Random Walks on Text Structures

Rada Mihalcea

University of North Texas,
Computer Science Department
rada@cs.unt.edu

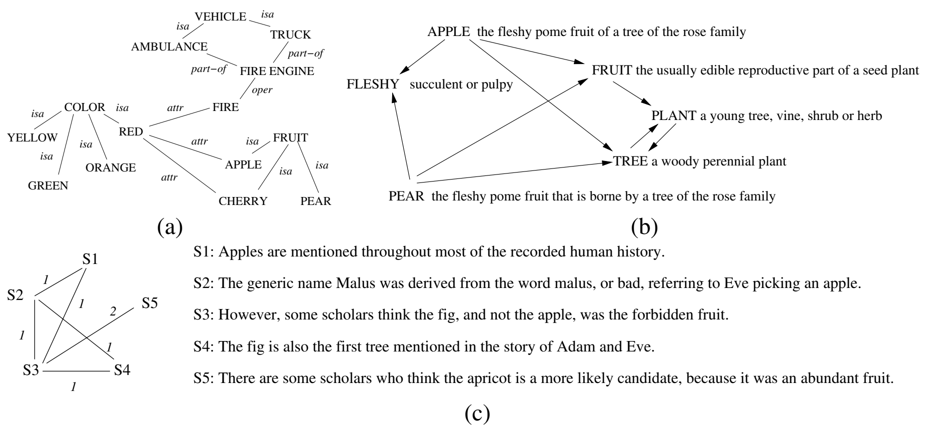
Abstract. Since the early ages of artificial intelligence, associative or semantic networks have been proposed as representations that enable the storage of language units and the relationships that interconnect them, allowing for a variety of inference and reasoning processes, and simulating some of the functionalities of the human mind. The symbolic structures that emerge from these representations correspond naturally to graphs – relational structures capable of encoding the meaning and structure of a cohesive text, following closely the associative or semantic memory representations. The activation or ranking of nodes in such graph structures mimics to some extent the functioning of human memory, and can be turned into a rich source of knowledge useful for several language processing applications. In this paper, we suggest a framework for the application of graph-based ranking algorithms to natural language processing, and illustrate the application of this framework to two traditionally difficult text processing tasks: word sense disambiguation and text summarization.

1 Introduction

Many language processing applications can be modeled by means of a graph. These data structures have the ability to encode in a natural way the meaning and structure of a cohesive text, and follow closely the associative or semantic memory representations. For instance, Figure 1 shows examples of graph representations of textual units[1] and the relationships that interconnect them: 1(a) (adapted from [6]) shows a network of concepts related by semantic relations – simulating a fragment of human memory, on which reasoning and inferences about various concepts represented in the network can be performed; 1(b) shows a network with similar structure, this time automatically derived via definitional links in a dictionary; finally, 1(c) is a graph representation of the cohesive structure of a text, by encoding similarity relationships between textual units.

Provided a graph representation of the text, algorithms for the activation or ranking of nodes in such structures can be used to simulate the functioning of human memory, consequently resulting in solutions for a variety of natural language processing tasks that can be modeled by means of a graph. In this paper, we suggest a framework for the application of graph-based ranking algorithms to text-based graph structures, and show how two text processing applications: word sense disambiguation and text summarization, can find successful solutions within this framework.

[1] We use the term *textual unit* to refer to the textual representation of a *cognitive unit* as defined by Anderson [1]. It can be a word, a concept, a sentence, or any other unit that can find a representation in language.


VEHICLE
TRUCK
AMBULANCE
FIRE ENGINE
FIRE
COLOR
YELLOW
GREEN
ORANGE
RED
FRUIT
APPLE
CHERRY
PEAR
isa
part–of
oper
attr
(a)
APPLE the fleshy pome fruit of a tree of the rose family
FLESHY succulent or pulpy
FRUIT the usually edible reproductive part of a seed plant
PLANT a young tree, vine, shrub or herb
TREE a woody perennial plant
PEAR the fleshy pome fruit that is borne by a tree of the rose family
(b)
S1
S2
S3
S4
S5
1
2
(c)

S1: Apples are mentioned throughout most of the recorded human history.

S2: The generic name Malus was derived from the word malus, or bad, referring to Eve picking an apple.

S3: However, some scholars think the fig, and not the apple, was the forbidden fruit.

S4: The fig is also the first tree mentioned in the story of Adam and Eve.

S5: There are some scholars who think the apricot is a more likely candidate, because it was an abundant fruit.

Fig. 1. Graph representations of textual units and relationships that interconnect them

2 Background

Although initiated from different theoretical backgrounds, there is nonetheless a close relation between the current graph-based ranking algorithms (first introduced in graph-theory), and the earlier models of spreading activation (due to cognitive psychology), which stands primarily in their common underlying graph representations and their fundamental idea of exploiting flow over a network.

The idea of associative networks as mental representations for cognitive units and the relationships between them goes back to early work in psychology [9] and psycholinguistics [27], [29]. Theories of semantic or associative memory [26] have initially emerged in cognitive psychology as models for human language representation and reasoning, and since then have been applied to a variety of computer-based applications.

Spreading activation can be regarded as an earlier version of current graph-based ranking algorithms. It refers to network[2]-based models where activation started from *one or more* source nodes is propagated over the network, activating more and more nodes, until a certain termination condition is met (usually the distance from the source nodes). In these models, it is the *activation* itself that matters, and thus the information recorded at node level is whether the node is *active* or *passive*.

In the more recently introduced graph-based ranking models (or node-ranking models), the propagation of flow over the network starts from *all* the nodes simultaneously, and runs repeatedly throughout the network until a stable state is achieved (convergence). In these models, the information recorded at node level is the *rank* of the node relative to all other nodes in the network.

Despite their potential appealing connection to models of human memory, the large scale application of spreading activation and graph-based ranking models to text processing tasks has been limited for various reasons: Early work in spreading activation methods was hindered by the complexity of underlying structures (e.g. entire semantic

[2] The terms *network* and *graph* are used interchangeably, a graph being the computer-based representation of a network.

networks), and as a result only few applications with small scale evaluations have been attempted. On the other hand, recent research in graph-based ranking algorithms has focused on social networks and Web-link analysis, and their application to text processing has not been explored.

3 Related Work in Natural Language Processing

Starting with the seminal work of Quillian on theories of semantic memory [26], associative or semantic networks and spreading activation processes have been used as the underlying model for several applications in language processing, including word sense disambiguation, automated reasoning, text generation, information retrieval, text summarization, and others.

A major impediment faced by early work in this area was the complexity of the resulting structures, which sometimes was not supported by the underlying computer hardware (although attempts were made to overcome these hardware limitations with parallel architectures for semantic processing such as SNAP [24]). This fact has consequently resulted in limitations on the size of applications attempted with these approaches. For similar reasons, the evaluation of such algorithms was most of the times performed on toy-size problems (e.g. Quillian has evaluated his proposed word sense disambiguation algorithm on nineteen ambiguous words [26]), and the scalability of the models was rarely, if ever, evaluated.

Spreading activation was previously used as a method for solving lexical ambiguity of words [11] through simultaneous identification of word senses and frame case slots. Another related line of work is the algorithm proposed in [31], where a large neural network is built by relating words through their dictionary definitions. Spreading activation was also suggested as a means for dictionary access [33], using a method that simulates processes of human mind, thus improving over other traditional ways for dictionary look-up. The application of spreading activation algorithms was also tested in information retrieval, in a monolingual domain specific environment [5], or more general multilingual environments for cross language information retrieval [2].

More recently, graph-based ranking algorithms (e.g. HITS [13] or PageRank [3]) have been successfully used in citation analysis, social networks [7], and the analysis of the link-structure of the World Wide Web [3]. A node ranking algorithm relying on PageRank [3] and the ArcRank extension [12] was used as a method for thesaurus construction starting with electronic dictionaries [12].

In recent work, we have shown how graph-based ranking algorithms designed for content-independent Web link analysis can be turned into a useful source of information for language processing tasks when applied to graphs extracted from natural language texts – with encouraging results on the problem of word sense disambiguation [21], sentence ranking and extraction for text summarization [17], and selection of important terms in a text [19]. The PageRank algorithm was also evaluated in a comparative study of coherence algorithms [32], where it was found to exceed the performance of other paragraph and word oriented algorithms for sentence ranking. Finally, the same ranking algorithm was integrated in an event-centric approach to summarization [30], where it was used to identify important elements (events or entities) in a text.

4 General Framework

We suggest a framework targeted to the application of graph-based ranking models to text processing tasks. Several new methods and representations are described, specifically tailored for the application of this framework to natural language processing. In Section 6, we describe two applications that can be successfully addressed within this framework.

4.1 A New Graph-Based Representation of Text

To enable the application of graph-based ranking algorithms to natural language texts, we have to build a graph that represents the text, and interconnects words or other text entities with meaningful relations. The graphs constructed in this way are centered around the target text, but can be extended with external graphs, such as off-the-shelf semantic or associative networks (e.g. WordNet [22]), or other similar structures automatically derived from large corpora.

Representation
Graph Nodes: Depending on the application at hand, text units of various sizes and characteristics can be added as vertices in the graph, e.g. words, collocations, word-senses, entire sentences, entire documents, or others. Note that the graph-nodes do not have to belong to the same category.

Graph Edges: Similarly, it is the application that dictates the type of relations that are used to draw connections between any two such vertices, e.g. lexical or semantic relations, measures of text cohesiveness, contextual overlap, membership of a word in a sentence, and others.

Algorithm
Regardless of the type and characteristics of the elements added to the graph, the application of the ranking algorithms to natural language texts consists of the following main steps:

1. Identify text units that best define the task at hand, and add them as vertices in the graph.
2. Identify relations that connect such text units, and use these relations to draw edges between vertices in the graph. Edges can be directed or undirected, weighted or unweighted.
3. Apply a graph-based ranking algorithm to find a ranking over the nodes in the graph. Iterate the graph-based ranking algorithm until convergence. Sort vertices based on their final score. Use the values attached to each vertex for ranking/selection decisions.

5 Graph-Based Ranking Algorithms

Graph-based ranking algorithms are essentially a way of deciding the importance (or "power") of a vertex within a graph, based on information drawn from the graph structure. The basic idea implemented by a graph-based ranking model is that of "voting" or “recommendation”. When one vertex links to another one, it is basically casting a vote for that other vertex. The higher the number of votes that are cast for a vertex, the higher the importance of the vertex. Graph-based ranking algorithms have been successfully used in citation analysis, social networks [7], and content-independent Web-link analysis [3].

These graph ranking algorithms are based on a random walk model, where a walker takes random steps on the graph G, with the walk being modeled as a Markov process – that is, the decision on what edge to follow is solely based on the vertex where the walker is currently located. Under certain conditions, this model converges to a stationary distribution of probabilities, associated with vertices in the graph, representing the probability of finding the walker at a certain vertex in the graph. Based on the Ergodic theorem for Markov chains [10], the algorithms are guaranteed to converge if the graph is both aperiodic and irreducible. The first condition is achieved for any graph that is a non-bipartite graph, while the second condition holds for any strongly connected graph. Both these properties are typically achieved for the text-based graphs constructed for the language processing applications considered in this work.

Let $G = (V, E)$ be a directed graph with the set of vertices V and set of edges E, where E is a subset of $V \times V$. For a given vertex V_i, let $In(V_i)$ be the set of vertices that point to it (predecessors), and let $Out(V_i)$ be the set of vertices that vertex V_i points to (successors). We describe below two graph-based ranking algorithms:

HITS (Hyperlinked Induced Topic Search) [13] is an iterative algorithm that was designed for ranking Web pages according to their degree of “authority”. The HITS algorithm makes a distinction between “authorities” (pages with a large number of incoming links) and “hubs” (pages with a large number of outgoing links). For each vertex, HITS produces two sets of scores – an “authority” score, and a “hub” score:

$$HITS_A(V_i) = \sum_{V_j \in In(V_i)} HITS_H(V_j) \tag{1}$$

$$HITS_H(V_i) = \sum_{V_j \in Out(V_i)} HITS_A(V_j) \tag{2}$$

PageRank [3] is perhaps one of the most popular ranking algorithms, and was designed as a method for Web link analysis. Unlike other ranking algorithms, PageRank integrates the impact of both incoming and outgoing links into one single model, and therefore it produces only one set of scores:

$$PR(V_i) = (1 - d) + d * \sum_{V_j \in In(V_i)} \frac{PR(V_j)}{|Out(V_j)|} \tag{3}$$

where d is a parameter that can be set between 0 and 1 [3] In matrix notation, the $PageRank$ vector of stationary probabilities is the principal eigenvector for the matrix A_{row}, which is obtained from the adjacency matrix A representing the graph, with all rows normalized to sum to 1: ($P = A_{row}^T P$).

[3] The damping factor d has the role of integrating into the model the probability of jumping from a given vertex to another random vertex in the graph. The factor d is usually set at 0.85 [3].

A ranking process starts by assigning arbitrary values to each node in the graph, followed by several iterations until convergence below a given threshold is achieved. Convergence is achieved when the error rate for any vertex in the graph falls below a given threshold, where the error rate of a vertex V_i is approximated with the difference between the scores computed at two successive iterations: $S^{k+1}(V_i) - S^k(V_i)$ (usually after 25-35 iteration steps). After running the algorithm, a score is associated with each vertex, which represents the "importance" (*rank*) of the vertex within the graph. Note that for such iterative algorithms, the final value obtained for each vertex is not affected by the choice of the initial value, only the number of iterations to convergence may be different.

The basic graph-based ranking framework can be improved with representations specifically tailored to language processing tasks. We describe below two such improvements, consisting of the application of graph-based ranking to *undirected* and *weighted* graphs.

Undirected Graphs. Although traditionally applied on directed graphs, algorithms for node activation or ranking can be also applied to undirected graphs. In such graphs, convergence is usually achieved after a larger number of iterations, and the final ranking can differ significantly compared to the ranking obtained on directed graphs.

Weighted Graphs. When the graphs are built from natural language texts, they may include multiple or partial links between the units (vertices) that are extracted from text. It may be therefore useful to indicate and incorporate into the model the "strength" of the connection between two vertices V_i and V_j as a weight w_{ij} added to the corresponding edge that connects the two vertices. Consequently, we introduce new formulae for graph-based ranking that take into account edge weights when computing the score associated with a vertex in the graph, e.g.

$$PR^W(V_i) = (1-d) + d * \sum_{V_j \in In(V_i)} w_{ji} \frac{PR^W(V_j)}{\sum\limits_{V_k \in Out(V_j)} w_{jk}} \tag{4}$$

Similar weighted versions can be defined for all other ranking algorithms, as also shown in our previous work [17]. The final vertex scores (and therefore rankings) for weighted graphs can differ significantly as compared to their unweighted alternatives.

6 Graph-Based Ranking Algorithms for Text Processing

Many natural language processing applications can be modeled by means of a graph, and thus the framework suggested in this paper can provide solutions for many important text processing problems. In this paper, we specifically address the application of graph-based ranking algorithms to text processing at two different levels of granularity: *sentence level* and *document level*. We show how the graph-based ranking algorithms can be applied to: (1) text processing at sentence level, where we specifically address the problem of word sense disambiguation; and (2) text processing at document level, with a focus on the problem of extractive summarization.

6.1 Text Processing at Sentence Level: Unsupervised Word Sense Disambiguation

The task of word sense disambiguation consists of assigning the most appropriate meaning to a polysemous word within a given context. To enable the application of algorithms for graph-based ranking to the disambiguation of all words in unrestricted text, we have to build a graph that represents the text and interconnects the words with meaningful relations.

We start by first formulating the word sense disambiguation problem as a sequence data labeling problem. Note that this formulation applies not only to word sense disambiguation, but also to other labeling problems, e.g. part-of-speech tagging, named entity resolution, etc. Given a sequence of words $W = \{w_1, w_2, ..., w_n\}$, each word w_i with corresponding admissible labels $L_{w_i} = \{l^1_{w_i}, l^2_{w_i}, ..., l^{N_{w_i}}_{w_i}\}$, we define a label graph G = (V,E) such that there is a vertex $v \in V$ for every possible label $l^j_{w_i}$, $i = 1..n$, $j = 1..N_{w_i}$. Dependencies between pairs of labels are represented as directed or indirected edges $e \in E$, defined over the set of vertex pairs $V \times V$. Such label dependencies can be learned from annotated data, or derived by other means, e.g. by measuring similarity between instances (see Figure 2 for an example). Note that the graph does not have to be fully connected, as not all label pairs can be related by a dependency.

Given such a label graph associated with a sequence of words, the likelihood of each label can be determined using an iterative graph-based ranking algorithm, which runs over the graph of labels and identifies the importance of each label (vertex) in the graph. We use the weighted version of the ranking algorithms, as they prove particularly useful for sequence data labeling, since the dependencies between pairs of sense labels are more naturally modeled through weights indicating their strength, rather than using binary $0/1$ values. Intuitively, the stationary probability associated with a vertex in the graph represents the probability of finding the walker at that vertex during the random walk, and thus it represents the importance of the vertex within the graph. In the context of sequence data labeling, the random walk is performed on the label graph associated with a sequence of words, and thus the resulting stationary distribution of probabilities can be used to decide on the most probable set of senses for the given sequence. Through the label graphs it builds for a given sequence of words, the algorithm exploits relations between word labels, and implements a concept of *recommendation*. A label recommends other related labels, and the strength of the recommendation is recursively computed based on the importance of the labels making the recommendation. In this way, the algorithm simultaneously annotates all the words in an input sequence, by identifying the most probable (most *recommended*) set of labels.

Given a sequence of words with their corresponding admissible labels, the algorithm for sequence data labeling seeks to identify a graph of label dependencies on which a random walk can be performed, resulting into a set of scores that can be used for label assignment. The algorithm consists of three main steps: (1) construction of label dependencies graph; (2) label scoring using graph-based ranking algorithms; (3) label assignment. First, a weighted graph of label dependencies is built, by adding a vertex for each admissible label, and an edge for each pair of labels for which a dependency is identified. A maximum allowable distance can be set ($MaxDist$), indicating a constraint over the distance between words for which a label dependency is sought. For instance, if $MaxDist$ is set to 3, no edges will be drawn between labels corresponding to words that are more than three words apart. Label dependencies are determined through a $Dependency$ function, whose definition depends on the application and type of resources available (e.g. dictionary definitions, annotated corpora, etc.). Next, scores are assigned to vertices using a graph-based ranking algorithm. Finally, the most probable set of labels is determined by identifying for each word the label that has the highest score. Note that all admissible labels corresponding to the words in the input sequence are assigned with a score, and thus the selection of two or more most probable labels for a word is also possible.

For the particular application of word sense disambiguation, we need information on labels (word senses) and dependencies (word sense dependencies). Word senses can be easily obtained from any sense inventory, e.g. WordNet or LDOCE, or any other machine readable dictionary.

Sense dependencies can be derived in various ways, depending on the type of resources available for the language and/or domain at hand. If only a dictionary is available, a sense dependency can be defined as a measure of similarity between word senses. There are several metrics that can be used for this purpose, see for instance [4] for an overview. If sense annotated corpora are available, the similarity between two word senses can be measured as a similarity between their corresponding feature vectors, where a feature vector could include features traditionally used for this task, e.g. surrounding words and their parts of speech, syntactic dependencies, keywords in context, etc. Word sense similarities can be also derived starting with raw corpora, by bootstrapping starting with a small set of labeled examples, or in a completely unsupervised fashion, through latent semantic analysis [14].

An Example. Consider the task of assigning senses to the words in the text *The church bells no longer rung on Sundays*[4]. For the purpose of illustration, we assume at most three senses for each word, which are shown in Figure 2. Word senses and definitions are obtained from the WordNet sense inventory [22]. All word senses are added as vertices in the label graph, and weighted edges are drawn as dependencies among word senses, derived using a definition-based similarity measure inspired from the Lesk algorithm [15]. The resulting label graph is an undirected weighted graph, as shown in Figure 2. After running the ranking algorithm, scores are identified for each word-sense in the graph, indicated between brackets next to each node. Selecting for each word the sense with the largest score results into the following sense assignment: *The church#2 bells#1 no longer rung#3 on Sundays#1*, which is correct according to annotations performed by professional lexicographers.

To evaluate the application of the graph ranking algorithms to word sense disambiguation, we implemented a sense relatedness measure based on definition overlap, used as an indicator of the dependency between sense labels. Given two word senses and their corresponding definitions, the sense similarity is determined as a function of definition overlap, measured as the number of common tokens between the two definitions, after running them through a simple filter that eliminates all stop-words. To avoid

[4] Example drawn from the data set provided during the SENSEVAL-2 English all-words task [25]. Manual sense annotations were also made available for this data.

church
1: one of the groups of Christians who have their own beliefs and form of worship
2: a place for public (especially Christian) worship
3: a service conducted in a church

bell
1: a hollow device made of metal that makes a ringing sound when struck
2: a push button at an outer door that gives a ringing or buzzing signal when pushed
3: the sound of a bell

ring
1: make a ringing sound
2: ring or echo with sound
3: make (bells) ring, often for the purposes of musical edification

Sunday
1: first day of the week; observed as a day of rest and worship by most Christians

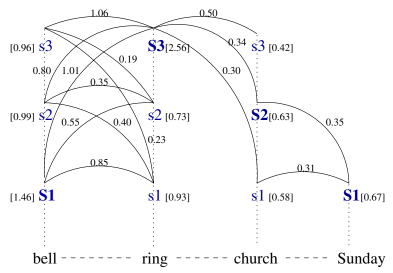


Fig. 2. Label graph for assigning senses to words in the sentence *The church bells no longer rung on Sundays*

promoting long definitions, we also use a normalization factor, and divide the content overlap of the two definitions with the length of each definition. This sense similarity measure is inspired by the definition of the Lesk algorithm [15].

The algorithm was primarily evaluated on the SENSEVAL-2 English all-words data set (and therefore sense distinctions are performed with respect to WordNet [22]), but evaluations were also run on other data sets. The performance of the algorithm is compared with the disambiguation accuracy obtained with a variation of the Lesk algorithm [15], which selects the meaning of an open-class word by finding the word sense that leads to the highest overlap between the corresponding dictionary definition and the current context. We thus compare the performance of sequence data labeling, which takes into account label dependencies and the flow of "importance" over them as implemented by the graph-based ranking algorithms, with individual data labeling, where a label is selected independent of the other labels in the text. Note that both algorithms rely on the same knowledge source, i.e. dictionary definitions, and thus they are directly comparable. Moreover, none of the algorithms take into account the dictionary sense order (e.g. the most frequent sense information provided in WordNet), and therefore they are both fully unsupervised.

Table 1 shows disambiguation results using: (a) sequence data labeling with iterative graph-based algorithms; (b) individual data labeling with a version of Lesk al-

Table 1. Disambiguation accuracy for graph-based sequence data labeling (SENSEGRAPH) and individual data labeling (LESK) on the SENSEVAL-2 data set

	GRAPH RANKING		SENSEGRAPH	
Part-of-speech	Precision	Recall	Precision	Recall
Noun	61.53%	28.86%	54.75%	25.68%
Verb	38.97%	8.75%	32.90%	7.39%
Adjective	61.05%	11.51%	53.39%	10.07%
Adverb	66.66%	7.84%	63.50%	7.47%
ALL	**56.97%**	**56.97%**	50.61%	50.61%

gorithm; and (c) random baseline. A baseline for this fully unsupervised setting consists of a random selection of senses, which results into a precision and recall of 37.9%.

The accuracy of the graph-based sequence data labeling algorithm exceeds by a large margin the individual data labeling algorithm, resulting into 12.87% error rate reduction, which is statistically significant ($p < 0.0001$, paired t-test). Performance improvements are equally distributed across all parts-of-speech, with comparable improvements obtained for nouns, verbs, and adjectives. Additional details on the algorithm, as well as evaluations on other data sets are reported in [18].

6.2 Text Processing at Document Level: Extractive Summarization

Another text processing application that finds an elegant solution within the graph-based ranking framework is the selection of sentences that are informative for the overall understanding of a given text. Iterative graph-based algorithms have the ability to identify important sentences based on the cohesive structure of a text, by taking into account the connections between sentences in a text.

For this task, the goal is to rank entire sentences, and therefore a *vertex* is added to the graph for each sentence in the text. To draw *edges* between vertices, we are defining a similarity relation, where “similarity” can be defined in various ways. In the experiments described in this paper, we use a simple cosine similarity based on a measure of text overlap. Such a relation between two sentences can be seen as a process of recommendation: a sentence that addresses certain concepts in a text, gives the reader a recommendation to refer to other sentences in the text that address the same or similar concepts. The resulting graph is highly connected, with a weight associated with each edge, and thus we use again the weighted version of the graph algorithms. The graph can be represented as: (a) simple *undirected* graph; (b) directed weighted graph with the orientation of edges set from a sentence to sentences that follow in the text (*directed forward*); or (c) directed weighted graph with the orientation of edges set from a sentence to previous sentences in the text (*directed backward*).

An Example. Figure 3 shows a text sample, and the associated weighted graph constructed for this text. The figure also shows sample weights attached to the edges connected to vertex 9, and the final score computed for each vertex, using the PageRank algorithm, applied on an undirected graph. The sentences with the highest rank are selected for inclusion in the abstract. For this sample article, sentences with id-s 9, 15, 16, 18 are extracted, resulting in a summary of about 100 words, which according to automatic evaluation measures, is ranked the second among summaries produced by 15 other systems.

We evaluate the sentence extraction algorithm in the context of a single-document summarization task [17], using 567 news articles provided during the Document Understanding Evaluations 2002 [8]. For each article, we generate a 100-words summary, by taking the sentences with the highest rank according to the graph-based ranking algorithm. For evaluation, we use the ROUGE toolkit, which is a method based on Ngram

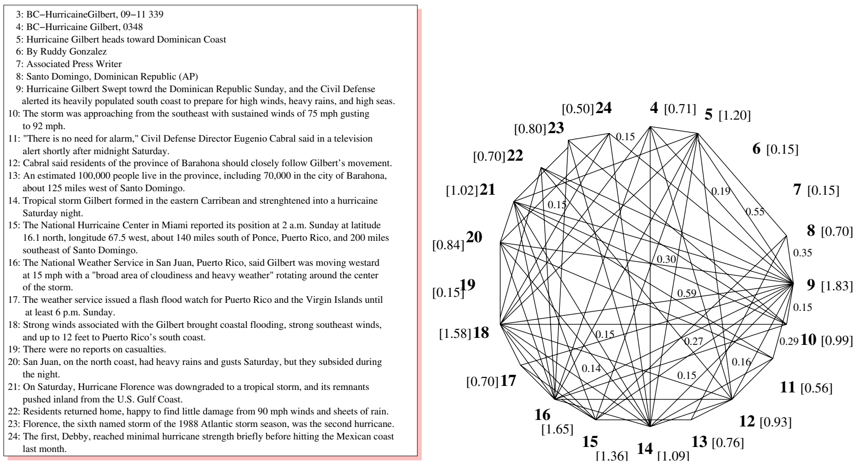

3: BC−HurricaineGilbert, 09−11 339
4: BC−Hurricaine Gilbert, 0348
5: Hurricaine Gilbert heads toward Dominican Coast
6: By Ruddy Gonzalez
7: Associated Press Writer
8: Santo Domingo, Dominican Republic (AP)
9: Hurricaine Gilbert Swept towrd the Dominican Republic Sunday, and the Civil Defense alerted its heavily populated south coast to prepare for high winds, heavy rains, and high seas.
10: The storm was approaching from the southeast with sustained winds of 75 mph gusting to 92 mph.
11: "There is no need for alarm," Civil Defense Director Eugenio Cabral said in a television alert shortly after midnight Saturday.
12: Cabral said residents of the province of Barahona should closely follow Gilbert's movement.
13: An estimated 100,000 people live in the province, including 70,000 in the city of Barahona, about 125 miles west of Santo Domingo.
14. Tropical storm Gilbert formed in the eastern Carribean and strenghtened into a hurricaine Saturday night.
15: The National Hurricaine Center in Miami reported its position at 2 a.m. Sunday at latitude 16.1 north, longitude 67.5 west, about 140 miles south of Ponce, Puerto Rico, and 200 miles southeast of Santo Domingo.
16: The National Weather Service in San Juan, Puerto Rico, said Gilbert was moving westard at 15 mph with a "broad area of cloudiness and heavy weather" rotating around the center of the storm.
17. The weather service issued a flash flood watch for Puerto Rico and the Virgin Islands until at least 6 p.m. Sunday.
18: Strong winds associated with the Gilbert brought coastal flooding, strong southeast winds, and up to 12 feet to Puerto Rico's south coast.
19: There were no reports on casualties.
20: San Juan, on the north coast, had heavy rains and gusts Saturday, but they subsided during the night.
21: On Saturday, Hurricane Florence was downgraded to a tropical storm, and its remnants pushed inland from the U.S. Gulf Coast.
22: Residents returned home, happy to find little damage from 90 mph winds and sheets of rain.
23: Florence, the sixth named storm of the 1988 Atlantic storm season, was the second hurricane.
24: The first, Debby, reached minimal hurricane strength briefly before hitting the Mexican coast last month.

Fig. 3. Sample graph build for extractive summarization from a newspaper article

Table 2. Summarization results using a graph-based ranking approach to sentence extraction

Algorithm	Graph		
	Undirected	Dir.Forward	Dir.Backward
$HITS_A^W$	0.4912	0.4584	**0.5023**
$HITS_H^W$	0.4912	**0.5023**	0.4584
$PageRank$	0.4904	0.4202	**0.5008**

Top five DUC 2002 systems [8]					Baseline
S27	S31	S28	S21	S29	
0.5011	0.4914	0.4890	0.4869	0.4681	0.4799

statistics, found to be highly correlated with human evaluations [16]. Two manually produced reference summaries are provided, and used in the evaluation process[5].

The summaries are evaluated using two graph-based ranking algorithms: HITS and PageRank, and Table 2 shows the results obtained with each algorithm, when using graphs that are: (a) undirected, (b) directed forward (with the orientation of edges set from a given sentence to sentences that follow in the text), or (c) directed backward (reversed orientation). For a comparative evaluation, the table also shows the results obtained on this data set by the top five (out of 15) performing systems participating in the single document summarization task at DUC 2002 [8]. It also lists the baseline performance, computed for 100-word summaries generated by taking the first sentences in each article.

The graph-based algorithms succeed in identifying the most important sentences in a text based on information exclusively drawn from the text itself. Unlike other su-

[5] The evaluation is done using the Ngram(1,1) setting of ROUGE, which was found to have the highest correlation with human judgments, at a confidence level of 95%. Only the first 100 words in each summary are considered.

pervised systems, which attempt to learn what makes a good summary by training on collections of summaries built for other articles, these algorithms are fully unsupervised, and rely only on the given text to derive an extractive summary, which represents a summarization model closer to what humans are doing when producing an abstract for a given document.

Another interesting aspect is that these algorithms provide a ranking over all sentences in a text – which means they can be easily adapted to extracting very short summaries (headlines consisting of one sentence), or longer more explicative summaries, consisting of more than 100 words. Finally, since the algorithms do not require any training corpora, they can be adapted to other languages and domains. In fact, we have recently shown they can be successfully applied to the summarization of texts in Portuguese, without any changes in the algorithm [20].

7 Conclusions

In this paper, we suggested a framework for the application of graph-based ranking algorithms to natural language processing problems. Inspired by early work on spreading activation, random walk algorithms have been traditionally and successfully applied to structured data, such as graphs of Web links or social networks, and much less to graphs derived from unstructured texts.

We described two text processing applications that were shown to find successful solutions within the suggested framework: (1) a sentence level text processing application targeting the resolution of the semantic ambiguity of all words in unrestricted text (word sense disambiguation), and (2) a document level text processing application, targeting the ranking of sentences in a text based on their importance for the overall understanding of the text (extractive summarization). Through evaluations performed on standard benchmarks, the accuracy achieved on both applications using the graph-based ranking algorithms was shown to be competitive with that of previously proposed state-of-the-art methods. An important aspect of these algorithms is that they do not require deep linguistic knowledge, nor domain or language specific annotated corpora, which makes them highly portable to other domains, genres, or languages.

References

1. ANDERSON, J. A spreading activation theory of memory. *Journal of Verbal Learning and Verbal Behavior 22* (1983).
2. BERGER, H., DITTENBACH, M., AND MERKL, D. An adaptive information retrieval system based on associative networks. In *Proceedings of the first Asian-Pacific conference on Conceptual modelling* (Dunedin, New Zealand, 2004).
3. BRIN, S., AND PAGE, L. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems 30*, 1–7 (1998).
4. BUDANITSKY, A., AND HIRST, G. Semantic distance in wordnet: An experimental, application-oriented evaluation of five measures. In *Proceedings of the NAACL Workshop on WordNet and Other Lexical Resources* (Pittsburgh, June 2001).
5. COHEN, P., AND KJELDSEN, R. Information retrieval by constrained spreading activation in semantic networks. *Information Processing and Management 23*, 4 (July 1987).

6. Collins, A. M., and Loftus, E. A spreading-activation theory of semantic processing. *Psychological Review 82*, 6 (1975).
7. Dom, B., Eiron, I., Cozzi, A., and Shang, Y. Graph-based ranking algorithms for e-mail expertise analysis. In *Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery* (San Diego, California, 2003).
8. DUC. Document understanding conference 2002, 2002. http://www-nlpir.nist.gov/projects/duc/.
9. Freud, S. *Psychopathology of everyday life*. Payot, 1901.
10. Grimmett, G., and Stirzaker, D. *Probability and Random Processes*. Oxford University Press, 1989.
11. Hirst, G. Resolving lexical ambiguity computationally with spreading activation and Polaroid words. In *Lexical Ambiguity Resolution*, S. Small, G. Cottrell, and M. Tanenhaus, Eds. Morgan Kaufmann, 1988.
12. Jannink, J. *A Word Nexus for Systematic Interoperation of Semantically Heterogeneous Data Sources*. PhD thesis, Stanford University, 2001.
13. Kleinberg, J. Authoritative sources in a hyperlinked environment. *Journal of the ACM 46*, 5 (1999), 604–632.
14. Landauer, T. K., Foltz, P., and Laham, D. Introduction to latent semantic analysis. *Discourse Processes 25* (1998).
15. Lesk, M. Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from an ice cream cone. In *Proceedings of the SIGDOC Conference 1986* (Toronto, June 1986).
16. Lin, C., and Hovy, E. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of Human Language Technology Conference (HLT-NAACL 2003)* (Edmonton, Canada, May 2003).
17. Mihalcea, R. Graph-based ranking algorithms for sentence extraction, applied to text summarization. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Lingusitics (ACL 2004) (companion volume)* (Barcelona, Spain, 2004).
18. Mihalcea, R. Large vocabulary unsupervised word sense disambiguation with graph-based algorithms for sequence data labeling. In *Proceedings of the Human Language Technology / Empirical Methods in Natural Language Processing conference* (Vancouver, 2005).
19. Mihalcea, R., and Tarau, P. TextRank – bringing order into texts. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2004)* (Barcelona, Spain, 2004).
20. Mihalcea, R., and Tarau, P. An algorithm for language independent single and multiple document summarization. In *Proceedings of the International Joint Conference on Natural Language Processing (IJCNLP-2005)* (Korea, 2005).
21. Mihalcea, R., Tarau, P., and Figa, E. PageRank on semantic networks, with application to word sense disambiguation. In *Proceedings of the 20st International Conference on Computational Linguistics (COLING 2004)* (Geneva, Switzerland, 2004).
22. Miller, G. Wordnet: A lexical database. *Communication of the ACM 38*, 11 (1995), 39–41.
23. Miller, G., Leacock, C., Randee, T., and Bunker, R. A semantic concordance. In *Proceedings of the 3rd DARPA Workshop on Human Language Technology* (Plainsboro, New Jersey, 1993).
24. Moldovan, D., Lee, W., and Lin, C. Parallel knowledge processing on SNAP. *IEEE Transactions on Knowledge and Data Engineering 5*, 1 (1993).
25. Palmer, M., Fellbaum, C., Cotton, S., Delfs, L., and Dang, H. English tasks: all-words and verb lexical sample. In *Proceedings of ACL/SIGLEX Senseval-2* (Toulouse, France, 2001).
26. Quillian, M. Semantic memory. In *Semantic Information Processing*, M. Minsky, Ed. MIT Press, 1968.

27. SCHVANEVELDT, R. *Pathfinder Associative networks: studies in knowledge organization.* Norwood, 1989.
28. SNYDER, B., AND PALMER, M. The English all-words task. In *Proceedings of ACL/SIGLEX Senseval-3* (Barcelona, Spain, July 2004).
29. SPITZER, M. *The mind within the net: models of learning, thinking, and acting.* MIT Press, 1999.
30. VANDERWENDE, L., BANKO, M., AND MENEZES, A. Event-centric summary generation. In *Proceedings of the Document Understanding Conference* (2004).
31. VERONIS, J., AND IDE, N. Word sense disambiguation with very large neural networks extracted from machine readable dictionaries. In *Proceedings of the 13th International Conference on Computational Linguistics (COLING 1990)* (Helsinki, Finland, August 1990).
32. WOLF, F., AND GIBSON, E. Paragraph-, word-, and coherence-based approaches to sentence ranking: A comparison of algorithm and human performance. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics* (Barcelona, Spain, July 2004).
33. ZOCK, M., AND BILAC, S. Word lookup on the basis of associations: from an idea to a roadmap. In *Proceedings of the Coling 2004 workshop on "Enhancing and Using Electronic Dictionaries"* (Geneva, Switzerland, August 2004).