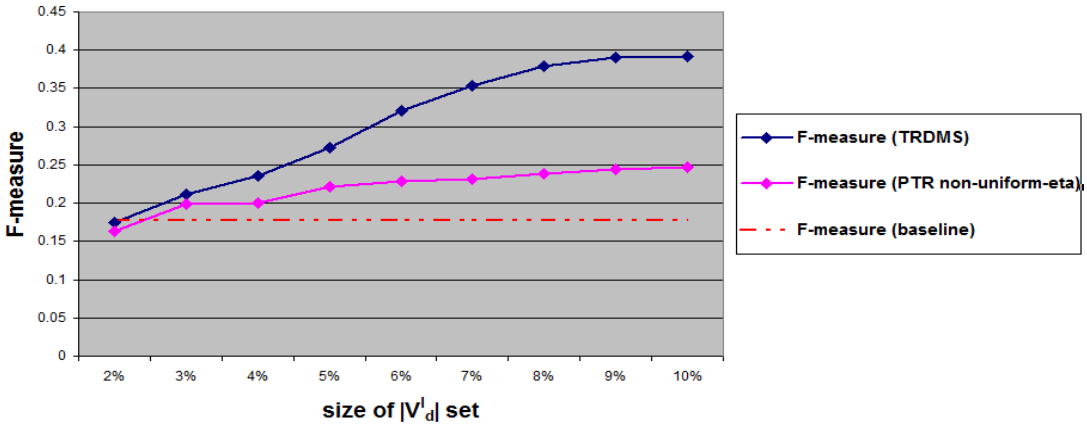

Figure 3: Impact of evidence set size on F-measure (best viewed in color)

4.1.1 Impact of evidence set size ($|V_d^l|$) on keyphrase generation results

To recap, elements in set V_d^l indicate the *evidence* we have for label l in the text graph of document d i.e. G_d. In order to investigate how the size of the evidence set i.e. $|V_d^l|$ impacts the performance of our system the following simulation was carried out. In different setups we randomly drop out elements from V_d^l so that the size of the resulting evidence set ranges from 2% to 10% of $|V_d|$, where $|V_d|$ represents the vertex set size of text graph G_d. We plot the impact this has on the F-measure in Figure 3. One observes that when the *evidence* set size is in the range 2-4% the gains over the $TPP_{baseline}$ baseline (0.177) are low to modest. As the evidence set size increases the gains over the baseline increases substantially.

5 Conclusion and Future Work

In this paper we presented the problem of extracting label specific keyphrases from a document. We pose the problem of extracting such keyphrases from a document as a random walk on a document's text graph. The methods proposed in this paper utilizes the *label specific features*, which are strongly associated with the label, to bias the walk towards terms that are more relevant to the label. We show through experiments that when generating label-l specific keyphrases it helps to consider both label-l specific features and features associated with labels other than l. As future work we would like to further assess the quality of the generated keyphrases by using these keyphrases for generating topic (or label) focused document summaries.

References

Takashi Tomokiyo and Matthew Hurst. 2003. *A language model approach to keyphrase extraction*. Proceedings of the ACL 2003 workshop on Multiword expressions: analysis, acquisition and treatment.

A. Hulth. 2003. *Improved Automatic Keyword Extraction Given More Linguistic Knowledge*. Proceedings of the Conference on Empirical Methods in Natural Language Processing.

Yutaka Matsuo and Mitsuru Ishizuka. 2004. *Keyword Extraction from a Single Document using Word Co-occurrence Statistical Information*. International Journal on Artificial Intelligence Tools.

Peter D. Turney. 2000. *Learning Algorithms for Keyphrase Extraction*. Information Retrieval.

Eibe Frank and W. Gordon Paynter and Ian H. Witten and Carl Gutwin and Craig G. Nevill-Manning. 1999. *Domain-Specific Keyphrase Extraction*. IJCAI '99: Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence.

Peter D. Turney. 2003. *Coherent Keyphrase Extraction via Web Mining*. IJCAI '03: Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence.

Min Song and Il-Yeol Song and Xiaohua Hu. 2003. *KPSpotter: a flexible information gain-based keyphrase extraction system*. Fifth International Workshop on Web Information and Data Management.

Olena Medelyan and Ian H. Witten. 2006. *Thesaurus based automatic keyphrase indexing*. JCDL '06: Proceedings of the 6th ACM/IEEE-CS joint conference on Digital libraries.

George Forman. 2003. *An Extensive Empirical Study of Feature Selection Metrics for Text Classification*. The Journal of Machine Learning Research.

George Forman. 2004. *A pitfall and solution in multi-class feature selection for text classification*. International Conference on Machine Learning.

Rada Mihalcea and Paul Tarau. 2004. *TextRank: Bringing Order into Texts*. Proceedings of EMNLP-04 and the 2004 Conference on Empirical Methods in Natural Language Processing.

Rada Mihalcea, Paul Tarau and Elizabeth Figa. 2004. *PageRank on Semantic Networks, with Application to Word Sense Disambiguation*. COLING.

Rada Mihalcea, Paul Tarau and Elizabeth Figa. 2004. *PageRank on Semantic Networks, with Application to Word Sense Disambiguation*. COLING.

Dengyong Zhou and Jason Weston and Arthur Gretton and Olivier Bousquet and Bernhard SchÃűlkopf. 2004. *Ranking on Data Manifolds*. Advances in Neural Information Processing Systems.

Ted Pedersen and Siddharth Patwardhan and Jason Michelizzi. 2004. *WordNet::Similarity: Measuring the Relatedness of Concepts*. Association for Computational Linguistics.

Rada Mihalcea and Paul Tarau. 2005. *A language independent algorithm for single and multiple document summarization*. In Proceedings of IJCNLPâĂŹ2005.

Xiaojun Wan. 2007. *TimedTextRank: adding the temporal dimension to multi-document summarization*. SIGIR.

Taher H. Haveliwala. 2002. *Topic-sensitive PageRank*. Proceedings of the Eleventh International World Wide Web Conference.

Xue-Qi Cheng and Pan Du and Jiafeng Guo and Xiaofei Zhu and Yixin Chen. 2011. *Ranking on Data Manifold with Sink Points*. IEEE Transactions on Knowledge and Data Engineering.

Jingrui He and Mingjing Li and Hong-Jiang Zhang and Hanghang Tong and Changshu Zhang. 2004. *Manifold-ranking Based Image Retrieval*. Proceedings of the 12th Annual ACM International Conference on Multimedia.

Xiaojun Wan and Jianwu Yang and Jianguo Xiao. 2007. *Manifold-Ranking Based Topic-Focused Multi-Document Summarization*. IJCAI.