Coling 2008

22nd International Conference on Computational Linguistics

Proceedings of the 3rd Textgraphs workshop on Graph-based Algorithms for Natural Language Processing

Workshop chairs:
Irina Matveeva, Chris Biemann, Monojit Choudhury, Mona Diab

24 August 2008
Manchester, UK

TextGraphs-3 Workshop at COLING 2008 was sponsored by Microsoft Research India

Order copies of this and other Coling proceedings from:

Association for Computational Linguistics (ACL)
209 N. Eighth Street
Stroudsburg, PA 18360
USA
Tel: +1-570-476-8006
Fax: +1-570-476-0860
acl@aclweb.org

ISBN 978-1-905593-57-6

Design by Chimney Design, Brighton, UK
Production and manufacture by One Digital, Brighton, UK

Introduction

Recent years have shown an increased interest in bringing the field of graph theory into Natural Language Processing. In many NLP applications entities can be naturally represented as nodes in a graph and relations between them can be represented as edges. Recent research has shown that graph-based representations of linguistic units as diverse as words, sentences and documents give rise to novel and efficient solutions in a variety of NLP tasks, ranging from part of speech tagging, word sense disambiguation and parsing to information extraction, semantic role assignment, summarization and sentiment analysis. The contribution of the graph representation, in addition to its intuitiveness, resides in the possibility to relate linguistic entities beyond pairwise comparison. This volume contains papers accepted for presentation at the TextGraphs-3 2008 Workshop on Graph- Based Algorithms for Natural Language Processing. This event took place on August 24, 2008, in Manchester, UK, immediately following COLING 2008, the 22nd International Conference on Computational Linguistics. It was the third workshop on this topic, building on the success of the first and second TextGraphs workshop at HLT-NAACL 2006 and 2007. The workshop aimed at bringing together researchers working on problems related to the use of graph-based algorithms for Natural Language Processing and on the theory of graph-based methods. It addressed a broad spectrum of research areas to foster exchange of ideas and to help identify principles of using the graph notions that go beyond an ad-hoc usage.

We issued calls for both regular and short, late-breaking papers. Six regular and three short papers were accepted for presentation, based on the careful reviews of our program committee. We are indebted to all program committee members for their thoughtful, high quality and elaborate reviews, especially considering our extremely tight time frame for reviewing. The papers appearing in this volume have surely benefited from their expert feedback. This year's workshop attracted papers employing graphs in a wide range of settings, so we are proud to present a very diverse program this year. N. Hathout acquires morphological structure from a lexicon employing the bipartite graph between headwords' formal semantic features. Mapping of text to a graph-based meaning representation is conducted by S. Muresan, using a recent grammar formalism. A. B. Massé et al. lay out a general theoretical framework for addressing the symbol grounding problem in digital dictionaries. A. Moschitti and F.M. Zanzotto use Kernel methods on tree pairs for recognizing textual entailment. Combining co-occurrence and phonological similarity, K. Ichioka and F. Fukumoto semantically cluster onomatopoetic words in Japanese. D. Rao et al. examine several random walk based approaches to measure word similarity. B. McGillivray et al. address cluster overlapping with correspondence analysis and apply their method to cluster English and Italian verbs and nouns. A domain-specific summarization method ranking nodes in a graph of concepts is introduced by L. Plaza Morales et al. The topology of associative concept dictionaries is modeled by H. Akama et al., who report interesting scale free properties of such networks.

Finally, having a prominent researcher as an invited speaker greatly contributes to the quality of the workshop. We thank Dragomir Radev for his talk and for the support he provided for this as well as all the previous Textgraphs workshops. We are also grateful to Microsoft Research India for sponsoring the travel and accomodation of the invited speaker.

Irina Matveeva, Chris Biemann, Monojit Choudhury and Mona Diab
August 2008

Organizers:

Irina Matveeva, Accenture Technology Labs
Chris Biemann, Powerset
Monojit Choudhury, Microsoft Research India
Mona Diab, Columbia University

Programme Committee:

Eneko Agirre, University of the Basque Country
Edo Airoldi, Princeton University
Regina Barzilay, MIT
Fernando Diaz, Yahoo! Montreal
Günes Erkan, Google
Michael Gamon, Microsoft Research Redmond
Andrew Goldberg, University of Wisconsin
Hany Hassan, IBM Egypt
Samer Hassan, University of North Texas
Gina Levow, University of Chicago
Rada Mihalcea, University of North Texas
Animesh Mukherjee, IIT Kharagpur
Dragomir Radev, University of Michigan
Uwe Quasthoff, University of Leipzig
Aitor Soroa, University of the Basque Country
Hans Friedrich Witschel, University of Leipzig
Fabio Massimo Zanzotto, University of Rome "Tor Vergata"
Thorsten Zesch, University of Darmstadt

Invited Speaker:

Dragomir Radev, University of Michigan, Ann Arbor

Table of Contents

Conference Programme

Sunday, August 24, 2008

9:20–9:25 Opening

9:30–10:30 Invited Talk by Dragomir Radev on "Lexical Affinity"

10:30–11:00 Break

Session I: Full Papers

11:00–11:30 *Acquistion of the Morphological Structure of the Lexicon Based on Lexical Similarity and Formal Analogy*
Nabil Hathout

11:30–12:00 *Learning to Map Text to Graph-Based Meaning Representations via Grammar Induction*
Smaranda Muresan

12:00–12:30 *How is Meaning Grounded in Dictionary Definitions?*
Alexandre Blondin Mass, Guillaume Chicoisne, Yassine Gargouri, Stevan Harnad, Odile Marcotte and Olivier Picard

12:30–14:00 Lunch

Session II: Full Papers

14:00–14:30 *Encoding Tree Pair-Based Graphs in Learning Algorithms: The Textual Entailment Recognition Case*
Alessandro Moschitti and Fabio Massimo Zanzotto

14:30–15:00 *Graph-Based Clustering for Semantic Classification of Onomatopoetic Words*
Kenichi Ichioka and Fumiyo Fukumoto

15:00–15:30 *Affinity Measures Based on the Graph Laplacian*
Delip Rao, David Yarowsky and Chris Callison-Burch

15:30–16:00 Break

Sunday, August 24, 2008 (continued)

Session III: Short Papers

16:00–16:20 *Semantic Structure from Correspondence Analysis*
Barbara McGillivray, Christer Johansson and Daniel Apollon

16:20–16:40 *Concept-Graph Based Biomedical Automatic Summarization Using Ontologies*
Laura Plaza, Alberto Daz and Pablo Gervs

16:40–17:00 *Random Graph Model Simulations of Semantic Networks for Associative Concept Dictionaries*
Hiroyuki Akama, Jaeyoung Jung, Terry Joyce and Maki Miyake

Acquistion of the morphological structure of the lexicon based on lexical similarity and formal analogy

Nabil Hathout
Université de Toulouse
Nabil.Hathout@univ-tlse2.fr

Abstract

The paper presents a computational model aiming at making the morphological structure of the lexicon emerge from the formal and semantic regularities of the words it contains. The model is purely lexeme-based. The proposed morphological structure consists of (1) binary relations that connect each headword with words that are morphologically related, and especially with the members of its morphological family and its derivational series, and of (2) the analogies that hold between the words. The model has been tested on the lexicon of French using the TLFi machine readable dictionary.

1 Lexeme-based morphology

Morphology is traditionally considered to be the field of linguistics that studies the structure of words. In this conception, words are made of morphemes which combine according to rules of inflexion, derivation and composition. If the morpheme-based theoretical framework is both elegant and easy to implement, it suffers many drawbacks pointed out by several authors (Anderson, 1992; Aronoff, 1994). The alternative theoretical models that have been proposed falls within lexeme-based or word-based morphology in which the minimal units are words instead of morphemes. Words then do not have any structure at all and morphology becomes a level of organization of the lexicon based on the sharing of semantic and formal properties.

The morpheme-based / lexeme-based distinction shows up on the computational level. In the morpheme-based conception, the morphological analysis of a word aims at segmenting it into a sequence of morphemes (Déjean, 1998; Goldsmith, 2001; Creutz and Lagus, 2002; Bernhard, 2006). In a lexeme-based approach, it is to discover the relations between the word and the other lexical items. These relations serve to identify the morphological family of the word, its derivational series, and the analogies in which it is involved. For instance, the analysis of the French word *dérivation* may be considered as satisfactory if it connects *dérivation* with enough members of its family (*dériver* 'derivate', *dérivationnel* 'derivational', *dérivable*, *dérive* 'drift', *dériveur* 'sailing dinghy', etc.) and of its derivational series (*formation* 'education', *séduction*, *variation*, *émission*, etc.). Each of these relations is integrated into a large collection of analogies that characterizes it semantically and formally. For instance, the relation between *dérivation* and *dérivable* is part of a series of analogies which includes *dérivation*:*dérivable*::*variation*:*variable*, *dérivation*:*dérivable*::*modification*:*modifiable*, etc. Similarly, *dérivation* and *variation* participates in a series of analogies such as *dérivation*:*variation*::*dériver*:*varier*, *dérivation*:*variation*::*dérivationnel*:*variationnel*, *dérivation*:*variation*::*dérivable*:*variable*.

2 Computational modeling

The paper describes a computational model aiming at making the morphological derivational structure of the lexicon emerge from the semantic and the formal regularities of the words it contains. A first experiment is currently underway on the lexicon of French using the TLFi machine readable dictio-

Coling 2008: Proceedings of 3rd Textgraphs workshop on Graph-Based Algorithms in Natural Language Processing, pages 1–8
Manchester, August 2008

nary.[1] The main novelty of the paper is the combination of lexical proximity with formal analogy. We first use lexical similarity in order to select a set of words that are likely to be morphologically related to each other. Then, these candidates are checked by means of analogy.

The two techniques are complementary. The first one brings closer the words that are morphologically close and especially the ones that are members of the same morphological families and the same derivational series. It is able to deal with large number of words, but it is too coarse-grained to discriminate the words that are actually morphological related from the ones that are not. The second technique, formal analogy, is then used to perform a fine-grained filtering. Technically, our model joins:

1. the representation of the lexicon as a graph and its exploration through random walks, along the line of (Gaume et al., 2002; Gaume et al., 2005; Muller et al., 2006), and

2. formal analogies on words (Lepage, 1998; Stroppa and Yvon, 2005). This approach does do not make use of morphemes. Correspondence between words is calculated directly on their graphemic representations.

More generally, our approach is original in that:

1. Our computational model is pure lexeme-based. The discovery of morphological relations between words do not involve the notions of morpheme, affix, morphological exponent, etc. nor any representation of these concepts.

2. The membership to the families and series is gradient. It accounts, for instance, for the fact that *dériveur* is morphologically and semantically closer to *dérive* than to *dérivationnellement*, even if the three words belong to the same family. The model connects the words that share semantic and / or formal features. The more features are shared, the closer the words are.

Besides, the model integrates semantic and formal informations in a uniform manner. All kind of semantic informations (lexicographic definitions, synonyms, synsets, etc.) and formal ones (graphemic, phonological, etc.) can be used. They can be cumulated easily in spite of the differences in nature and origin. The model takes advantage of the redundancy of the features and is fairly insensitive to variation and exceptions.

[1] *Trésor de la Langue Française* (http://atilf.atilf.fr/).

3 Related work

Many works in the field of computational morphology aim at the discovery of relations between lexical units. All of them rely primarily on finding similarities between the word graphemic forms. These relations are mainly prefixal or suffixal with two exceptions, (Yarowsky and Wicentowski, 2000) and (Baroni et al., 2002), who use string edit distances to estimate formal similarity. As far as we know, all the other perform some sort of segmentation even when the goal is not to find morphemes as in (Neuvel and Fulop, 2002). Our model differs from these approaches in that the graphemic similarities are determined solely on the basis of the sharing of graphemic features. It is the main contribution of this paper.

Our model is also related to approaches that combine graphemic and semantic cues in order to identify morphemes or morphological relations between words. Usually, these semantic informations are automatically acquired from corpora by means of various techniques as latent semantic analysis (Schone and Jurafsky, 2000), mutual information (Baroni et al., 2002) or co-occurrence in an n-word window (Xu and Croft, 1998; Zweigenbaum and Grabar, 2003). In the experiment we present here, semantic informations are extracted from a machine readable dictionary and semantic similarity is calculated through random walks in a lexical graph. Our approach can also be compared with (Hathout, 2002) where morphological knowledge is acquired by using semantic informations extracted from dictionaries of synonyms or from WordNet.

4 Lexeme Description

In our model, the lexical units and their properties are represented in a bipartite graph with the vertices representing the lexemes in one sub-set and the vertices representing the formal and semantic features in the other. Lexeme vertices are identified by the lemma and the grammatical category.

In the experiment reported in the paper, the formal properties are the n-grams of letters that occur in the lexemes lemma. Figure 1 shows a sub-set of

```
$or; $ori; $orie; ...
$orientation; ori; orie; ...
orientation; orientation$; ...
tio; tion; tion$; ion; ion$; on$
```

Figure 1: Excerpt of the formal features associated with the noun *orientation*.

```
N.action; N.action X.de; N.action
X.de V.orienter; X.de; X.de
V.orienter; V.orienter; X.de
V.s'orienter; V.s'orienter;
N.résultat; N.résultat X.de;
N.résultat X.de X.ce; N.résultat
X.de X.ce N.action; X.de X.ce;
X.de X.ce N.action; X.ce; X.ce
N.action; N.action
```

Figure 2: Semantic features induced by the definition "Action d'orienter, de s'orienter ; résultat de cette action." of the noun *orientation*

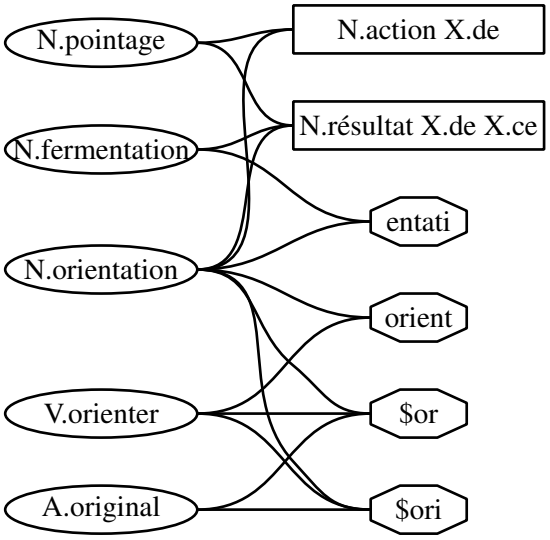


Figure 3: Excerpt of the bipartite graph which represents the lexicon. Words are displayed in ovals, semantic feature in rectangles and formal features in octagons. The graph is symmetric.

the formal features associated with the word *orientation*. The beginning and the end of the lemma are marked by the character `$`. We impose a minimum size on the n-grams ($n \geq 3$).

The model is pure lexeme-based because this decomposition does not confer a special status to any of the individual n-grams which characterize the lexemes. All n-grams play the same role and therefore no one has the status of morpheme. These features are only used to bring closer the words that share the same sounds.

The semantic properties we have used are extracted from the TLFi definitions. Each headword is provided with the n-grams of words that occur in its definitions. The n-grams that contain punctuation marks are eliminated. In other words, we only use n-grams of words that occur between two punctuation marks. For instance, the semantic features induced by the definition *Action d'orienter, de s'orienter ; résultat de cette action.* ('act of orienting, of finding one's way; result of this action') of the noun *orientation* are presented in figure 2. The words in the definitions are POS tagged and lemmatized. The tags are A for adjectives, N for nouns, R for adverbs, V for verbs and X for all other categories.

This is a very coarse semantic representation inspired from the repeated segments (Lebart et al., 1998). It offers three advantages: (1) being heavily redundant, it can capture various levels of similarity between the definitions; (2) it integrates informations of a syntagmatic nature without a deep syntactic analysis of the definitions; (3) it slightly reduces the strong variations in the lexicographical treatment of the headwords, especially in the division into sub-senses and in the definitions.

The bipartite graph is built up by symmetrically connecting each headword to its semantic and formal features. For instance, the noun *orientation* is connected with the formal feature `$or`, `$ori`, `$orie`, `$orien`, etc. which are in turn connected with the words *orienter*, *orientable*, *orientement* 'orientation', *orienteur* 'orientor', etc. Likewise, *orientation* is connected with the semantic features `N.action X.de`, `N.résultat X.de X.ce N.action`, etc. which are themselves connected with the nouns *orientement*, *harmonisation* 'synchronization', *pointage* 'checking', etc. The general schema is illustrated in figure 4. This representation corresponds precisely to the Network Model of Bybee (1995).

We use a bipartite graph mainly for two reasons: (1) We can spread an activation synchronously into the formal and the semantic sub-graphs. (2) It contains representations of the formal and the semantic properties of the lexemes which, for instance, could be used in order to describe the semantics of the *-able* suffixation or the characteristic endings of the boat names (*-ier*, *-eur*, etc.). However, the bipartite structure is not essential and we only need

to be able to compute morphological distances between words.

5 Random walks

The computational side of the method is based on the estimation of the proximity between words represented in a lexical graph (Gaume et al., 2002; Gaume et al., 2005; Muller et al., 2006). The graphs used in this approach are slightly different from the ones presented above. All their vertices represent words and the edges describe semantic relations such as synonymy. The proximity is computed by simulating the spreading into the graph of an activation initiated at a vertice. Following the spreading, the nodes which are most excited are regarded as being the closest to the initial vertice.

The same method can be used to estimate the morphological proximity between words that are described in a bipartite graph like the one we propose (see figure 4). It then connects words that have the same semantic and formal features. One has just to propagate the activation into the bipartite graph for an even number of times. When the graph is heavily redundant, two steps of propagation are sufficient to obtain the intended proximity estimations.

In the example in figure 4, the morphological neighbors of the noun *orientation* are identified by activating the vertice which represents it. In the first step, the activation is spread toward the vertices which represent its formal and semantic features. In the second step, the activation located on the feature vertices is spread toward the headword vertices. For instance, *orienter* becomes activated via the formal features `$or`, `$ori`, `orien` and *fermentation* through the formal feature `entati` and the semantic feature `N.résultat X.de X.ce`. The greater the number of features shared by a headword with *orientation*, the stronger the activation it receives.

The spreading of activation is simulated as a random walk in the lexical graph, classically computed as a multiplication of the stochastic adjacency matrix. More precisely, let $G = (V, E, w)$ be a weighted graph consisting of a set of vertices $V = \{v_1, \ldots, v_n\}$, a set of edges $E \subset V^2$ and of a weight function $w : E \rightarrow \mathbb{R}$. Let A be the adjacency matrix of G, that is a $n \times n$ matrix such that $A_{ij} = 0$ if $(v_i, v_j) \notin E$ and $A_{ij} = w(v_i, v_j)$ if $(v_i, v_j) \in E$. (In the experiment, $w(e) = 1, \forall e \in E$.) We normalize the rows of A in order to get a stochastic matrix M. M^n_{ij} is the probability of reaching node v_j from the node v_i through a walk of n steps. This probability can also be regarded as an activation level of node v_j following an n-step spreading initiated at vertice v_i.

In the experiment presented in this paper, the activation is spread for one half toward the semantic feature and for the other toward the formal features. The edges of the bipartite graph can be divided in three parts $E = J \cup K \cup L$ where J contains the edges that connect a headword to a formal feature, K the edges that connect a headword to a semantic feature and L the edges that connect a formal or semantic feature to a headword. The values of M are defined as follows:

- if $e_{ij} = (v_i, v_j) \in J$, $M_{ij} = \frac{A_{ij}}{2 \sum_{e_{ih} \in J} A_{ih}}$ if v_i is connected to a semantic feature and $M_{ij} = \frac{A_{ij}}{\sum_{e_{ik} \in J} A_{ik}}$ otherwise.
- if $e_{ik} = (v_i, v_k) \in K$, $M_{ik} = \frac{A_{ik}}{2 \sum_{e_{ih} \in K} A_{ih}}$ if v_i is connected to a formal feature and $M_{ik} = \frac{A_{ik}}{\sum_{e_{ih} \in K} A_{ih}}$ otherwise.
- if $e_{il} = (v_i, v_l) \in L$, $M_{il} = \frac{A_{il}}{\sum_{e_{ih} \in L} A_{ih}}$.

6 Lexical neighborhood

The graph used in the experiment has been built from the definitions of the TLFi. We only removed the definitions of non standard uses (old, slang, etc.). The extraction and cleaning-up of the definitions have been carried out in collaboration with Bruno Gaume and Philippe Muller. The bipartite graph has been created from 225 529 definitions describing 75 024 headwords (lexemes). We then removed all the features associated only with one headword. This reduces the size of the graph significantly without changing the connections that hold between the headwords. Table 1 shows that this reduction is stronger for the semantic feature (93%) than it is for the formal ones (69%). Indeed, semantic descriptions show greater variability than formal ones.

The use of the graph is illustrated in figure 4. It shows the 20 nearest neighbors of the verb *fructifier* for various propagation configurations. The examples in (a) and (b) show clearly that formal features are the more predictive ones while semantic features are the less reliable ones. The example in (c) illustrates the contribution of the semantic

(a) V.fructifier N.fructification A.fructificateur A.fructifiant A.fructifère V.sanctifier V.rectifier *A.rectifier V.fructidoriser N.fructidorien N.fructidor* N.fructuosité R.fructueusement A.fructueux *N.rectifieur A.obstructif A.instructif A.destructif A.constructif* N.infructuosité

(b) V.fructifier *V.trouver N.missionnaire N.mission A.missionnaire N.saisie N.police N.hangar N.dîme N.ban V.affruiter N.melon N.saisonnement N.azédarach A.fruitier A.bifère V.saisonner N.roman N.troubadour V.contaminer*

(c) V.fructifier A.fructifiant N.fructification A.fructificateur *V.trouver* A.fructifère V.rectifier V.sanctifier *A.rectifier V.fructidoriser N.fructidor N.fructidorien N.missionnaire N.mission A.missionnaire* A.fructueux R.fructueusement N.fructuosité *N.rectifieur N.saisie*

Figure 4: The 20 nearest neighbors of the verb *fructifier* when the activation is spread (a) only toward the formal features, (b) only toward the semantic ones, (c) toward both the semantic and formal features. Words that do not belong to the family or series of *fructifier* are emphasized.

graph	complete	reduced
formal features	1 306 497	400 915
semantic features	7 650 490	548 641

Table 1: Number of the semantic and formal features coming from TLFi.

features. They reorder the formal neighbors and introduce among them the nearest semantic neighbors. We see in the lists in (a) and (c) that the family members are the nearest neighbors and that the members of the series come next.

7 Analogy

The members of the series and families are massively involved in the analogies which structure the lexicon. A word x belonging to a family F_x participates in several analogies with a large number of other members of F_x. The analogies that involve two words $(x, y) \in F^2$ include two other words (z, t) that belong to one same family F'. On the other hand, if x is a complex word that belongs to a series S_x, then $z \in S_x$, $x \in S_z$, $y \in S_t$ and $t \in S_y$. For instance, the couple of words *fructifier* and *fructification* form analogies with of members of other families (*rectifier*, *rectification*), (*certifier*, *certification*), (*plastifier*, *plastification*), etc. Moreover, the first elements of these couples belong to series of *fructifier* and the second ones to the series of *fructification*.

In a dual manner, a word u belonging to a series S participates in a set of analogies with a large number of other members of S. The analogies that involve two elements of the same series are made up with words which themselves belong to a same series. For instance, *fructifier* and *sanctifier* form analogies with the members of other series (*fructificateur*, *sanctificateur*), (*fructification*, *sanctification*) or (*fructifiant*, *sanctifiant*). These couples are respectively made of members of the families of *fructifier* and *sanctifier*.

7.1 Analogies and neighborhoods

The analogies that involve members of families and series can be used to efficiently filter the morphological neighbors that are identified by the method presented above. If v is a correct morphological neighbor of w, then it is either a member of the family of m or a member of its series. Therefore, it exists another neighbor v' of w (v' belong to the family of w if v belongs to the series of w or vice versa) such that it exists a neighbor w' of v and of v' such that $w : v :: v' : w'$.[2] Therefore, we have two configurations:

1. if $v \in F_w$, then $\exists v' \in S_w, \exists w' \in S_v \cap F_{v'}, w : v :: v' : w'$
2. if $v \in S_w$, then $\exists v' \in F_w, \exists w' \in F_v \cap S_{v'}, w : v :: v' : w'$

The first case is illustrated by the above examples with $w =$ *fructifier* and $v =$ *fructification*, and the second one with $w =$ *fructifier* et $v =$ *rectifier*.

7.2 Formal analogy

A formal or graphemic analogy is a relation $a : b :: c : d$ that holds between four strings such that the graphemic differences between a

[2]The notation $a : b :: c : d$ is used as a shorthand for the statement that (a, b, c, d) forms an analogical quadruplet, or in other words that a is to b as c is to d.

and b are the same as the ones between c and d. It can be exemplified with the four Arabic words `kataba`:`maktoubon`::`fa3ala`:`maf3oulon` which respectively are transcriptions of the verb 'write', the noun 'document', the verb 'do' and the noun 'effect.'[3] The differences between the first two words and between the two last ones can be described as in figure 5. They are identical for the two couples of words.

ϵ	k	a	t	a	b	a
ma	k	ϵ	t	ou	b	on

ϵ	f	a	3	a	l	a
ma	f	ϵ	3	ou	l	on

Figure 5: Formal analogy `kataba`: `maktoubon`::`fa3ala`:`maf3oulon`. The differences are locates in frame boxes.

More generally, formal analogies can be defined in terms of factorization (Stroppa and Yvon, 2005). Let L be an alphabet and $a \in L^\star$ a string over L. A factorization of a is a sequence $f = (f_1, \cdots, f_n) \in L^{\star n}$ such that $a = f_1 \oplus \cdots \oplus f_n$ where $\oplus$ denotes the concatenation. For instance, (`ma`, `k`, ϵ, `t`, `ou`, `b`, `on`) is a factorization of length 7 of `maktoubon`. Morphological analogies can be defined as follows. Let $(a, b, c, d) \in L^{\star 4}$ be for strings. $a : b :: c : d$ is a formal analogy iff there exists $n \in \mathbb{N}$ and four factorizations of length n of the four strings $(f(a), f(b), f(c), f(d)) \in L^{\star 4}$ such that, $\forall i \in [1, n], (f_i(b), f_i(c)) \in \{(f_i(a), f_i(d)), (f_i(d), f_i(a))\}$. For the analogy `kataba`:`maktoubon`::`fa3ala`:`maf3oulon`, the property holds for $n = 7$ (see figure 5).

7.3 Implementation

A formal analogy $a : b :: c : d$ can be easily checked by comparing the sequences of string edit operations between (a, b) and between (c, d). Both sequences must minimize Levenshtein edit distance (i.e. have a minimal cost). Each sequence corresponds to a path in the edit lattices of the couple of words. The lattice are represented by a matrix computed using the standard string edit algorithm (Jurafsky and Martin, 2000). The path which describes the sequence of string edit operations starts at the last cell of the matrix and climbs to the first one. Only three directions are allowed: upward (deletion), to the left (insertion) or in the upper left diagonal direction (substitution). Figure 6 shows the sequence of edit operations for the couple `fructueux`:`infructueusement`. Sequences of edit operations can be simplified by merging the series of identical character matchings. The sequence in figure 6 then becomes ((I,ϵ,`i`), (I,ϵ,`n`), (M,`fructueu`,`fructueu`), (S,`x`,`s`), (I,ϵ,`e`), (I,ϵ,`m`), (I,ϵ,`e`), (I,ϵ,`n`), (I,ϵ,`t`)). This simplified sequence is identical to the one for the couple `soucieux`:`insoucieusement` except for the matching operation: ((I,ϵ,`i`), (I,ϵ,`n`), (M,`soucieu`,`soucieu`), (S,`x`,`s`), (I,ϵ,`e`), (I,ϵ,`m`), (I,ϵ,`e`), (I,ϵ,`n`), (I,ϵ,`t`)). The two sequences can be made identical if the matching sub-strings are not specified. The resulting sequence can then be assigned to both couples as their edit signatures (σ). The formal analogy `fructueux`:`infructueusement`:: `soucieux`:`insoucieusement` can be stated in terms of identity the edit signatures:

$\sigma($`fructueux`, `infructueusement`$) =$
$\sigma($`soucieux`, `insoucieusement`$) =$
((I,ϵ,`i`), (I,ϵ,`n`), (M,@,@), (S,`x`,`s`), (I,ϵ,`e`), (I,ϵ,`m`), (I,ϵ,`e`), (I,ϵ,`n`), (I,ϵ,`t`))

More generally, four strings $(a, b, c, d) \in L^{\star 4}$ form a formal analogy $a : b :: c : d$ iff $\sigma(a, b) = \sigma(c, d)$ or $\sigma(a, c) = \sigma(b, d)$.

7.4 First results

The computational model we have just presented has been implemented and a first experiment has been carried out. It consists in determining the 100 closest neighbors of every headword for the three configurations presented in § 6. All the formal analogies that hold between these words have then been collected. We have not been able to do a standard evaluation in terms of recall and precision because of the lack of morphological resources for French. However, we have manually checked the analogies of 22 headwords belonging to 4 morphological families. An analogy $a : b :: c : d$ is accepted as correct if:

- b belongs to the family of a, c belongs to the series of a, d belongs to series of b and to the family of c, or
- b belongs to the series of a, c belongs to the family of a, d belongs to family of b and to the series of c.

[3] This example is adapted from examples in (Lepage, 1998; Lepage, 2003).

I	I	M	M	M	M	M	M	M	M	S	I	I	I	I	I
ϵ	ϵ	f	r	u	c	t	u	e	u	x	ϵ	ϵ	ϵ	ϵ	ϵ
i	n	f	r	u	c	t	u	e	u	s	e	m	e	n	t

Figure 6: Sequence of edit operations that transform `fructueux` into `infructueusement`. The type of each operation is indicated on the first line: D for deletion, I for insertion, M for matching and S for a substitution by a different character.

configuration	analogies	correct	errors
formal	169	163	3.6%
semantics	5	5	0.0%
sem + form	130	128	1.5%

Table 2: Number of the analogies collected for a sample of 22 headwords and error rate.

The results are summarized in table 2. Their quality is quite satisfactory. However, the number of analogies strongly depends on the configuration of propagation. The best trade-off is a simultaneous propagation toward the semantic and formal features. Here are some of the correct and erroneous analogies collected:

- R.fructueusement:R.affectueusement:: A.infructueux:A.inaffectueux
- N.fructification:N.identification:: V.fructifier:V.identifier
- N.fruiterie:N.fruitier::N.laiterie:N.laitier
- * N.fruit:N.bruit::V.frusquer:V.brusquer

The first example is particularly interesting because it involves on one side suffixed words and on the other prefixed ones.

The performance of the method strongly depends on the length of the headwords. Table 3 presents the number of analogies and the error rate for 13 groups of 5 words. The words of each group are of the same length. Lengths range from 4 to 16 letters.

length	analogies	correct	errors
4	29	15	51.7%
5	22	8	36.4%
6	8	1	12.5%
7	10	2	20.0%
8	55	1	1.8%
9	29	2	6.9%
10	30	0	0.0%
11	32	0	0.0%
12	19	0	0.0%
13	11	0	0.0%
14	35	0	0.0%
15	63	0	0.0%
16	39	0	0.0%

Table 3: Number of the analogies and error rate for headwords of length 4 to 16.

8 Conclusion

We have presented a computational model that makes the morphological structure of the lexicon emerge from the formal and semantic regularities of the words it contains. The model is radically lexeme-based. It integrates the semantic and formal properties of the words in a uniform manner and represents them into a bipartite graph. Random walks are used to simulate the spreading of activations in this lexical network. The level of activation obtained after the propagation indicates the lexical relatedness of the words. The members of the morphological family and the derivational series of each word are then identified among its lexical neighbors by means of formal analogies.

This is work in progress and we still have to separate the members of the families from the members of the series. We also intend to conduct a similar experiment on the English lexicon and to evaluate our results in a more classical manner by using the CELEX database (Baayen et al., 1995) as gold standard. The evaluation should also be done with respect to well known systems like *Linguistica* (Goldsmith, 2001) or the morphological analyzer of Bernhard (2006).

Acknowledgments

I would like to thank the ATILF laboratory and Jean-Marie Pierrel for making available to me the TLFi. I am in debt to Bruno Gaume and Philippe Muller for the many discussions and exchanges we have had on the cleaning-up of the TFLi and its exploitation through random walks. I am also grateful to Gilles Boyé, Olivier Haute-Cœur and Lu-

dovic Tanguy for their comments and suggestions. All errors are mine.

References

Anderson, Stephen R. 1992. *A-Morphous Morphology*. Cambridge University Press, Cambridge, UK.

Aronoff, Mark. 1994. *Morphology by Itself. Stem and Inflexional Classes*. MIT Press, Cambridge, Mass.

Baayen, R. Harald, Richard Piepenbrock, and Leon Gulikers. 1995. The CELEX lexical database (release 2). CD-ROM. Linguistic Data Consortium, University of Pennsylvania, Pennsylvania, USA.

Baroni, Marco, Johannes Matiasek, and Harald Trost. 2002. Unsupervised discovery of morphologically related words based on orthographic and semantic similarity. In *Proceedings of the Workshop on Morphological and Phonological Learning of ACL-2002*, pages 48–57, Philadelphia. ACL.

Bernhard, Delphine. 2006. Automatic acquisition of semantic relationships from morphological relatedness. In *Advances in Natural Language Processing, Proceedings of the 5th International Conference on NLP, FinTAL 2006*, volume 4139 of *Lecture Notes in Computer Science*, pages 121–13. Springer.

Bybee, Joan L. 1995. Regular morphology and the lexicon. *Language and cognitive processes*, 10(5):425–455.

Creutz, Mathias and Krista Lagus. 2002. Unsupervised discovery of morphemes. In *Proceedings of the ACL Workshop on Morphological and Phonological Learning*, pages 21–30, Philadelphia, Penn. ACL.

Déjean, Hervé. 1998. Morphemes as necessary concept for structures discovery from untagged corpora. In *Proceedings of the Workshop on Paradigms and Grounding in Natural Language Learning*, pages 295–299, Adelaide, Australia.

Gaume, Bruno, Karine Duvigneau, Olivier Gasquet, and Marie-Dominique Gineste. 2002. Forms of meaning, meaning of forms. *Journal of Experimental and Theoretical Artificial Intelligence*, 14(1):61–74.

Gaume, B., F. Venant, and B. Victorri. 2005. Hierarchy in lexical organization of natural language. In Pumain, D., editor, *Hierarchy in natural and social sciences*, Methodos series, pages 121–143. Kluwer.

Goldsmith, John. 2001. Unsupervised learning of the morphology of natural language. *Computational Linguistics*, 27(2):153–198.

Hathout, Nabil. 2002. From wordnet to celex: acquiring morphological links from dictionaries of synonyms. In *Proceedings of the Third International Conference on Language Resources and Evaluation*, pages 1478–1484, Las Palmas de Gran Canaria. ELRA.

Jurafsky, Daniel and James H. Martin. 2000. *Speech and language processing*. Prentice-Hall.

Lebart, Ludovic, André Salem, and Lisette Berry. 1998. *Exploring textual data*. Kluwer Academic Publishers, Dordrecht.

Lepage, Yves. 1998. Solving analogies on words: an algorithm. In *Proceedings of COLING-ACL'98*, volume 2, pages 728–735, Montréal, Canada.

Lepage, Yves. 2003. *De l'analogie rendant compte de la commutation en linguistique*. Mémoire de HDR, Université Joseph Fourier, Grenoble.

Muller, Philippe, Nabil Hathout, and Bruno Gaume. 2006. Synonym extraction using a semantic distance on a dictionary. In Radev, Dragomir and Rada Mihalcea, editors, *Proceedings of the HLT/NAACL workshop Textgraphs*, pages 65–72, New York, NY. Association for Computational Linguistics.

Neuvel, Sylvain and Sean A. Fulop. 2002. Unsupervised learning of morphology without morphemes. In *Proceedings of the Workshop on Morphological and Phonological Learning 2002*, Philadelphia. ACL Publications.

Schone, Patrick and Daniel S. Jurafsky. 2000. Knowledge-free induction of morphology using latent semantic analysis. In *Proceedings of the Conference on Natural Language Learning 2000 (CoNLL-2000)*, pages 67–72, Lisbon, Portugal.

Stroppa, Nicolas and François Yvon. 2005. An analogical learner for morphological analysis. In *Proceedings of the 9th Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 120–127, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Xu, Jinxi and W. Bruce Croft. 1998. Corpus-based stemming using co-occurrence of word variants. *ACM Transaction on Information Systems*, 16(1):61–81.

Yarowsky, David and Richard Wicentowski. 2000. Minimally supervised morphological analysis by multimodal alignment. In *Proceedings of the Association of Computational Linguistics (ACL-2000)*, pages 207–216, Hong Kong.

Zweigenbaum, Pierre and Natalia Grabar. 2003. Learning derived words from medical corpora. In *9th Conference on Artificial Intelligence in Medicine Europe*, pages 189–198, Cyprus.

Learning to Map Text to Graph-based Meaning Representations via Grammar Induction

Smaranda Muresan
Laboratory for Computational Linguistics and Information Processing
Institute for Advanced Computer Studies
University of Maryland
College Park, MD 20742, USA
smara@umiacs.umd.edu

Abstract

We argue in favor of using a graph-based representation for language meaning and propose a novel learning method to map natural language text to its graph-based meaning representation. We present a grammar formalism, which combines syntax and semantics, and has ontology constraints at the rule level. These constraints establish links between language expressions and the entities they refer to in the real world. We present a relational learning algorithm that learns these grammars from a small representative set of annotated examples, and show how this grammar induction framework and the ontology-based semantic representation allow us to directly map text to graph-based meaning representations.

1 Introduction

Recent work (Wong and Mooney, 2007; Zettlemoyer and Collins, 2005; He and Young, 2006) has developed learning algorithms for the problem of mapping sentences to their underlying semantic representations. These semantic representations vary from λ-expressions (Bos et al., 2004; Zettlemoyer and Collins, 2005; Wong and Mooney, 2007) to DB query languages and command-like languages (RoboCup Coach Language, CLang) (Ge and Mooney, 2005).

In this paper we focus on an ontology-based semantic representation which allows us to encode the meaning of a text as a direct acyclic graph. Recently, there is a growing interest on ontology-based NLP, starting from efforts in defining ontology-based semantic representations (Nirenburg and Raskin, 2004), to using ontological resources in NLP applications, such as question answering (Basili et al., 2004; Beale et al., 2004), and building annotated corpora, such as the OntoNotes project (Hovy et al., 2006).

There are three novel properties to ontology-based semantics that we propose in this paper:

- There is a *direct link* between the ontology and the grammar through constraints at the grammar rule level. These ontology constraints enable access to meaning during language processing (parsing and generation).
- Our ontology-based semantic representation is expressive enough to capture various phenomena of natural language, yet restrictive enough to facilitate *grammar learning*. The representation encodes both ontological meaning (concepts and relations among concepts) and extra-ontological meaning, such as voice, tense, aspect, modality.
- Our representation and grammar learning framework allow a direct mapping of text to its meaning, encoded as a direct acyclic graph (DAG). We consider that "understanding" a text is the ability to correctly answer, at the conceptual level, all the questions asked w.r.t to that text, and thus *Meaning = Text + all Questions/Answers* w.r.t that *Text*. Under this assumption, obtaining the meaning of a text is reduced to a question answering process, which in our framework is a *DAG matching problem*.

First, we review our grammar formalism introduced in (Muresan, 2006; Muresan and Rambow, 2007), called *Lexicalized Well-Founded Grammars*. Second, we present a relational learning algorithm for inducing these grammars from a representative sample of strings annotated with their semantics, along with minimal assumptions about

Coling 2008: Proceedings of 3rd Textgraphs workshop on Graph-Based Algorithms in Natural Language Processing, pages 9–16
Manchester, August 2008

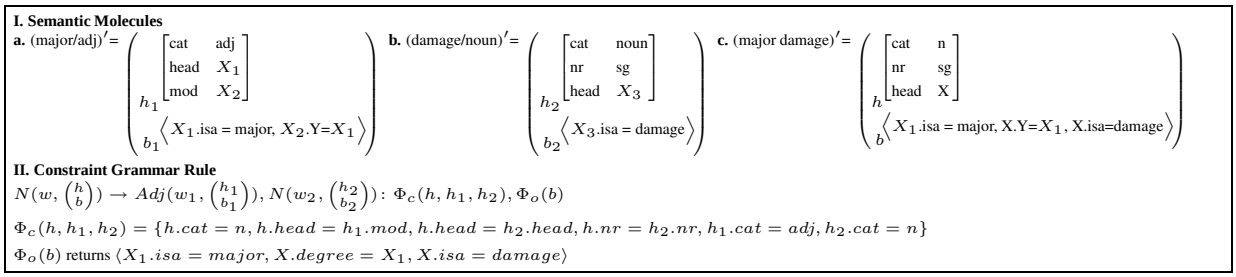


Figure 1: Examples of three semantic molecules (I), and a constraint grammar rule together with the semantic composition and ontology-based interpretation constraints, Φ_c and Φ_o (II)

syntax. Then, we describe the levels of representation we use to go from utterances to their graph-based meaning representations, and show how our representation is suitable to define the meaning of an utterance/text through answers to questions. As a proof of concept we discuss how our framework can be used to acquire terminological knowledge from natural language definitions and to query this knowledge using wh-questions.

2 Grammar Formalism

Lexicalized Well-Founded Grammars (LWFGs) introduced in (Muresan, 2006; Muresan and Rambow, 2007) are a type of Definite Clause Grammars (Pereira and Warren, 1980) where: (1) the context-free backbone is extended by introducing a partial ordering relation among nonterminals (the basis for "well-founded"); (2) each string is associated with a syntactic-semantic representation called a *semantic molecule*; and (3) grammar rules have two types of constraints: one for semantic composition and one for ontology-based semantic interpretation. The last two properties allow us to have a syntactic-semantic grammar. The ontology constraints provide access to meaning during language learning, parsing and generation. The first property allows us to learn these grammars from a small set of annotated examples.

The *semantic molecule* is a syntactic-semantic representation of natural language strings $w' = \binom{h}{b}$, where h *(head)* encodes the information required for semantic composition, and b *(body)* is the actual semantic representation of the string. Figure 1 gives examples of semantic molecules for an adjective, a noun and a noun phrase, as presented in (Muresan and Rambow, 2007).

The head h of the semantic molecule is a flat feature structure (i.e., feature values are atomic), having at least two attributes that encode the syntactic category of the associated string, cat, and the head of the string, head. In addition, attributes for agreement and other grammatical features can be present (e.g., nr, pers for number and person). The set of attributes is finite and known a-priori for each syntactic category. Being a one-level feature structure, no recursive or embedded structures are allowed (unlike other grammar formalisms such as HPSG, LFG), which makes this representation appealing for a learning framework. Recursion in the grammar is obtained through the recursive grammar rules and the composition constraint.

The body, b, of a semantic molecule is a flat representation, called OntoSeR (Ontology-based Semantic Representation). No embedding of predicates is allowed, as in Minimal Recursion Semantics (MRS) (Copestake et al., 1999). Unlike MRS, OntoSeR is a logical form built as a conjunction of *atomic predicates* $\langle concept\rangle.\langle attr\rangle = \langle concept\rangle$, where variables are either concept or slot (attr) identifiers in an ontology. For example, the adjective *major* is represented as $\langle X_1.isa = major, X_2.Y = X_1\rangle$, which says that the meaning of an adjective is a concept X_1 ($X_1.isa = major$) that is the value of a property of another concept X_2 ($X_2.Y = X_1$) in the ontology.

A LWFG specifies one or more semantic molecules for each string that can be parsed by the grammar. The lexicon of a LWFG consists of words paired with their semantic molecules shown in Figure 1(Ia and Ib). In addition to the lexicon, a LWFG has a set of constraint grammar rules. An example of a LWFG rule is given in Figure 1(II). Grammar nonterminals are augmented with pairs of strings and their semantic molecules. These pairs are called *syntagmas*, and are denoted by $\sigma = (w, w') = (w, \binom{h}{b})$. This rule generates the syntagma corresponding to *major damage* whose semantic molecule is given in Figure 1(Ic). There are two types of constraints at the grammar rule level — one for *semantic composition* (how the

meaning of a natural language expression is composed from the meaning of its parts) and one for *ontology-based semantic interpretation*. The composition constraints Φ_c are applied to the heads of the semantic molecules, the bodies being just concatenated. Figure 1 shows that the body of the semantic molecule for *major damage* is a concatenation of the bodies of the adjective *major* and noun *damage*, together with a variable substitution. This variable substitution $\{X_2/X, X_3/X\}$ is a result of Φ_c, which is a system of equations — a simplified version of "path equations" (Shieber et al., 1983), because the heads are flat feature structures. These constraints are learned together with the grammar rules. The ontology-based constraints Φ_o represent the validation on the ontology, and are applied to the body of the semantic molecule associated with the left-hand side nonterminal. The ontology-based interpretation is not done during the composition operation, but afterwords. Thus, for example, the head of the noun phrase *major damage* does not need to store the slot Y, a fact that allows us to use flat feature structures to represent the head of the semantic molecules. The ontology-based constraints are not learned; rather, Φ_o is a general predicate applied to the logical form semantic representation which fully contains all the required information needed for validation on the ontology. Thus, it is independent of grammatical categories. This predicate can succeed or fail as a result of querying the ontology — when it succeeds, it instantiates the variables of the semantic representation with concepts/slots in the ontology ($Y = degree$). For example, given the phrase *major damage*, Φ_o succeeds and returns $\langle X_1.isa = major, X.degree = X_1, X.isa = damage\rangle$, while given *major birth* it fails.

3 Grammar Learning Algorithm

Unlike stochastic grammar learning for syntactic parsing (e.g., (Collins, 1999)), LWFG is well suited to learning from reduced-size training data. Furthermore, unlike previous formalisms used for deeper representations (e.g, HPSG, LFG), our LWFG formalism is characterized by a formal guarantee of polynomial learnability (Muresan, 2006).

A key to these properties is the partial ordering among grammar nonterminals, i.e., the set of nonterminals is well-founded. This partial ordering among nonterminals allows us to define the *representative examples* of a LWFG, and to learn LWFGs from this small set of examples. The representative examples E_R of a LWFG, G, are the simplest syntagmas ground-derived by the grammar G — i.e., for each grammar rule, there exists a syntagma which is ground-derived from it in the minimum number of steps. Informally, representative examples are building blocks from which larger structures can be inferred via reference to a larger corpus E_σ which can be only weakly annotated (i.e., bracketed), or unannotated. This larger corpus, E_σ, is used for generalization during learning (Figure 2).

The theoretical learning model is *Grammar Approximation by Representative Sublanguage (GARS)* introduced in (Muresan, 2006; Muresan and Rambow, 2007). We proved that the search space for grammar induction is a complete grammar lattice, and we gave a learnability theorem for LWFG induction. The GARS model uses a polynomial algorithm for LWFG learning that takes advantage of the building blocks nature of representative examples. The learning algorithm belongs to the class of Inductive Logic Programming methods (ILP), based on entailment (Muggleton, 1995; Dzeroski, 2007). Unlike existing ILP methods that use randomly-selected examples, our algorithm learns from a set of representative examples allowing a polynomial efficiency for learning a syntactico-semantic constraint-based grammar, suitable to capture large fragments of natural language (Muresan, 2006).

The LWFG induction algorithm is a cover set algorithm, where at each step a new constraint grammar rule is learned from the current representative example, $\sigma \in E_R$. Then this rule is added to the grammar rule set. The process continues until all the representative examples are covered. We describe below the process of learning a grammar rule from the current representative example, illustrated as well in Figure 2.

Step 1. In the first step, the most specific grammar rule is generated from the current representative example. The category name annotated in the representative example gives the name of the left-hand-side nonterminal ("predicate invention", in ILP terminology), while a robust parser returns the minimum number of chunks covering the representative example. The categories of the chunks give the nonterminals of the right-hand side of the most specific rule. For ex-

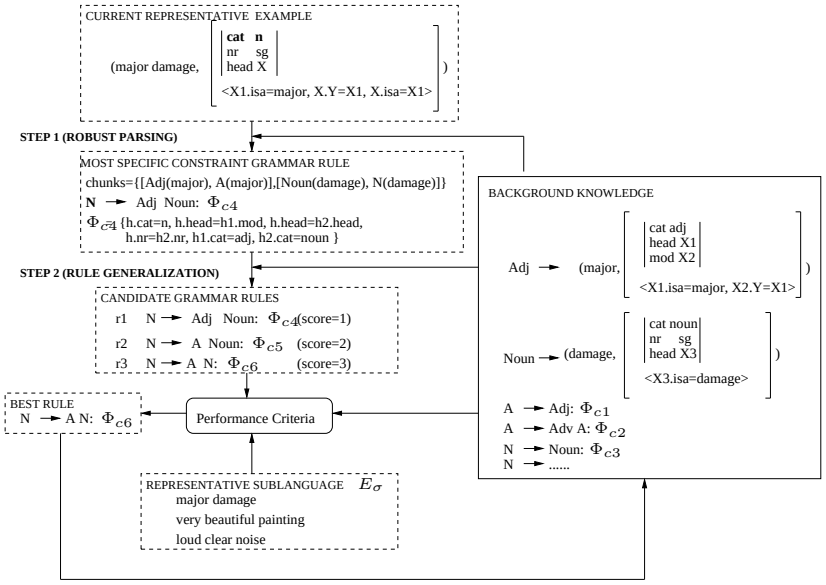


Figure 2: An iteration step of the learning algorithm

ample, in Figure 2, given the representative example *major damage* annotated with its semantic molecule, and the background knowledge containing the already learned rules $A \rightarrow Adj$ and $N \rightarrow Noun$,[1] the robust parser generates the chunks corresponding to the adjective *major* and the noun *damage*: [Adj(major),A(major)] and [Noun(damage),N(damage)], respectively. The most specific rule generated is thus $N \rightarrow Adj\ Noun$: Φ_{c4}, where the left hand side nonterminal is given by the category of the representative example, in this case n. The compositional constraints Φ_{c4} are learned as well. It can be seen that the annotation of the representative example does not require us to provide ontology-specific roles or concepts. Thus, grammar learning is general, and can be done using a small, generic lexicon.

Step 2. In the second step, this most specific rule is generalized, obtaining a set of candidate grammar rules. The performance criterion in choosing the best grammar rule among these candidate hypotheses is the number of the examples in the representative sublanguage E_σ (generalization corpus) that can be parsed using the candidate grammar rule together with the previous learned rules. In Figure 2 given the representative sublanguage E_σ={ *major damage, loud clear noise, very beautiful painting*} the learner will generalize to the recursive rule $N \rightarrow A\ N$: Φ_6, since only this rule can parse all the examples in E_σ.

[1]For readability, we only show the context-free backbone of the grammar rules, and Φ_o are not discussed since they are not learned.

4 Levels of Representation

In order to transform natural language utterances to knowledge, we consider three levels of representation: the utterance level, the text level and the ontology level. In Section 4.4 we show that these levels of representation allow us to define meaning as *Meaning=Text+all Questions/Answers* w.r.t that *Text*, using a DAG matching approach.

4.1 Utterance-level Representation

At the *utterance level*, the semantic representation corresponds directly to a syntagma σ after the ontology constraint Φ_o is applied. This representation is called Ontology-based Semantic Representation **OntoSeR.** At this level, the attrIDs are instantiated with values of the slots from the ontology, while the conceptIDs remain variables to allow further composition to take place. At OntoSeR level we can exploit the reversibility of the grammar, since this representation is used during parsing/generation.

In Figure 3 we show the semantic representation OntoSeR for the utterance *Hepatitis B is an acute viral hepatitis caused by a virus that tends to persist in the blood serum*, obtained using our parser in conjunction with our learned grammar. The composition constraints bind the conceptID variables, while the ontology constraint instantiates the attrID variables with values of slots in the ontology. The ontology constraint can be

Hepatitis B is an acute viral hepatitis caused by a virus that tends to persist in the blood serum.

OntoSeR = ⟨(A.name=hepatitisB)$_{HepatitisB}$, (A.tense=pr)$_{is}$, (A.det=an)$_{an}$, (B.is_a=acute, A.duration=B)$_{acute}$, (C.is_a=viral, A.kind_of=C)$_{viral}$, (A.is_a=hepatitis)$_{hepatitis}$, (D.vft=ed, D.voice=pas, D.is_a=cause, D.ag=E, D.th=A)$_{caused}$, (ag.is_a=by, D.ag=E)$_{by}$, (E.det=a)$_{a}$, (E.is_a=virus)$_{virus}$, (E.is_a=that)$_{that}$, (F.tense=pr, F.is_a=tend, F.no_ag=E, F.prop=G)$_{tends}$, (G.vft=to, G.is_a=persist, G.th=E)$_{to\ persist}$, (loc.is_a=in, G.loc=H)$_{in}$, (H.det=the)$_{the}$, (I.is_a=blood, H.of=I)$_{blood}$, (H.is_a=serum)$_{serum}$ ⟩

TKR

```
~29.name= hepatitisB      ~33.det= virus
~29.tense= pr             ~33.is_a= that
~20.det= an               ~34.tense= pr
~30.is_a= acute           ~34.is_a= tend
~29.duration=~2           ~34.no_role=~33
~31.is_a= viral           ~34.prop=~35
~29.kind_of=~3            ~35.vft= to
~29.is_a= hepatitis       ~35.is_a= persist
~32.vft= ed               ~35.th=~33
~32.voice= pas            loc.is_a= in
~32.is_a= cause           ~35.loc=~36
~32.ag=~5                 ~36.det= the
~32.th=~1                 ~37.is_a= blood
ag.is_a= by              ~36.of=~37
~32.ag=~33                 ~36.is_a= serum
~33.det= a
```

OKR

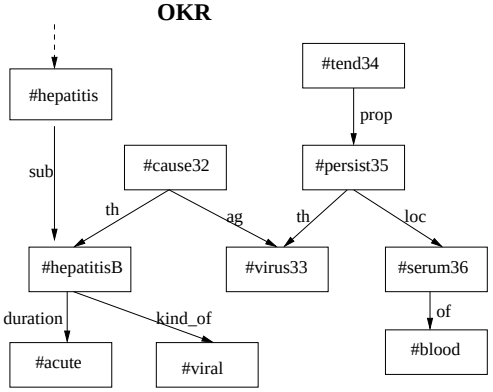


Figure 3: Example of an utterance and its levels of representation

seen as a *local semantic interpretation* at the utterance/grammar rule level, providing access to meaning during parsing/generation. In this paper, this semantic interpretation is based only on a weak "ontological model". For the verb thematic roles we considered the thematic roles derived from Dorr's LCS Database (e.g., ag=agent, th=theme, prop=proposition) (Dorr, 1997). For adjectives and adverbs we took the roles (properties) from WordNet (Miller, 1990). For prepositions we considered the LCS Database. We also have manually added specific/dummy semantic roles when they were not present in these resources (e.g., of between *blood* and *serum*).

The example in Figure 3 shows the output of our parser in conjunction with the learned grammar for a definitional sentence that contains several linguistic phenomena such as copula *to-be* predicative, reduced relative clauses (*caused by ...*), relative clauses (*virus that ...*), raising construction (*tends to persist*, where *virus* is not the argument of *tends* but the argument of *persist*), and noun compounds (*blood serum*). For readability, we indicate what part of OntoSeR corresponds to each lexical item. It can be noticed that OntoSeR contains representations of both ontological meaning (concepts and relations among concepts) as well as extra-ontological meaning such as tense and voice ($D.voice = pas$; $F.tense = pr$).

4.2 Text-level Representation

The *text-level* representation **TKR**, or discourse level representation, represents asserted representations. ConceptIDs become constants, and no composition can happen at this level. However, we still have (indirect) reversibility, since TKR represents all the asserted OntoSeRs. Therefore, all the information needed for reversibility is still present. Figure 3 shows an example of the TKR for the above utterance.

4.3 Ontology-level Representation

Ontology-level knowledge representation **OKR** is obtained after task-specific interpretation, which can be seen as a *global semantic interpretation*. OKR is a directed acyclic graph (DAG) $G = (V, E)$. Edges, E, are either semantic roles given by verbs, prepositions, adjectives and adverbs, or extra-ontological meaning properties, such as tense, aspect, modality, negation. Vertices, V are either concepts (corresponding to nouns, verbs, adjectives, adverbs, pronouns, cf. Quine's criterion (Sowa, 1999, page 496)), or values of the extra-ontological properties such as present corresponding to tense property. In this paper, the task-specific interpretation is geared mainly towards terminological interpretation. We filter from OntoSeR determiners and some verb forms, such as tense, aspect, since temporal relations appear less in terminological knowledge than in factual

knowledge. However, we treat modals and negation, as they are relevant for terminological knowledge. An example of OKR for the above utterance is given in Figure 3.

We consider both concepts (e.g., #acute, #blood), and instances of concepts (e.g., #virus33, #cause32). Concepts are denoted in OKR by #name_concept, and they form a hierarchy of concepts based on the subsume relation (sub), which is the inverse of the is_a relation. An instance of a concept is denoted by the name of a concept followed by the instance number (e.g., #virus33). A concept and an instance of this concept are two different vertices in OKR, having the same name. At the OKR level we assume the *principle of concept identity* which means that there is a bijection between a vertex in OKR and a referent. For example, if we do not have pronoun resolution, the pronoun and the noun it refers to will be represented as two separate vertices in the graph. Currently, our semantic interpreter implements only a *weak concept identity principle* which facilitates structure sharing and inheritance.

To give these two properties we first introduce some notations. A DAG is called *rooted* at a vertex $u \in V$, if there exists a path from u to each vertex of the DAG. We have the following definition:

Definition 1. *Two subDAGs rooted at two vertices u, u' are equal if the set of the adjacent vertices to u and u' respectively, are equal and if the edges incident from u and u' have the same semantic roles as labels.*

Property 1 (Structure Sharing)**.** *In an OKR, all vertices $u, u' \in V$ with the same name, and whose subDAGs are equal are identical (i.e., the same vertex in OKR).*

Using a hash table, there is a linear algorithm $O(|V| + |E|)$ which transforms an OKR to an equivalent OKR which satisfies Property 1. In Figure 4 it can be seen that the OKRs of Hepatitis A and Hepatitis B share the representation corresponding to *blood serum* (i.e., *blood serum* is the same concept instance and due to Property 1 we have that #serum36=#serum27 and thus they have the same vertex in the OKR).

Property 2 (Inheritance)**.** *A concept in a hierarchy of concepts can be linked by the* sub *relation only to its parent(s), and not to any other ancestors. A subDAG defining a property of a concept from the hierarchy of concepts can be found only once in the OKR at the level of the most general concept that has this property.*

For terminological knowledge we have that any instance of a concept is a concept, and the definition is the naming of a concept instance. For example, the definition of Hepatitis B, is an instance of a concept #hepatitis which has additional attributes *acute*, *viral* and *caused by a virus that tends to persist in the blood serum*. Thus, an additional instance of concept #hepatitis is created, which is named #hepatitisB. The fact that we can have the definition as a naming of a concept instance is facilitated also by our treatment of copula *to-be* at the OntoSeR level ($A.name = hepatitisB, \ldots, A.is\_a = hepatitis$ in Figure 3)

4.4 Meaning as Answers to Questions

We consider that "understanding" a text is the ability to correctly answer, at the conceptual level, all the questions asked w.r.t to that text, and thus *Meaning = Text + all Questions/Answers* w.r.t that *Text*. In our framework we consider the principle of natural language as problem formulation, and not problem solving. Thus, we can represent at OKR level a paradox formulation in natural language, even if the reasoning about its solution cannot be emphasized. Our levels of representations allow us to define the meaning of questions, answers and utterances using a DAG matching approach.

Definition 2. *The meaning of a question, q, with respect to an utterance/discourse, is the set of all answers that can be directly obtained from that utterance/discourse. The semantic representation of a question is a subgraph of the utterance graph where the* wh-word *substitutes the answer concept(s).*

Definition 3. *The answer to a question is the concept that matches the wh-word through the DAG matching algorithm between the question's subDAG and the utterance/discourse DAG.*

Definition 4. *The meaning of an utterance u is the set of all questions that can be asked w.r.t that utterance, together with their answers.*

Unlike meaning as truth conditions, where the problem of meaning equivalence is reduced to logical form equivalence, in our case meaning equivalence is reduced to semantic equivalence of DAGs/subDAGs which obey the concept identity principle (weak, or strong). The matching algo-

rithm obtains the same answers to questions, relative to semantic equivalent DAGs. If we consider only the weak concept identity principle given by Properties 1 and 2, the problem is reduced to DAG/subDAG identity.

5 Discussion

The grammar formalism, learning model and our ontology-based representation allow us to directly map text to graph-based meaning representations. Our method relies on a general grammar learning framework and a task-specific semantic interpreter. Learning is done based on annotated examples that do not contain ontology-specific roles or concepts as we saw in Section 3, and thus our learning framework is general. We can use any ontology, depending on the application. The task-specific semantic interpreter we are currently using is targeted for terminological knowledge, and uses a weak "ontological model" based on admissibility relations we can find at the level of lexical entries and a weak concept identity principle.

In (Muresan, 2006) we showed that our grammar formalism and induction model allow us to learn diverse and complex linguistic phenomena: complex noun phrases (e.g., noun compounds, nominalization), prepositional phrases, reduced relative clauses, finite and non-finite verbal constructions (including, tense, aspect, negation), coordination, copula *to be*, raising and control constructions, and rules for wh-questions (including long-distance dependencies).

In this section we discuss the processes of knowledge acquisition and natural language querying, by presenting an example of constructing terminological knowledge from definitions of *hepatitis*, *Hepatitis A* and *Hepatitis B*. The definitional text and OKRs are presented in Figure 4, OKR being shown only for the last two definitions for readability reasons. A question and answer related to the resulting OKR are also given.

The definiendum is always a concept, and it is part of the sub hierarchy. The concepts in the sub hierarchy are presented in bold in Figure 4. In addition to the concepts that are defined, we can also have *concepts that are referred* (i.e., they are part of the definiens), *if they do not have any modification* (e.g., #blood in definition of Hepatitis A, and Hepatitis B). If a referred concept has modifications, it is represented as an instance of a concept in OKR. As a consequence, various verbalizations of concept properties can be differentiated in OKR, allowing us to obtain direct answers that are specific to each verbalization. For example, the term *virus* appears in the definition of both *Hepatitis A* and *Hepatitis B*. In OKR, they are two different instances of a concept, #virus25 and #virus33, since they have different modifications: *persists in the blood serum*, *does not persists in the blood serum*, respectively. These modifications are an essential part of the *differentia* of the two concepts #hepatitisA and #hepatitisB, causing the distinction between the two. When we ask the question *What is caused by a virus that persists in the blood serum?* we obtain only the correct answer #hepatitisB (Figure 4).

Another important aspect that shows the adequacy of our representation for direct acquisition and query is the OKR-equivalences that we obtain for different syntactic forms. They are related mainly to verbal constructions. Among OKR-equivalences we have: 1) active and passive constructions; 2) *-ed* and *-ing* verb forms in reduced relative clauses are equivalent to passive/active verbal constructions; 3) constructions involving raising verbs, where we can take advantage of the fact that the controller is not the semantic argument of the raising verb (e.g., in the definition of Hepatitis B we have ... *caused by a virus that tends to persist in the blood serum*, while the question can be asked without the raising verb *What is caused by a virus that persists in the blood serum?*; see Figure 4).

Besides acquisition of terminological knowledge, our grammar and semantic interpreter facilitates natural language querying of the acquired knowledge base, by treatment of wh-questions. Querying is a DAG matching problem, where the wh-word is matched to the answer concept.

6 Conclusions

This paper has presented a learning framework to automatically map natural language to graph-based meaning representations via grammar induction. We presented an ontology-based semantic representation that allows us to define meaning as *Meaning=Text+all Questions/Answers* w.r.t that *Text*, using a DAG matching approach.

In the future, we plan to extend this work in two main directions. First, we plan to use a stronger semantic context with hierarchies of concepts and semantic roles, selectional restrictions, as well as

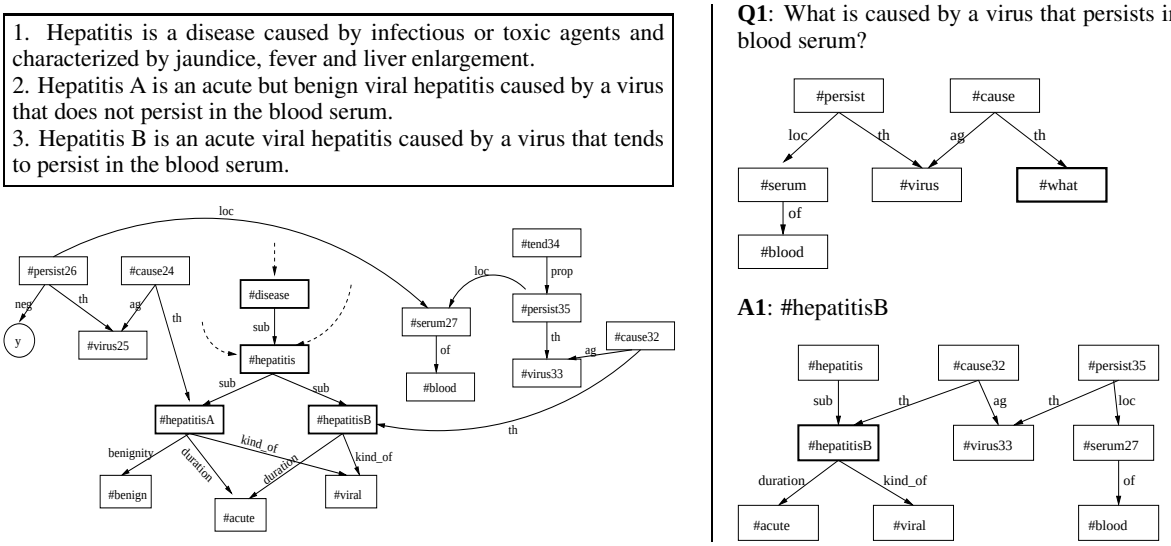


Figure 4: Acquisition/Query of terminological knowledge

semantic equivalences based on synonymy and anaphora. The second direction is to enhance the ontology with probabilities.

References

Basili, Roberto, Dorte H. Hansen, Patrizia Paggio, Maria Teresa Pazienza, and Fabio Zanzotto. 2004. Ontological resources and question answering. In *Workshop on Pragmatics of Question Answering, held jointly with NAACL 2004*.

Beale, Stephen, Benoit Lavoie, Marjorie McShane, Sergei Nirenburg, and Tanya Korelsky. 2004. Question answering using ontological semantics. In *ACL 2004: Second Workshop on Text Meaning and Interpretation*.

Bos, Johan, Stephen Clark, Mark Steedman, James R. Curran, and Julia Hockenmaier. 2004. Wide-coverage semantic representations from a CCG parser. In *Proceedings of COLING-04*.

Collins, Michael. 1999. *Head-Driven Statistical Models for Natural Language Parsing*. Ph.D. thesis, University of Pennsylvania.

Copestake, Ann, Dan Flickinger, Ivan A. Sag, and Carl Pollard. 1999. Minimal Recursion Semantics: An introduction.

Dorr, Bonnie J. 1997. Large-scale dictionary construction for foreign language tutoring and interlingual machine translation. *Machine Translation*, 12(4):271–322.

Dzeroski, Saso. 2007. Inductive logic programming in a nutshell. In Getoor, Lise and Ben Taskar, editors, *Introduction to Statistical Relational Learning*. The MIT Press.

Ge, Ruifang and Raymond J. Mooney. 2005. A statistical semantic parser that integrates syntax and semantics. In *Proceedings of CoNLL-2005*.

He, Yulan and Steve Young. 2006. Spoken language understanding using the hidden vector state model. *Speech Communication Special Issue on Spoken Language Understanding in Conversational Systems*, 48(3-4).

Hovy, Eduard, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 2006. Ontonotes: The 90% solution. In *Proceedings of HLT-NAACL 2006*.

Miller, George. 1990. WordNet: An on-line lexical database. *Journal of Lexicography*, 3(4):235–312.

Muggleton, Stephen. 1995. Inverse Entailment and Progol. *New Generation Computing, Special Issue on Inductive Logic Programming*, 13(3-4):245–286.

Muresan, Smaranda and Owen Rambow. 2007. Grammar approximation by representative sublanguage: A new model for language learning. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Muresan, Smaranda. 2006. Learning constraint-based grammars from representative examples: Theory and applications. Technical report, PhD Thesis, Columbia University.

Nirenburg, Sergei and Victor Raskin. 2004. *Ontological Semantics*. MIT Press.

Pereira, Fernando C. and David H.D Warren. 1980. Definite Clause Grammars for language analysis. *Artifi cial Intelligence*, 13:231–278.

Shieber, Stuart, Hans Uszkoreit, Fernando Pereira, Jane Robinson, and Mabry Tyson. 1983. The formalism and implementation of PATR-II. In Grosz, Barbara J. and Mark Stickel, editors, *Research on Interactive Acquisition and Use of Knowledge*, pages 39–79. SRI International, Menlo Park, CA, November.

Sowa, John F. 1999. *Knowledge Representation: Logical, Philosophical, and Computational Foundations*. Brooks Cole Publishing Co., Pacific Grove, CA.

Wong, Yuk Wah and Raymond Mooney. 2007. Learning synchronous grammars for semantic parsing with lambda calculus. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL-2007)*.

Zettlemoyer, Luke S. and Michael Collins. 2005. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. In *Proceedings of UAI-05*.

How Is Meaning Grounded in Dictionary Definitions?

A. Blondin Massé
Laboratoire de combinatoire et d'informatique mathématique
Université du Québec à Montréal
Montréal (QC), CANADA H3C 3P8
alexandre.blondin.masse@gmail.com

G. Chicoisne, Y. Gargouri, S. Harnad, O. Picard
Institut des sciences cognitives
Université du Québec à Montréal
Montréal (QC), CANADA H3C 3P8
chicoisne.guillaume@uqam.ca, yassinegargouri@hotmail.com
harnad@ecs.soton.ac.uk, olivierpicard18@hotmail.com

O. Marcotte
Groupe d'études et de recherche en analyse des décisions (GERAD) and UQÀM
HEC Montréal
Montréal (Québec) Canada H3T 2A7
Odile.Marcotte@gerad.ca

Abstract

Meaning cannot be based on dictionary definitions all the way down: at some point the circularity of definitions must be broken in some way, by grounding the meanings of certain words in sensorimotor categories learned from experience or shaped by evolution. This is the "symbol grounding problem". We introduce the concept of a *reachable* set — a larger vocabulary whose meanings can be learned from a smaller vocabulary through definition alone, as long as the meanings of the smaller vocabulary are themselves already grounded. We provide simple algorithms to compute reachable sets for any given dictionary.

1 Introduction

We know from the 19th century philosopher-mathematician Frege that the *referent* and the *meaning* (or "sense") of a word (or phrase) are not the same thing: two different words or phrases can refer to the very same object without having the same meaning (Frege, 1948): "George W. Bush" and "the current president of the United States of America" have the same referent but a different meaning. So do "human females" and "daughters". And "things that are bigger than a breadbox" and "things that are not the size of a breadbox or smaller".

A word's "extension" is the set of things to which it refers, and its "intension" is the rule for defining what things fall within its extension.. A word's meaning is hence something closer to *a rule for picking out its referent.* Is the dictionary definition of a word, then, its meaning?

Clearly, if we do not know the meaning of a word, we look up its definition in a dictionary. But what if we do not know the meaning of any of the words in its dictionary definition? And what if we don't know the meanings of the words in the definitions of the words defining those words, and so on? This is a problem of infinite regress, called the "symbol grounding problem" (Harnad, 1990; Harnad, 2003): the meanings of words in dictionary definitions are, in and of themselves, ungrounded. The meanings of some of the words, at least, have to be grounded by some means other than dictionary definition look-up.

How are word meanings grounded? Almost certainly in the sensorimotor capacity to pick out their referents (Harnad, 2005). Knowing *what to do with what* is not a matter of definition but of adaptive sensorimotor interaction between autonomous, behaving systems and categories of "objects" (including individuals, kinds, events, actions, traits and states). Our embodied sensorimotor systems can also be described as applying information processing rules to inputs in order to generate the right outputs, just as a thermostat defending a temperature of 20 degrees can be. But this dynamic process is in no useful way analogous to looking up a definition in a dictionary.

We will not be discussing sensorimotor grounding (Barsalou, 2008; Glenberg & Robertson, 2002; Steels, 2007) in this paper. We will assume some sort of grounding as given: when we consult a dictionary, we already know the meanings of at least some words,

Coling 2008: Proceedings of 3rd Textgraphs workshop on Graph-Based Algorithms in Natural Language Processing, pages 17–24
Manchester, August 2008

somehow. A natural first hypothesis is that the grounding words ought to be more concrete, referring to things that are closer to our overt sensorimotor experience, and learned earlier, but that remains to be tested (Clark, 2003). Apart from the question of the boundary conditions of grounding, however, there are basic questions to be asked about the structure of word meanings in dictionary definition space.

In the path from a word, to the definition of that word, to the definition of the words in the definition of that word, and so on, through what sort of a structure are we navigating (Ravasz & Barabasi, 2003; Steyvers & Tenenbaum, 2005)? Meaning is compositional: A definition is composed of words, combined according to syntactic rules to form a proposition (with a truth value: true or false). For example, the word to be defined w (the "definiendum") might mean $w_1 \,\&\, w_2 \,\&\, \ldots \,\&\, w_n$, where the w_i are other words (the "definientes") in its definition. Rarely does that proposition provide the full necessary and sufficient conditions for identifying the referent of the word, w, but the approximation must at least be close enough to allow most people, armed with the definition, to understand and use the defined word most of the time, possibly after looking up a few of its definientes d_w, but without having to cycle through the entire dictionary, and without falling into circularity or infinite regress.

If enough of the definientes are grounded, then there is no problem of infinite regress. But we can still ask the question: What is the size of the grounding vocabulary? and what words does it contain? What is the length and shape of the path that would be taken in a recursive definitional search, from a word, to its definition, to the definition of the words in its definition, and so on? Would it eventually cycle through the entire dictionary? Or would there be disjoint subsets?

This paper raises more questions than it answers, but it develops the formal groundwork for a new means of finding the answers to questions about how word meaning is *explicitly* represented in real dictionaries — and perhaps also about how it is *implicitly* represented in the "mental lexicon" that each of us has in our brain (Hauk et al., 2008).

The remainder of this paper is organized as follows: In Section 2, we introduce the graph-theoretical definitions and notations used for formulating the symbol grounding problem in Section 3. Sections 4 and 5 deal with the implication of this approach in cognitive sciences and show in what ways grounding kernels may be useful.

2 Definitions and Notations

In this section, we give mathematical definitions for the dictionary-related terminology, relate them to natural language dictionaries and supply the pertinent graph theoretical definitions. Additional details are given to ensure mutual comprehensibility to specialists in the three disciplines involved (mathematics, linguistics and psychology). Complete introductions to graph theory and discrete mathematics are provided in (Bondy & Murty, 1978; Rosen, 2007).

2.1 Relations and Functions

Let A be any set. A *binary relation on* A is any subset R of $A \times A$. We write xRy if $(x, y) \in R$. The relation R is said to be (1) *reflexive* if for all $x \in A$, we have xRx, (2) *symmetric* if for all $x, y \in A$ such that xRy, we have yRx and (3) *transitive* if for all $x, y, z \in A$ such that xRy and yRz, we have xRz. The relation R is an *equivalence relation* if it is reflexive, symmetric and transitive. For any $x \in A$, the *equivalence class of* x, designated by $[x]$, is given by $[x] = \{y \in A \mid xRy\}$. It is easy to show that $[x] = [y]$ if and only if xRy and that the set of all equivalence classes forms a partition of A.

Let A be any set, $f : A \to A$ a function and k a positive integer. We designate by f^k the function $f \circ f \circ \ldots \circ f$ (k times), where $\circ$ denotes the *composition of functions*.

2.2 Dictionaries

At its most basic level, a dictionary is a set of associated pairs: a *word* and its *definition*, along with some disambiguating parameters. The *word*[1] to be defined, w, is called the *definiendum* (plural: *definienda*) while the finite nonempty set of words that defines w, d_w, is called the set of *definientes* of w (singular: *definiens*).

Each dictionary entry accordingly consists of a definiendum w followed by its set of definientes d_w. A *dictionary* D then consists of a finite set of pairs (w, d_w) where w is a word and $d_w = \{w_1, w_2, \ldots, w_n\}$, where $n \geq 1$, is its definition, satisfying the property that for all $(w, d_w) \in D$ and for all $d \in d_w$, there exists $(w', d_{w'}) \in D$ such that $d = w'$. A pair (w, d_w) is called an *entry* of D. In other words, a dictionary is a finite set of words, each of which is defined, and each of its defining words is likewise defined somewhere in the dictionary.

2.3 Graphs

A *directed graph* is a pair $G = (V, E)$ such that V is a finite set of *vertices* and $E \subseteq V \times V$ is a finite set of *arcs*. Given $V' \subseteq V$, the *subgraph induced by* V', designated by $G[V']$, is the graph $G[V'] = (V', E')$ where $E' = E \cap (V' \times V')$. For any $v \in V$, $N^-(v)$ and $N^+(v)$ designate, respectively, the set of incoming and outgoing neighbors of v, i.e.

$$\begin{aligned} N^-(v) &= \{u \in V \mid (u, v) \in E\} \\ N^+(v) &= \{u \in V \mid (v, u) \in E\}. \end{aligned}$$

We write $\deg^-(v) = |N^-(v)|$ and $\deg^+(v) = |N^+(v)|$, respectively. A *path* of G is a sequence

[1] In the context of this mathematical analysis, we will use "word" to mean a finite string of uninterrupted letters having some associated meaning.

$(v_1, v_2, \ldots, v_n)$, where n is a positive integer, $v_i \in V$ for $i = 1, 2, \ldots, n$ and $(v_i, v_{i+1}) \in E$, for $i = 1, 2, \ldots, n-1$. A *uv-path* is a path starting with u and ending with v. Finally, we say that a uv-path is a *cycle* if $u = v$.

Given a directed graph $G = (V, E)$ and $u, v \in V$, we write $u \to v$ if there exists a uv-path in G. We define a relation $\sim$ as

$$u \sim v \Leftrightarrow u \to v \text{ and } v \to u.$$

It is an easy exercise to show that $\sim$ is an equivalence relation. The equivalence classes of V with respect to $\sim$ are called the *strongly connected components* of G. In other words, in a directed graph, it might be possible to go directly from point A to point B, without being able to get back from point B to point A (as in a city with only one-way streets). Strongly connected components, however, are subgraphs in which whenever it is possible to go from point A to point B, it is also possible to come back from point B to point A (the way back may be different).

There is a very natural way of representing definitional relations using graph theory, thus providing a formal tool for analyzing grounding properties of dictionaries: words can be represented as vertices, with arcs representing definitional relations, i.e. there is an arc (u, v) between two words u and v if the word u appears in the definition of the word v. More formally, for every dictionary D, its *associated graph* $G = (V, E)$ is given by

$$\begin{aligned} V &= \{w \mid \exists d_w \text{ such that } (w, d_w) \in D\}, \\ E &= \{(v, w) \mid \exists d_w \text{ such that } (w, d_w) \in D \text{ and } \\ &\quad v \in d_w\}. \end{aligned}$$

Note that every vertex v of G satisfies $\deg^-_G(v) > 0$, but it is possible to have $\deg^+_G(v) = 0$. In other words, whereas every word has a definition, some words are not used in any definition.

Example 1. *Let D be the dictionary whose definitions are given in Table 1. Note that every word appearing in some definition is likewise defined in D (this is one of the criteria for D to be a dictionary). The associated graph G of D is represented in Figure 1. Note that (not, good, eatable, fruit) is a path of G while (good, bad, good) is a cycle (as well as a path) of G.*

3 A Graph-Theoretical Formulation of the Problem

We are now ready to formulate the symbol grounding problem from a mathematical point of view.

3.1 Reachable and Grounding Sets

Given a dictionary D of n words and a person x who knows m out of these n words, assume that the only way x can learn new words is by consulting the dictionary definitions. Can all n words be learned by x through dictionary look-up alone? If not, then exactly what subset of words can be learned by x through dictionary look-up alone?

Word	Definition	Word	Definition
apple	red fruit	bad	not good
banana	yellow fruit	color	dark or light
dark	not light	eatable	good
fruit	eatable thing	good	not bad
light	not dark	not	not
or	or	red	dark color
thing	thing	tomato	red fruit
yellow	light color		

Table 1: Definitions of the dictionary D

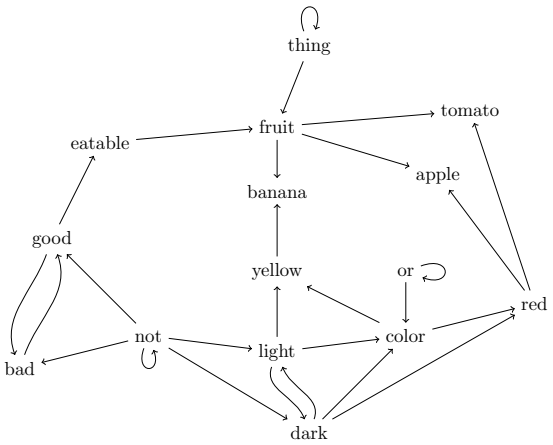


Figure 1: Graph representation of the dictionary D.

For this purpose, let $G = (V, E)$ be a directed graph and consider the following application, where 2^V denotes the collection of all subsets of V:

$$\begin{aligned} R_G : \quad 2^V &\longmapsto 2^V \\ U &\longmapsto U \cup \{v \in V \mid N^-(v) \subseteq U\}. \end{aligned}$$

When the context is clear, we omit the subscript G. Also we let R^k denote the k^{th} power of R. We say that $v \in V$ is *k-reachable from U* if $v \in R^k(U)$ and k is a nonnegative integer. It is easy to show that there exists an integer k such that $R^\ell(U) = R^k(U)$, for every integer $\ell > k$. More precisely, we have the following definitions:

Definition 2. *Let $G = (V, E)$ be a directed graph, U a subset of V, and k an integer such that $R^\ell(U) = R^k(U)$ for all $\ell > k$. The set $R^k(U)$ is called the* reachable set from U *and is denoted by $R^*(U)$. Moreover, if $R^*(U) = V$, then we say that U is a* grounding set *of G.*

We say that G is *p-groundable* if there exists $U \subseteq V$ such that $|U| = p$ and U is a grounding set of G. The *grounding number* of a graph G is the smallest integer p such that G is p-groundable.

Reachable sets can be computed very simply using a breadth-first-search type algorithm, as shown by Algo-

rithm 1.

Algorithm 1 Computing reachable sets

```
1: function REACHABLESET(G, U)
2:     R ← U
3:     repeat
4:         S ← {v ∈ V | N_G^-(v) ⊆ R} − R
5:         R ← R ∪ S
6:     until S = ∅
7:     return R
8: end function
```

We now present some examples of reachable sets and grounding sets.

Example 3. *Consider the dictionary D and the graph G of Example 1. Let $U = \{bad, light, not, thing\}$. Note that*

$$\begin{aligned} R^0(U) &= U \\ R^1(U) &= U \cup \{dark, good\}, \\ R^2(U) &= R^1(U) \cup \{eatable\} \\ R^3(U) &= R^2(U) \cup \{fruit\} \\ R^4(U) &= R^3(U) \end{aligned}$$

so that $R^*(U) = \{bad, dark, eatable, fruit, good, light, not, thing\}$ (see Figure 2). In particular, this means that the word "eatable" is 2-reachable (but not 1-reachable) from U and all words in U are 0-reachable from U. Moreover, we observe that U is not a grounding set of G ("color", for example, is unreachable). On the other hand, the set $U' = U \cup \{or\}$ is a grounding set of G, so that G is 5-groundable.

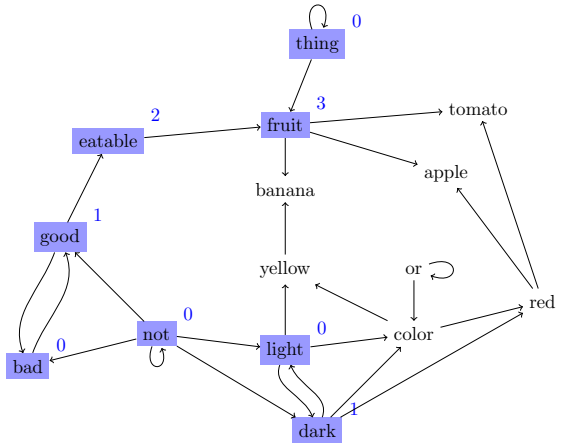


Figure 2: The set $R^*(U)$ (the words in squares) obtained from U

3.2 The Minimum Grounding Set Problem

Given a dictionary and its associated graph G, we are interested in finding minimum grounding sets of G. (Note that in general, there is more than one grounding set of minimum cardinality.) This is related to a natural decision problem: we designate by k-GS the problem of deciding whether G is k-groundable. We show that k-GS is closely related to the problem of finding minimum feedback vertex sets. First, we recall the definition of a feedback vertex set.

Definition 4. *Let $G = (V, E)$ be a directed graph and U a subset of V. We say that U is a* feedback vertex set *of G if for every cycle C of G, we have $U \cap C \neq \emptyset$. In other words, U covers every cycle of G.*

The *minimum feedback vertex set problem* is the problem of finding a feedback vertex set of G of minimum cardinality. To show that feedback vertex sets and grounding sets are the same, we begin by stating two simple lemmas.

Lemma 5. *Let $G = (V, E)$ be a directed graph, C a cycle of G and $U \subseteq V$ a grounding set of G. Then $U \cap C \neq \emptyset$.*

Proof. By contradiction, assume that $U \cap C = \emptyset$ and, for all $v \in C$, there exists an integer k such that v belongs to $R^k(U)$. Let ℓ be the smallest index in the set $\{k \mid \exists u \in C$ such that $u \in R^k(U)\}$. Let u be a vertex in $C \cap R^\ell(U)$ and w the predecessor of u in C. Since $U \cap C = \emptyset$, k must be greater than 0 and w a member of $R^{\ell-1}(U)$, contradicting the minimality of ℓ. □

Lemma 6. *Every directed acyclic graph G is 0-groundable.*

Proof. We prove the statement by induction on $|V|$.

BASIS. If $|V| = 1$, then $|E| = 0$, so that the only vertex v of G satisfies $N_G^-(v) = \emptyset$. Hence $R(\emptyset) = V$.

INDUCTION. Let v be a vertex such that $\deg^+(v) = 0$. Such a vertex exists since G is acyclic. Moreover, let G' be the (acyclic) graph obtained from G by removing vertex v and all its incident arcs. By the induction hypothesis, there exists an integer k such that $R^k_{G'}(\emptyset) = V - \{v\}$. Therefore, $V - \{v\} \subseteq R^k_G(\emptyset)$ so that $R^{k+1}_G(\emptyset) = V$. □

The next theorem follows easily from Lemmas 5 and 6.

Theorem 7. *Let $G = (V, E)$ be a directed graph and $U \subseteq V$. Then U is a grounding set of G if and only if U is a feedback vertex set of G.*

Proof. ($\Rightarrow$) Let C be a cycle of G. By Lemma 5, $U \cap C \neq \emptyset$, so that U is a minimum feedback vertex set of G. ($\Leftarrow$) Let G' be the graph obtained from G by removing U. Then G' is acyclic and $\emptyset$ is a grounding set of G'. Therefore, $U \cup \emptyset = U$ is a grounding set of G. □

Corollary 8. *k-GS is NP-complete.*

Proof. Denote by k-FVS the problem of deciding whether a directed graph G admits a feedback vertex set of cardinality at most k. This problem is known to be NP-complete and has been widely studied (Karp, 1972; Garey & Johnson, 1979). It follows directly from Theorem 7 that k-GS is NP-complete as well since the problems are equivalent. □

The fact that problems k-GS and k-FVS are equivalent is not very surprising. Indeed, roughly speaking, the minimum grounding problem consists of finding a minimum set large enough to enable the reader to learn (reach) all the words of the dictionary. On the other hand, the minimum feedback vertex set problem consists of finding a minimum set large enough to break the circularity of the definitions in the dictionary. Hence, the problems are the same, even if they are stated differently.

Although the problem is NP-complete in general, we show that there is a simple way of reducing the complexity of the problem by considering the strongly connected components.

3.3 Decomposing the Problem

Let $G = (V, E)$ be a directed graph and G_1, G_2, ..., G_m the subgraphs induced by its strongly connected components, where $m \geq 1$. In particular, there are no cycles of G containing vertices in different strongly connected components. Since the minimum grounding set problem is equivalent to the minimum feedback vertex set problem, this means that when seeking a minimum grounding set of G, we can restrict ourselves to seeking minimum grounding sets of G_i, for $i = 1, 2, \ldots, m$. More precisely, we have the following proposition.

Proposition 9. *Let $G = (V, E)$ be a directed graph with m strongly connected components, with $m \geq 1$, and let $G_i = (V_i, E_i)$ be the subgraph induced by its i-th strongly connected component, where $1 \leq i \leq m$. Moreover, let U_i be a minimum grounding set of G_i, for $i = 1, 2, \ldots, m$. Then $U = \bigcup_{i=1}^{m} U_i$ is a minimum grounding set of G.*

Proof. First, we show that U is a grounding set of G. Let C be a cycle of G. Then C is completely contained in some strongly connected component of G, say G_j, where $1 \leq j \leq m$. But $U_j \subseteq U$ is a grounding set of G_j, therefore $U_j \cap C \neq \emptyset$ so that $U \cap C \neq \emptyset$. It remains to show that U is a minimum grounding set of G. By contradiction, assume that there exists a grounding set U' of G, with $|U'| < |U|$ and let $U'_i = U' \cap V_i$. Then there exists an index j, with $1 \leq j \leq m$, such that $|U'_j| < |U_j|$, contradicting the minimality of $|U_j|$. □

Note that this proposition may be very useful for graphs having many small strongly connected components. Indeed, by using Tarjan's Algorithm (Tarjan, 1972), the strongly connected components can be computed in linear time. We illustrate this reduction by an example.

Example 10. *Consider again the dictionary D and the graph G of Example 1. The strongly connected components of G are encircled in Figure 3 and minimum grounding sets (represented by words in squares) for each of them are easily found. Thus the grounding number of G is 5.*

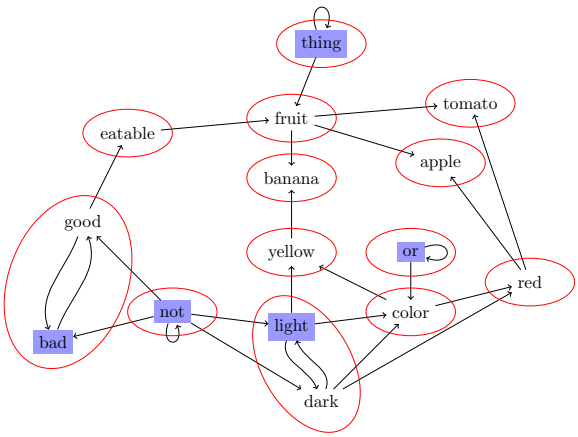


Figure 3: The strongly connected components and a minimum grounding set of G

3.4 The Grounding Kernel

In Example 10, we have seen that there exist some strongly connected components consisting of only one vertex without any loop. In particular, there exist vertices with no successor, i.e. vertices v such that $N_G^+(v) = 0$. For instance, this is the case of the words "apple", "banana" and "tomato", which are not used in any definition in the dictionary. Removing these three words, we notice that "fruit", "red" and "yellow" are in the same situation and they can be removed as well. Pursuing the same idea, we can now remove the words "color" and "eatable". At this point, we cannot remove any further words. The set of remaining words is called the *grounding kernel* of the graph G. More formally, we have the following definition..

Definition 11. *Let D be a dictionary, $G = (V, E)$ its associated graph and $G_1 = (V_1, E_1)$, $G_2 = (V_2, E_2)$, ..., $G_m = (V_m, E_m)$ the subgraphs induced by the strongly connected components of G, where $m \geq 1$. Let V' be the set of vertices u such that $\{u\}$ is a strongly connected component without any loop (i.e., (u, u) is not an arc of G). For any u, let $N^*(u)$ denote the set of vertices v such that G contains a uv-path. Then the* grounding kernel *of G, denoted by K_G, is the set $V - \{u \mid u \in V' \text{ and } N^*(u) \subseteq V'\}$.*

Clearly, every dictionary D admits a grounding kernel, as shown by Algorithm 2. Moreover, the grounding kernel is a grounding set of its associated graph G and every minimum grounding set of G is a subset of the grounding kernel. Therefore, in studying the symbol grounding problem in dictionaries, we can restrict

Algorithm 2 Computing the grounding kernel

```
function GroundingKernel(G)
    G' ← G
    repeat
        Let W be the set of vertices of G'
        U ← {v ∈ W | N+_G'(v) = ∅}
        G' ← G'[W − U]
    until U = ∅
    return G'
end function
```

ourselves to the grounding kernel of the graph G corresponding to D. This phenomenon is interesting because every dictionary contains many words that can be recursively removed without compromising the understanding of the other definitions. Formally, this property relates to the *level* of a word: we will say of a word w that it is of *level* k if it is k-reachable from K_G but not ℓ-reachable from K_G, for any $\ell < k$. In particular, level 0 indicates that the word is part of the grounding kernel. A similar concept has been studied in (Changizi, 2008).

Example 12. *Continuing Example 10 and from what we have seen so far, it follows that the grounding kernel of G is given by*

$$K_G = \{\textit{bad, dark, good, light, not, or, thing}\}.$$

Level 1 *words are "color" and "eatable", level* 2 *words are "fruit", "red" and "yellow", and level* 3 *words are "apple", "banana" and "tomato".*

4 Grounding Sets and the Mental Lexicon

In Section 3, we introduced all the necessary terminology to study the symbol grounding problem using graph theory and digital dictionaries. In this section, we explain how this model can be useful and on what assumptions it is based.

A dictionary is a formal symbol system. The preceding section showed how formal methods can be applied to this system in order to extract formal features. In cognitive science, this is the basis of *computationalism* (or cognitivism or "disembodied cognition" (Pylyshyn, 1984)), according to which cognition, too, is a formal symbol system – one that can be studied and explained independently of the hardware (or, insofar as it concerns humans, the wetware) on which it is implemented. However, pure computationalism is vulnerable to the problem of the grounding of symbols too (Harnad, 1990). Some of this can be remedied by the competing paradigm of embodied cognition (Barsalou, 2008; Glenberg & Robertson, 2002; Steels, 2007), which draws on dynamical (noncomputational) systems theory to ground cognition in sensorimotor experience. Although computationalism and symbol grounding provide the background context for our investigations and findings, the present paper does not favor any particular theory of mental representation of meaning.

A dictionary is a symbol system that relates words to words in such a way that the meanings of the definienda are conveyed via the definientes. The user is intended to arrive at an understanding of an unknown word through an understanding of its definition. What was formally demonstrated in Section 3 agrees with common sense: although one can learn new word meanings from a dictionary, the entire dictionary cannot be learned in this way because of circular references in the definitions (*cycles*, in graph theoretic terminology). Information – *nonverbal* information – must come from outside the system to ground at least some of its symbols by some means other than just formal definition (Cangelosi & Harnad, 2001). For humans, the two options are learned sensorimotor grounding and innate grounding. (Although the latter is no doubt important, our current focus is more on the former.)

The need for information from outside the dictionary is formalized in Section 3. Apart from confirming the need for such external grounding, we take a symmetric stance: In natural language, some word meanings — especially highly abstract ones, such as those of mathematical or philosophical terms — are not or cannot be acquired through direct sensorimotor grounding. They are acquired through the *composition* of previously known words. The meaning of some of those words, or of the words in their respective definitions, must in turn have been grounded through direct sensorimotor experience.

To state this in another way: Meaning is not just formal definitions all the way down; nor is it just sensorimotor experience all the way up. The two extreme poles of that continuum are *sensorimotor induction* at one pole (trial and error experience with corrective feedback; observation, pointing, gestures, imitation, etc.), and *symbolic instruction* (definitions, descriptions, explanation, verbal examples etc.) at the other pole. Being able to identify from their lexicological structure which words were acquired one way or the other would provide us with important clues about the cognitive processes underlying language and the mental representation of meaning.

To compare the word meanings acquired via sensorimotor induction with word meanings acquired via symbolic instruction (definitions), we first need access to the encoding of that knowledge. In this component of our research, our hypothesis is that the representational structure of word meanings in dictionaries shares some commonalities with the representational structure of word meanings in the human brain (Hauk et al., 2008). We are thus trying to extract from dictionaries the grounding kernel (and eventually a minimum grounding set, which in general is a proper subset of this kernel), from which the rest of the dictionary can be reached through definitions alone. We hypothesize that this kernel, identified through formal structural analy-

sis, will exhibit properties that are also reflected in the mental lexicon. In parallel ongoing studies, we are finding that the words in the grounding kernel are indeed (1) more frequent in oral and written usage, (2) more concrete, (3) more readily imageable, and (4) learned earlier or at a younger age. We also expect they will be (5) more universal (across dictionaries, languages and cultures) (Chicoisne et al., 2008).

5 Grounding Kernels in Natural Language Dictionaries

In earlier research (Clark, 2003), we have been analyzing two special dictionaries: the Longman's Dictionary of Contemporary English (LDOCE) (Procter, 1978) and the Cambridge International Dictionary of English (CIDE) (Procter, 1995). Both are officially described as being based upon a *defining vocabulary*: a set of 2000 words which are purportedly the only words used in all the definitions of the dictionary, including the definitions of the defining vocabulary itself. A closer analysis of this defining vocabulary, however, has revealed that it is not always faithful to these constraints: A significant number of words used in the definitions turn out not to be in the defining vocabulary. Hence it became evident that we would ourselves have to generate a grounding kernel (roughly equivalent to the defining vocabulary) from these dictionaries.

The method presented in this paper makes it possible, given the graph structure of a dictionary, to extract a grounding kernel therefrom. Extracting this structure in turn confronts us with two further problems: *morphology* and *polysemy*. Neither of these problems has a definite algorithmic solution. Morphology can be treated through stemming and associated look-up lists for the simplest cases (*i.e.*, was → to be, and children → child), but more elaborate or complicated cases would require syntactic analysis or, ultimately, human evaluation. Polysemy is usually treated through statistical analysis of the word context (as in Latent Semantic Analysis) (Kintsch, 2007) or human evaluation. Indeed, a good deal of background knowledge is necessary to analyse an entry such as: "*dominant*: the fifth note of a musical scale of eight notes" (the LDOCE notes 16 different meanings of *scale* and 4 for *dominant*, and in our example, none of these words are used with their most frequent meaning).

Correct disambiguation of a dictionary is time-consuming work, as the most effective way to do it for now is through consensus among human evaluators. Fortunately, a fully disambiguated version of the WordNet database (Fellbaum, 1998; Fellbaum, 2005) has just become available. We expect the grounding kernel of WordNet to be of greater interest than the defining vocabulary of either CIDE or LDOCE (or what we extract from them and disambiguate automatically, and imperfectly) for our analysis.

6 Future Work

The main purpose of this paper was to introduce a formal approach to the symbol grounding problem based on the computational analysis of digital dictionaries. Ongoing and future work includes the following:

The minimum grounding set problem. We have seen that the problem of finding a minimum grounding set is NP-complete for general graphs. However, graphs associated with dictionaries have a very specific structure. We intend to describe a class of graphs including those specific graphs and to try to design a polynomial-time algorithm to solve the problem. Another approach is to design approximation algorithms, yielding a solution close to the optimal solution, with some known guarantee.

Grounding sets satisfying particular constraints. Let D be a dictionary, $G = (V, E)$ its associated graph, and $U \subseteq V$ any subset of vertices satisfying a given property P. We can use Algorithm 1 to test whether or not U is a grounding set. In particular, it would be interesting to test different sets U satisfying different cognitive constraints.

Relaxing the grounding conditions. In this paper we imposed strong conditions on the learning of new words: One must know all the words of the definition fully in order to learn a new word from them. This is not realistic, because we all know one can often understand a definition without knowing every single word in it. Hence one way to relax these conditions would be to modify the learning rule so that one need only understand at least $r\%$ of the definition, where r is some number between 0 and 100. Another variation would be to assign weights to words to take into account their morphosyntactic and semantic properties (rather than just treating them as an unordered list, as in the present analysis). Finally, we could consider "quasi-grounding sets", whose associated reachable set consists of $r\%$ of the whole dictionary.

Disambiguation of definitional relations. Analyzing real dictionaries raises, in its full generality, the problem of word and text disambiguation in free text; this is a very difficult problem. For example, if the word "make" appears in a definition, we do not know which of its many senses is intended — nor even what its grammatical category is. To our knowledge, the only available dictionary that endeavors to provide fully disambiguated definitions is the just-released version of WordNet. On the other hand, dictionary definitions have a very specific grammatical structure, presumably simpler and more limited than the general case of free text. It might hence be feasible to develop automatic disambiguation algorithms specifically dedicated to the special case of dictionary definitions.

Concluding Remark: Definition can reach the sense (sometimes), but only the senses can reach the referent.

Research funded by Canada Research Chair in Cognitive Sciences, SSHRC (S. Harnad)and NSERC (S. Harnad & O. Marcotte)

References

Barsalou, L. (2008) *Grounded Cognition*. Annual Review of Psychology (in press).

Bondy, J.A. & U.S.R. Murty. (1978) *Graph theory with applications*. Macmillan, New York.

Cangelosi, A. & Harnad, S. (2001) *The Adaptive Advantage of Symbolic Theft Over Sensorimotor Toil: Grounding Language in Perceptual Categories*. Evol. of Communication 4(1) 117-142.

Changizi, M.A. (2008) *Economically organized hierarchies in WordNet and the Oxford English Dictionary*. Cognitive Systems Research (in press).

Chicoisne G., A. Blondin-Massé, O. Picard, S. Harnad (2008) *Grounding Abstract Word Definitions In Prior Concrete Experience*. 6th Int. Conf. on the Mental Lexicon, Banff, Alberta.

Clark G. (2003) *Recursion Through Dictionary Definition Space: Concrete Versus Abstract Words*. (U. Southampton Tech Report).

Fellbaum, C. (1998) *WordNet: An electronic lexical database*. Cambridge: MIT Press.

Fellbaum, C. (2005) *Theories of human semantic representation of the mental lexicon*. In: Cruse, D. A. (Ed.), Handbook of Linguistics and Communication Science, Berlin, Germany: Walter de Gruyter, 1749-1758.

Frege G. (1948) *Sense and Reference*. The Philosophical Review 57 (3) 209-230.

Garey, M.R. & D.S. Johnson (1979) *Computers and Intractability: A Guide to the Theory of NP-completeness*. W.H. Freeman, New York.

Glenberg A.M. & D.A. Robertson (2002) *Symbol Grounding and Meaning: A Comparison of High-Dimensional and Embodied Theories of Meaning*. Journal of Memory and Language 43 (3) 379-401.

Harnad, S. (1990) *The Symbol Grounding Problem*. Physica D 42:335-346.

Harnad, S. (2003) *Symbol-Grounding Problem*. Encylopedia of Cognitive Science. Nature Publishing Group. Macmillan.

Harnad, S. (2005) *To Cognize is to Categorize: Cognition is Categorization*. In Lefebvre, C. and Cohen, H. (Eds.), Handbook of Categorization. Elsevier.

Hauk, O., M.H. Davis, F. Kherif, F. Pulvermüller. (2008) *Imagery or meaning? Evidence for a semantic origin of category-specific brain activity in metabolic imaging*. European Journal of Neuroscience 27 (7) 1856-1866.

Karp, R.M. (1972) *Reducibility among combinatorial problems*. In: R.E. Miller, J.W. Thatcher (Eds.), Complexity of Computer Computations, Plenum Press, New York, 1972, pp. 85-103.

Kintsch, W. (2007) *Meaning in Context*. In T.K. Landauer, D.S. McNamara, S. Dennis & W. Kintsch (Eds.), Handbook of Latent Semantic Analysis. Erlbaum.

Procter, P. (1978) *Longman Dictionary of Contemporary English*. Longman Group Ltd., Essex, UK.

Procter, P. (1995) *Cambridge International Dictionary of English (CIDE)*. Cambridge University Press.

Pylyshyn, Z. W. (1984) *Computation and Cognition: Towards a Foundation for Cognitive Science*. Cambridge: MIT Press.

Ravasz, E. & Barabasi, A. L. (2003) *Hierarchical organization in complex networks*. Physical Review E 67, 026112.

Rosen, K.H. (2007) *Discrete mathematics and its applications*, 6th ed. McGraw-Hill.

Steels, L. (2007) *The symbol grounding problem is solved, so what's next?* In De Vega, M. and G. Glenberg and A. Graesser (Eds.), Symbols, embodiment and meaning. Academic Press, North Haven.

Steyvers, M. & Tenenbaum J.B. (2005) *The large-scale structure of semantic networks: statistical analyses and a model of semantic growth*. Cognitive Science, 29(1) 41-78.

Tarjan, R. (1972) *Depth-first search and linear graph algorithms*. SIAM Journal on Computing. 1 (2) 146-160.

Encoding Tree Pair-based Graphs in Learning Algorithms: the Textual Entailment Recognition Case

Alessandro Moschitti
DISI, University of Trento
Via Sommarive 14
38100 POVO (TN) - Italy
moschitti@dit.unitn.it

Fabio Massimo Zanzotto
DISP, University of Rome "Tor Vergata"
Via del Politecnico 1
00133 Roma, Italy
zanzotto@info.uniroma2.it

Abstract

In this paper, we provide a statistical machine learning representation of textual entailment via syntactic graphs constituted by tree pairs. We show that the natural way of representing the syntactic relations between text and hypothesis consists in the huge feature space of all possible syntactic tree fragment pairs, which can only be managed using kernel methods. Experiments with Support Vector Machines and our new kernels for paired trees show the validity of our interpretation.

1 Introduction

Recently, a lot of valuable work on the recognition of textual entailment (RTE) has been carried out (Bar Haim et al., 2006). The aim is to detect implications between sentences like:

$T_1 \Rightarrow H_1$	
T_1	*"Wanadoo bought KStones"*
H_1	*"Wanadoo owns KStones"*

where T_1 and H_1 stand for text and hypothesis, respectively.

Several models, ranging from the simple lexical similarity between T and H to advanced Logic Form Representations, have been proposed (Corley and Mihalcea, 2005; Glickman and Dagan, 2004; de Salvo Braz et al., 2005; Bos and Markert, 2005). However, since a linguistic theory able to analytically show how to computationally solve the RTE problem has not been developed yet, to design accurate systems, we should rely upon the application of machine learning. In this perspective, TE training examples have to be represented in terms of statistical feature distributions. These typically consist in word sequences (along with their lexical similarity) and the syntactic structures of both text and hypothesis (e.g. their parse trees). The interesting aspect with respect to other natural language problems is that, in TE, features useful at describing an example are composed by pairs of features from Text and Hypothesis.

For example, using a word representation, a text and hypothesis pair, $\langle T, H \rangle$, can be represented by the sequences of words of the two sentences, i.e. $\langle t_1, .., t_n \rangle$ and $\langle h_1, .., h_m \rangle$, respectively. If we carry out a blind and complete statistical correlation analysis of the two sequences, the entailment property would be described by the set of subsequence pairs from T and H, i.e. the set $R = \{\langle s_t, s_h \rangle : s_t = \langle t_{i_1}, .., t_{i_l} \rangle, s_h = \langle h_{j_1}, .., h_{j_r} \rangle, l \leq n, r \leq m\}$. The relation set R constitutes a naive and complete representation of the example $\langle T, H \rangle$ in the feature space $\{\langle v, w \rangle : v, w \in V^*\}$, where V is the corpus vocabulary[1].

Although the above representation is correct and complete from a statistically point of view, it suffers from two practical drawbacks: (a) it is exponential in V and (b) it is subject to high degree of data sparseness which may prevent to carry out effective learning. The traditional solution for this problem relates to consider the syntactic structure of word sequences which provides their generalization.

The use of syntactic trees poses the problem of representing structures in learning algorithms.

[1] V^* is larger than the actual space, which is the one of all possible subsequences with gaps, i.e. it only contains all possible concatenations of words respecting their order.

Coling 2008: Proceedings of 3rd Textgraphs workshop on Graph-Based Algorithms in Natural Language Processing, pages 25–32
Manchester, August 2008

For this purpose, kernel methods, and in particular tree kernels allow for representing trees in terms of all possible subtrees (Collins and Duffy, 2002). Unfortunately, the representation in entailment recognition problems requires the definition of kernels over graphs constituted by tree pairs, which are in general different from kernels applied to single trees. In (Zanzotto and Moschitti, 2006), this has been addressed by introducing semantic links (placeholders) between text and hypothesis parse trees and evaluating two distinct tree kernels for the trees of texts and for those of hypotheses. In order to make such disjoint kernel combination effective, all possible assignments between the placeholders of the first and the second entailment pair were generated causing a remarkable slowdown.

In this paper, we describe the feature space of all possible tree fragment pairs and we show that it can be evaluated with a much simpler kernel than the one used in previous work, both in terms of design and computational complexity. Moreover, the experiments on the RTE datasets show that our proposed kernel provides higher accuracy than the simple union of tree kernel spaces.

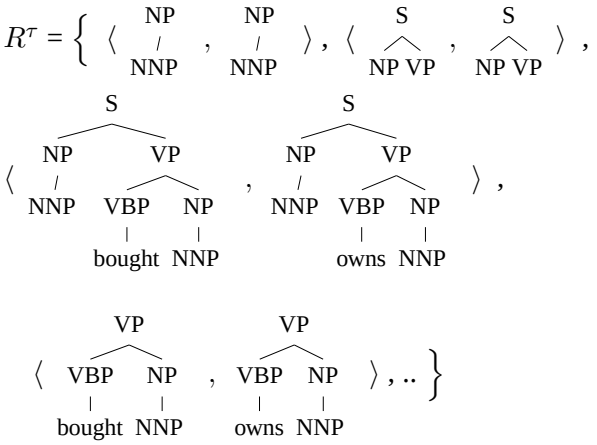
2 Fragments of Tree Pair-based Graphs

The previous section has pointed out that RTE can be seen as a relational problem between word sequences of Text and Hypothesis. The syntactic structures embedded in such sequences can be generalized by natural language grammars. Such generalization is very important since it is evident that entailment cases depend on the syntactic structures of Text and Hypothesis. More specifically, the set R described in the previous section can be extended and generalized by considering syntactic derivations[2] that generate word sequences in the training examples. This corresponds to the following set of tree fragment pairs:

$$R^{\tau} = \{\langle \tau_t, \tau_h \rangle : \tau_t \in \mathcal{F}(T), \tau_h \in \mathcal{F}(H)\}, \quad (1)$$

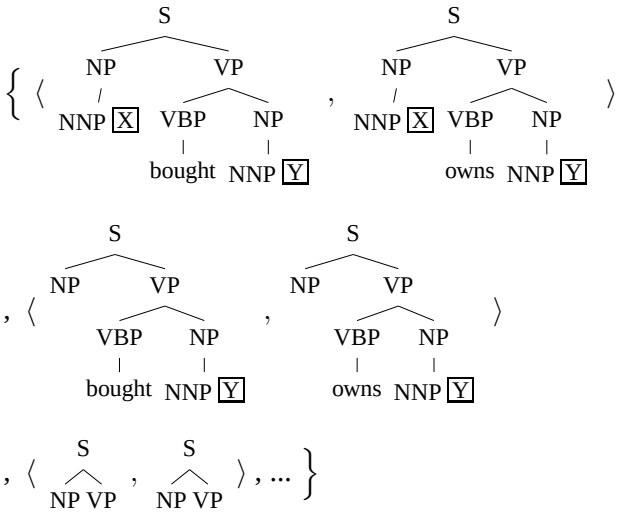
where $\mathcal{F}(\cdot)$ indicates the set of tree fragments of a parse tree (i.e. the one of the text T or of the hypothesis H). R^{τ} contains less sparse relations than R. For instance, given T_1 and H_1 of the previous section, we would have the following relational description:

[2] By cutting derivation at different depth, different degrees of generalization can be obtained.



These features (relational pairs) generalize the entailment property, e.g. the pair ⟨[VP [VBP bought] [NP]], [VP [VBP own] [NP]]⟩ generalizes many word sequences, i.e. those external to the verbal phrases and internal to the NPs.

We can improve this space by adding semantic links between the tree fragments. Such links or placeholders have been firstly proposed in (Zanzotto and Moschitti, 2006). A placeholder assigned to a node of τ_t and a node of τ_h states that such nodes dominate the same (or similar) information. In particular, placeholders are assigned to nodes whose words t_i in T are equal, similar, or semantically dependent on words h_j in H. Using placeholders, we obtain a richer fragment pair based representation that we call $R^{\tau p}$, exemplified hereafter:



The placeholders (or variables) indicated with X and Y specify that the NNPs labeled by the same variables dominate similar or identical words. Therefore, an automatic algorithm that assigns placeholders to semantically similar constituents is needed. Moreover, although $R^{\tau p}$ contains more semantic and less sparse features than

both R^τ and R, its cardinality is still exponential in the number of the words of T and H. This means that standard machine learning algorithms cannot be applied. In contrast, tree kernels (Collins and Duffy, 2002) can be used to efficiently generate the huge space of tree fragments but, to generate the space of pairs of tree fragments, a new kernel function has to be defined.

The next section provides a solution to both problems. i.e. an algorithm for placeholders assignments and for the computation of paired tree kernels which generates R^τ and $R^{\tau p}$ representations.

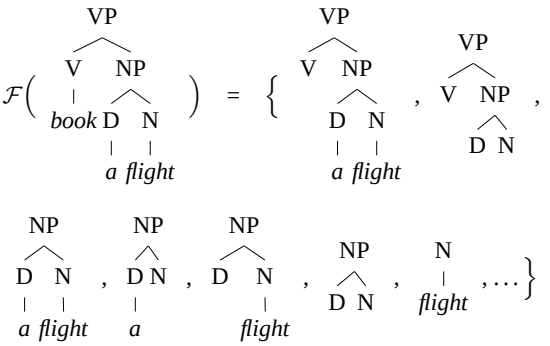


Figure 1: A syntactic parse tree.

3 Kernels over Semantic Tree Pair-based Graphs

The previous section has shown that placeholders enrich a tree-based graph with relational information, which, in turn, can be captured by means of word semantic similarities $sim_w(w_t, w_h)$, e.g. (Corley and Mihalcea, 2005; Glickman et al., 2005). More specifically, we use a two-step greedy algorithm to anchor the content words (verbs, nouns, adjectives, and adverbs) in the hypothesis W_H to words in the text W_T.

In the first step, each word w_h in W_H is connected to all words w_t in W_T that have the maximum similarity $sim_w(w_t, w_h)$ with it (more than one w_t can have the maximum similarity with w_h). As result, we have a set of anchors $A \subset W_T \times W_H$. $sim_w(w_t, w_h)$ is computed by means of three techniques:

1. Two words are maximally similar if they have the same surface form $w_t = w_h$.

2. Otherwise, WordNet (Miller, 1995) similarities (as in (Corley and Mihalcea, 2005)) and different relation between words such as verb entailment and derivational morphology are applied.

3. The edit distance measure is finally used to capture the similarity between words that are missed by the previous analysis (for misspelling errors or for the lack of derivational forms in WordNet).

In the second step, we select the final anchor set $A' \subseteq A$, such that $\forall w_t$ (or w_h) $\exists! \langle w_t, w_h \rangle \in A'$. The selection is based on a simple greedy algorithm that given two pairs $\langle w_t, w_h \rangle$ and $\langle w'_t, w_h \rangle$ to be selected and a pair $\langle s_t, s_h \rangle$ already selected, considers word proximity (in terms of number of words) between w_t and s_t and between w'_t and s_t; the nearest word will be chosen.

Once the graph has been enriched with semantic information we need to represent it in the learning algorithm; for this purpose, an interesting approach is based on kernel methods. Since the considered graphs are composed by only two trees, we can carried out a simplified computation of a graph kernel based on tree kernel pairs.

3.1 Tree Kernels

Tree Kernels (e.g. see NLP applications in (Giuglea and Moschitti, 2006; Zanzotto and Moschitti, 2006; Moschitti et al., 2007; Moschitti et al., 2006; Moschitti and Bejan, 2004)) represent trees in terms of their substructures (fragments) which are mapped into feature vector spaces, *e.g.* $\Re^n$. The kernel function measures the similarity between two trees by counting the number of their common fragments. For example, Figure 1 shows some substructures for the parse tree of the sentence `"book a flight"`. The main advantage of tree kernels is that, to compute the substructures shared by two trees τ_1 and τ_2, the whole fragment space is not used. In the following, we report the formal definition presented in (Collins and Duffy, 2002).

Given the set of fragments $\{f_1, f_2, ..\} = \mathcal{F}$, the indicator function $I_i(n)$ is equal 1 if the target f_i is rooted at node n and 0 otherwise. A tree kernel is then defined as:

$$TK(\tau_1, \tau_2) = \sum_{n_1 \in N_{\tau_1}} \sum_{n_2 \in N_{\tau_2}} \Delta(n_1, n_2) \qquad (2)$$

where N_{τ_1} and N_{τ_2} are the sets of the τ_1's and τ_2's

nodes, respectively and

$$\Delta(n_1, n_2) = \sum_{i=1}^{|\mathcal{F}|} I_i(n_1) I_i(n_2)$$

The latter is equal to the number of common fragments rooted in the n_1 and n_2 nodes and Δ can be evaluated with the following algorithm:

1. if the productions at n_1 and n_2 are different then $\Delta(n_1, n_2) = 0$;
2. if the productions at n_1 and n_2 are the same, and n_1 and n_2 have only leaf children (i.e. they are pre-terminals symbols) then $\Delta(n_1, n_2) = 1$;
3. if the productions at n_1 and n_2 are the same, and n_1 and n_2 are not pre-terminals then

$$\Delta(n_1, n_2) = \prod_{j=1}^{nc(n_1)} (1 + \Delta(c^j_{n_1}, c^j_{n_2})) \quad (3)$$

where $nc(n_1)$ is the number of the children of n_1 and c^j_n is the j-th child of the node n. Note that since the productions are the same, $nc(n_1) = nc(n_2)$.

Additionally, we add the decay factor λ by modifying steps (2) and (3) as follows[3]:

2. $\Delta(n_1, n_2) = \lambda$,
3. $\Delta(n_1, n_2) = \lambda \prod_{j=1}^{nc(n_1)} (1 + \Delta(c^j_{n_1}, c^j_{n_2}))$.

The computational complexity of Eq. 2 is $O(|N_{\tau_1}| \times |N_{\tau_2}|)$ although the average running time tends to be linear (Moschitti, 2006).

3.2 Tree-based Graph Kernels

The above tree kernel function can be applied to the parse trees of two texts or those of the two hypotheses to measure their similarity in terms of the shared fragments. If we sum the contributions of the two kernels (for texts and for hypotheses) as proposed in (Zanzotto and Moschitti, 2006), we just obtain the feature space of the union of the fragments which is completely different from the space of the tree fragments pairs, i.e. R^{τ}. Note that the union space is not useful to describe which grammatical and lexical property is at the same time held by T and H to trig the implication.

Therefore to generate the space of the fragment pairs we need to define the kernel between two pairs of entailment examples $\langle T_1, H_1 \rangle$ and $\langle T_2, H_2 \rangle$ as

$$K_p(\langle T_1, H_1 \rangle, \langle T_2, H_2 \rangle) = \\ = \sum_{n_1 \in T_1} \sum_{n_2 \in T_2} \sum_{n_3 \in H_1} \sum_{n_4 \in H_2} \Delta(n_1, n_2, n_3, n_4),$$

where Δ evaluates the number of subtrees rooted in n_1 and n_2 combined with those rooted in n_3 and n_4. More specifically, each fragment rooted into the nodes of the two texts' trees is combined with each fragment rooted in the two hypotheses' trees. Now, since the number of subtrees rooted in the texts is independent of the number of trees rooted in the hypotheses,

$$\Delta(n_1, n_2, n_3, n_4) = \Delta(n_1, n_2)\Delta(n_3, n_4).$$

Therefore, we can rewrite K_p as:

$$\begin{aligned} &K_p(\langle T_1, H_1 \rangle, \langle T_2, H_2 \rangle) = \\ &= \sum_{n_1 \in T_1} \sum_{n_2 \in T_2} \sum_{n_3 \in H_1} \sum_{n_4 \in H_2} \Delta(n_1, n_2)\Delta(n_3, n_4) = \\ &= \sum_{n_1 \in T_1} \sum_{n_2 \in T_2} \Delta(n_1, n_2) \sum_{n_3 \in H_1} \sum_{n_4 \in H_2} \Delta(n_3, n_4) = \\ &= K_t(T_1, T_2) \times K_t(H_1, H_2). \end{aligned} \quad (4)$$

This result shows that the natural kernel to represent textual entailment sentences is the kernel product, which corresponds to the set of all possible syntactic fragment pairs. Note that, such kernel can be also used to evaluate the space of fragment pairs for trees enriched with relational information, i.e. by placeholders.

4 Approximated Graph Kernel

The feature space described in the previous section correctly encodes the fragment pairs. However, such huge space may result inadequate also for algorithms such as SVMs, which are in general robust to many irrelevant features. An approximation of the fragment pair space is given by the kernel described in (Zanzotto and Moschitti, 2006). Hereafter we illustrate its main points.

First, tree kernels applied to two texts or two hypotheses match identical fragments. When placeholders are added to trees, the labeled fragments

[3] To have a similarity score between 0 and 1, we also apply the normalization in the kernel space, i.e. $K'(\tau_1, \tau_2) = \frac{TK(\tau_1, \tau_2)}{\sqrt{TK(\tau_1, \tau_1) \times TK(\tau_2, \tau_2)}}$.

are matched only if the basic fragments and the assigned placeholders match. This means that we should use the same placeholders for all texts and all hypotheses of the corpus. Moreover, they should be assigned in a way that similar syntactic structures and similar relational information between two entailment pairs can be matched, i.e. same placeholders should be assigned to the potentially similar fragments.

Second, the above task cannot be carried out at pre-processing time, i.e. when placeholders are assigned to trees. At the running time, instead, we can look at the comparing trees and make a more consistent decision on the type and order of placeholders. Although, there may be several approaches to accomplish this task, we apply a basic heuristic which is very intuitive:

Choose the placeholder assignment that maximizes the tree kernel function over all possible correspondences

More formally, let A and A' be the placeholder sets of $\langle T, H\rangle$ and $\langle T', H'\rangle$, respectively, without loss of generality, we consider $|A| \geq |A'|$ and we align a subset of A to A'. The best alignment is the one that maximizes the syntactic and lexical overlapping of the two subtrees induced by the aligned set of anchors. By calling C the set of all bijective mappings from $S \subseteq A$, with $|S| = |A'|$, to A', an element $c \in C$ is a substitution function. We define the best alignment c_{max} the one determined by

$$c_{max} = argmax_{c\in C}(TK(t(T,c),t(T',i))+ TK(t(H,c),t(H',i)),$$

where (1) $t(\cdot, c)$ returns the syntactic tree enriched with placeholders replaced by means of the substitution c, (2) i is the identity substitution and (3) $TK(\tau_1, \tau_2)$ is a tree kernel function (e.g. the one specified by Eq. 2) applied to the two trees τ_1 and τ_2.

At the same time, the desired similarity value to be used in the learning algorithm is given by the kernel sum: $TK(t(T, c_{max}), t(T', i)) + TK(t(H, c_{max}), t(H', i))$, i.e. by solving the following optimization problem:

$$K_s(\langle T, H\rangle, \langle T', H'\rangle) = max_{c\in C}(TK(t(T,c),t(T',i))+ TK(t(H,c),t(H',i)), \quad (5)$$

For example, let us compare the following two pairs (T_1, H_1) and (T_2, H_2) in Fig. 2.

To assign the placeholders 1, 2 and 3 of (T_2, H_2) to those of (T_1, H_1), i.e. X and Y, we need to maximize the similarity between the two texts' trees and between the two hypotheses' trees. It is straightforward to derive that X=1 and Y=3 allow more substructures (i.e. large part of the trees) to be identical, e.g. [S [NP 1 X VP]] , [VP [VBP NP 3 Y]], [S [NP 1 X VP [VBP NP 3 Y]]].

Finally, it should be noted that, (a) $K_s(\langle T, H\rangle, \langle T', H'\rangle)$ is a symmetric function since the set of derivation C are always computed with respect to the pair that has the largest anchor set and (b) it is not a valid kernel as the max function does not in general produce valid kernels. However, in (Haasdonk, 2005), it is shown that when kernel functions are not positive semidefinite like in this case, SVMs still solve a data separation problem in pseudo Euclidean spaces. The drawback is that the solution may be only a local optimum. Nevertheless, such solution can still be valuable as the problem is modeled with a very rich feature space.

Regarding the computational complexity, running the above kernel on a large training set may result very expensive. To overcome this drawback, in (Moschitti and Zanzotto, 2007), it has been designed an algorithm to factorize the evaluation of tree subparts with respect to the different substitution. The resulting speed-up makes the application of such kernel feasible for datasets of ten of thousands of instances.

5 Experiments

The aim of the experiments is to show that the space of tree fragment pairs is the most effective to represent Tree Pair-based Graphs for the design of Textual Entailment classifiers.

5.1 Experimental Setup

To compare our model with previous work we implemented the following kernels in SVM-light (Joachims, 1999):

- $K_s(e_1, e_2) = K_t(T_1, T_2) + K_t(H_1, H_2)$, where $e_1 = \langle T_1, H_1\rangle$ and $e_2 = \langle T_2, H_2\rangle$ are two text and hypothesis pairs and K_t is the syntactic tree kernel (Collins and Duffy, 2002) presented in the previous section.
- $K_p(e_1, e_2) = K_t(T_1, T_2) \times K_t(H_1, H_2)$, which (as shown in the previous sections) en-

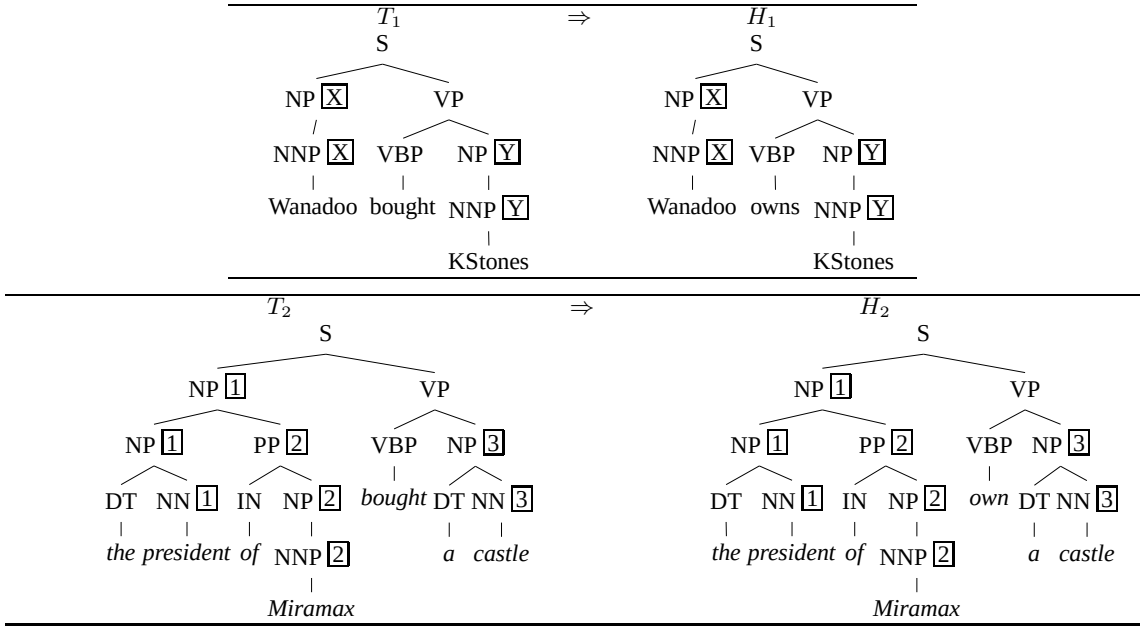


Figure 2: The problem of finding the correct mapping between placeholders

codes the tree fragment pairs with and without placeholders.

- $K_{max}(e_1, e_2) = \max_{c \in \mathcal{C}} \big(K_t(\phi_c(T_1), \phi_c(T_2)) + K_t(\phi_c(H_1), \phi_c(H_2))\big)$, where c is a possible placeholder assignment which connects nodes from the first pair with those of the second pair and $\phi_c(\cdot)$ transforms trees according to c.

- $K_{pmx}(e_1, e_2) = \max_{c \in \mathcal{C}} \big(K_t(\phi_c(T_1), \phi_c(T_2)) \times K_t(\phi_c(H_1), \phi_c(H_2))\big)$.

Note that K_{max} is the kernel proposed in (Zanzotto and Moschitti, 2006) and K_{pmx} is a hybrid kernel based on the maximum K_p, which uses the space of tree fragment pairs. For all the above kernels, we set the default cost factor and trade-off parameters and we set λ to 0.4.

To experiment with entailment relations, we used the data sets made available by the first (Dagan et al., 2005) and second (Bar Haim et al., 2006) Recognizing Textual Entailment Challenge. These corpora are divided in the development sets $D1$ and $D2$ and the test sets $T1$ and $T2$. $D1$ contains 567 examples whereas $T1$, $D2$ and $T2$ all have the same size, i.e. 800 instances. Each example is an ordered pair of texts for which the entailment relation has to be decided.

5.2 Evaluation and Discussion

Table 1 shows the results of the above kernels on the split used for the RTE competitions. The first column reports the kernel model. The second and third columns illustrate the model accuracy for RTE1 whereas column 4 and 5 show the accuracy for RTE2. Moreover, $\neg$ P indicates the use of standard syntactic trees and P the use of trees enriched with placeholders. We note that:

First, the space of tree fragment pairs, generated by K_p improves the one generated by K_s (i.e. the simple union of the fragments of texts and hypotheses) of 4 (58.9% vs 54.9%) and 0.9 (53.5% vs 52.6%) points on RTE1 and RTE2, respectively. This suggests that the fragment pairs are more effective for encoding the syntactic rules describing the entailment concept.

Second, on RTE1, the introduction of placeholders does not improve K_p or K_s suggesting that for their correct exploitation an extension of the space of tree fragment pairs should be modeled.

Third, on RTE2, the impact of placeholders seems more important but only K_{max} and K_s are able to fully exploit their semantic contribution. A possible explanation is that in order to use the set of all possible assignments (required by K_{max}), we needed to prune the "too large" syntactic trees as also suggested in (Zanzotto and Moschitti, 2006). This may have negatively biased the statistical distribution of tree fragment pairs.

Finally, although we show that K_p is better

Kernels	RTE1		RTE2	
	¬ P	P	¬ P	P
K_s	54.9	50.0	52.6	59.5
K_p	**58.9**	55.5	**53.5**	56.0
$K_{\max}$	-	58.25	-	61.0
K_{pmx}	-	50.0	-	56.8

Table 1: Accuracy of different kernel models using (P) and not using (¬ P) placeholder information on RTE1 and RTE2.

suited for RTE than the other kernels, its accuracy is lower than the state-of-the-art in RTE. This is because the latter uses additional models like the lexical similarity between text and hypothesis, which greatly improve accuracy.

6 Conclusion

In this paper, we have provided a statistical machine learning representation of textual entailment via syntactic graphs constituted by tree pairs. We have analytically shown that the natural way of representing the syntactic relations between text and hypothesis in learning algorithms consists in the huge feature space of all possible syntactic tree fragment pairs, which can only be managed using kernel methods.

Therefore, we used tree kernels, which allow for representing trees in terms of all possible subtrees. More specifically, we defined a new model for the entailment recognition problems, which requires the definition of kernels over graphs constituted by tree pairs. These are in general different from kernels applied to single trees. We also studied another alternative solution which concerns the use of semantic links (placeholders) between text and hypothesis parse trees (to form relevant semantic fragment pairs) and the evaluation of two distinct tree kernels for the trees of texts and for those of hypotheses. In order to make such disjoint kernel combination effective, all possible assignments between the placeholders of the first and the second entailment pair have to be generated (causing a remarkable slowdown).

Our experiments on the RTE datasets show that our proposed kernel may provide higher accuracy than the simple union of tree kernel spaces with a much simpler and faster algorithm. Future work will be devoted to make the tree fragment pair space more effective, e.g. by using smaller and accurate tree representation for text and hypothesis.

Acknowledgments

We would like to thank the anonymous reviewers for their professional and competent reviews and for their invaluable suggestions.
Alessandro Moschitti would like to thank the European Union project, LUNA (spoken **L**anguage **UN**derstanding in multilingu**A**l communication systems) contract n 33549 for supporting part of his research.

References

Bar Haim, Roy, Ido Dagan, Bill Dolan, Lisa Ferro, Danilo Giampiccolo, Bernardo Magnini, and Idan Szpektor. 2006. The II PASCAL RTE challenge. In *PASCAL Challenges Workshop*, Venice, Italy.

Bos, Johan and Katja Markert. 2005. Recognising textual entailment with logical inference. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 628–635, Vancouver, British Columbia, Canada, October. Association for Computational Linguistics.

Collins, Michael and Nigel Duffy. 2002. New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In *Proceedings of ACL02*.

Corley, Courtney and Rada Mihalcea. 2005. Measuring the semantic similarity of texts. In *Proc. of the ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailment*, pages 13–18, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Dagan, Ido, Oren Glickman, and Bernardo Magnini. 2005. The PASCAL RTE challenge. In *PASCAL Challenges Workshop*, Southampton, U.K.

de Salvo Braz, R., R. Girju, V. Punyakanok, D. Roth, and M. Sammons. 2005. An inference model for semantic entailment in natural language. In *Proceedings of AAAI*, pages 1678–1679.

Giuglea, Ana-Maria and Alessandro Moschitti. 2006. Semantic role labeling via framenet, verbnet and propbank. In *Proceedings of Coling-ACL*, Sydney, Australia.

Glickman, Oren and Ido Dagan. 2004. Probabilistic textual entailment: Generic applied modeling of language variability. In *Proceedings of the Workshop on Learning Methods for Text Understanding and Mining*, Grenoble, France.

Glickman, Oren, Ido Dagan, and Moshe Koppel. 2005. Web based probabilistic textual entailment. In *Proceedings of the 1st Pascal Challenge Workshop*, Southampton, UK.

Haasdonk, Bernard. 2005. Feature space interpretation of SVMs with indefinite kernels. *IEEE Trans Pattern Anal Mach Intell*, 27(4):482–92, Apr.

Joachims, Thorsten. 1999. Making large-scale svm learning practical. In Schlkopf, B., C. Burges, and A. Smola, editors, *Advances in Kernel Methods-Support Vector Learning*. MIT Press.

Miller, George A. 1995. WordNet: A lexical database for English. *Communications of the ACM*, 38(11):39–41, November.

Moschitti, Alessandro and Cosmin Adrian Bejan. 2004. A semantic kernel for predicate argument classification. In *CoNLL-2004*, USA.

Moschitti, A. and F. Zanzotto. 2007. Fast and effective kernels for relational learning from texts. In Ghahramani, Zoubin, editor, *Proceedings of the 24th Annual International Conference on Machine Learning (ICML 2007)*.

Moschitti, Alessandro, Daniele Pighin, and Roberto Basili. 2006. Semantic Role Labeling via Tree Kernel Joint Inference. In *Proceedings of CoNLL-X*.

Moschitti, Alessandro, Silvia Quarteroni, Roberto Basili, and Suresh Manandhar. 2007. Exploiting syntactic and shallow semantic kernels for question answer classification. In *Proceedings ACL*, Prague, Czech Republic.

Moschitti, Alessandro. 2006. Efficient convolution kernels for dependency and constituent syntactic trees. In *ECML'06*.

Zanzotto, Fabio Massimo and Alessandro Moschitti. 2006. Automatic learning of textual entailments with cross-pair similarities. In *Proceedings of the 21st Coling and 44th ACL*, pages 401–408, Sydney, Australia, July.

Graph-based Clustering for Semantic Classification of Onomatopoetic Words

Kenichi Ichioka
Interdisciplinary Graduate School of Medicine and Engineering
University of Yamanashi, Japan
g07mk001@yamanashi.ac.jp

Fumiyo Fukumoto
Interdisciplinary Graduate School of Medicine and Engineering
University of Yamanashi, Japan
fukumoto@yamanashi.ac.jp

Abstract

This paper presents a method for semantic classification of onomatopoetic words like "ひゅーひゅー (*hum*)" and "からんころん (*clip clop*)" which exist in every language, especially Japanese being rich in onomatopoetic words. We used a graph-based clustering algorithm called $Newman$ clustering. The algorithm calculates a simple quality function to test whether a particular division is meaningful. The quality function is calculated based on the weights of edges between nodes. We combined two different similarity measures, distributional similarity, and orthographic similarity to calculate weights. The results obtained by using the Web data showed a 9.0% improvement over the baseline single distributional similarity measure.

1 Introduction

Onomatopoeia which we call *onomatopoetic* word (*ono* word) is the formation of words whose sound is imitative of the sound of the noise or action designated, such as 'hiss' (McLeod, 1991). It is one of the linguistic features of Japanese. Consider two sentences from Japanese.

(1) 私は廊下のスリッパの音で起こされたので、とても眠い。
"I'm too sleepy because I awoke to the slippers in the hall."

(2) 私は廊下をぱたぱた走るスリッパの音で起こされたので、とても眠い。
"I'm too sleepy because I awoke to the *pit-a-pat* of slippers in the hall."

Sentences (1) and (2) are almost the same sense. However, sentence (2) which includes *ono* word, "ぱたぱた (*pit-a-pat*)" is much better to make the scene alive, or represents an image clearly. Therefore large-scale semantic resource of *ono* words is indispensable for not only NLP, but also many semantic-oriented applications such as Question Answering, Paraphrasing, and MT systems. Although several machine-readable dictionaries which are fine-grained and large-scale semantic knowledge like WordNet, COMLEX, and EDR dictionary exist, there are none or few onomatopoetic thesaurus. Because (i) it is easy to understand its sense of *ono* word for Japanese, and (ii) it is a fast-changing linguistic expressions, as it is a vogue word. Therefore, considering this resource scarcity problem, semantic classification of *ono* words which do not appear in the resource but appear in corpora is very important.

In this paper, we focus on Japanese onomatopoetic words, and propose a method for classifying them into a set with similar meaning. We used the Web as a corpus to collect *ono* words, as they appear in different genres of dialogues including broadcast news, novels and comics, rather than a well-edited, balanced corpus like newspaper articles. The problem using a large, heterogeneous collection of Web data is that the Web counts are far more noisy than counts obtained from textual corpus. We thus used a graph-based clustering algorithm, called $Newman$ clustering for classifying *ono* words. The algorithm does not simply calculate the number of shortest paths between pairs of nodes, but instead calculates a quality function

Coling 2008: Proceedings of 3rd Textgraphs workshop on Graph-Based Algorithms in Natural Language Processing, pages 33–40
Manchester, August 2008

of how good a cluster structure found by an algorithm is, and thus makes the computation far more efficient. The efficacy of the algorithm depends on a quality function which is calculated by using the weights of edges between nodes. We combined two different similarity measures, and used them to calculate weights. One is co-occurrence based distributional similarity measure. We tested mutual information (MI) and a χ^2 statistic as a similarity measure. Another is orthographic similarity which is based on a feature of *ono* words called "sound symbolism". Sound symbolism indicates that phonemes or phonetic sequences express their senses. As *ono* words imitate the sounds associated with the objects or actions they refer to, their phonetic sequences provide semantic clues for classification. The empirical results are encouraging, and showed a 9.0% improvement over the baseline single distributional similarity measure.

2 Previous Work

There are quite a lot of work on semantic classification of words with corpus-based approach. The earliest work in this direction are those of (Hindle, 1990), (Lin, 1998), (Dagan et al., 1999), (Chen and Chen, 2000), (Geffet and Dagan, 2004) and (Weeds and Weir, 2005). They used distributional similarity. Similarity measures based on distributional hypothesis compare a pair of weighted feature vectors that characterize two words. Features typically correspond to other words that co-occur with the characterized word in the same context. Lin (1998) proposed a word similarity measure based on the distributional pattern of words which allows to construct a thesaurus using a parsed corpus. He compared the result of automatically created thesaurus with WordNet and Roget, and reported that the result was significantly closer to WordNet than Roget Thesaurus was.

Graph representations for word similarity have also been proposed by several researchers (Jannink and Wiederhold, 1999; Galley and McKeown, 2003; Muller et al., 2006). Sinha and Mihalcea (2007) proposed a graph-based algorithm for unsupervised word sense disambiguation which combines several semantic similarity measures including Resnik's metric (Resnik, 1995), and algorithms for graph centrality. They reported that the results using the SENSEVAL-2 and SENSEVAL-3 English all-words data sets lead to relative error rate reductions of $5-8\%$ as compared to the previous work (Mihalcea, 2005).

In the context of graph-based clustering of words, Widdows and Dorow (2002) used a graph model for unsupervised lexical acquisition. The graph structure is built by linking pairs of words which participate in particular syntactic relationships. An incremental cluster-building algorithm using the graph structure achieved 82% accuracy at a lexical acquisition task, evaluated against WordNet 10 classes, and each class consists of 20 words. Matsuo *et al.* (2006) proposed a method of word clustering based on a word similarity measure by Web counts. They used Newman clustering for clustering algorithm. They evaluated their method using two sets of word classes. One is derived from the Web data, and another is from WordNet.[1] Each set consists of 90 noun words. They reported that the results obtained by Newman clustering were better than those obtained by average-link agglomerative clustering. Our work is similar to their method in the use of Newman clustering. However, they classified Japanese noun words, while our work is the first to aim at detecting semantic classification of onomatopoetic words. Moreover, they used only a single similarity metric, co-occurrence based similarity, while Japanese, especially "kanji" characters of noun words provide semantic clues for classifying words.

3 System Description

The method consists of three steps: retrieving co-occurrences using the Web, calculating similarity between *ono* words, and classifying *ono* words by using Newman clustering.

3.1 Retrieving Co-occurrence using the Web

One criterion for calculating semantic similarity between *ono* words is co-occurrence based similarity. We retrieved frequency of two *ono* words occurring together by using the Web search engine, Google. The similarity between them is calculated based on their co-occurrence frequency. Like much previous work on semantic classification of the lexicons, our assumption is that semantically similar words appear in similar contexts. A lot of strategies for searching words are provided in Google. Of these we focused on two methods: Boolean search AND and $phrase$-based search.

[1] They used WordNet hypernym information. It consists of 10 classes. They assigned 90 Japanese noun words to each class.

When we use AND boolean search, *i.e.*, (O_i O_j) where O_i and O_j are *ono* words, we can retrieve the number of documents which include both O_i and O_j. In contrast, $phrase$-based search, *i.e.*, ("O_i O_j") retrieves documents which include two adjacent words O_i and O_j.

3.2 Similarity Measures

The second step is to calculate semantic similarity between *ono* words. We combined two different similarity measures: the co-occurrence frequency based similarity and orthographic similarity measures.

3.2.1 Co-occurrence based Similarity Measure

We focused on two popular measures: the mutual information (MI) and χ^2 statistics.

1. Mutual Information

Church and Hanks (1990) discussed the use of the mutual information statistics as a way to identify a variety of interesting linguistic phenomena, ranging from semantic relations of the doctor/nurse type (content word/content word) to lexico-syntactic co-occurrence preferences between verbs and prepositions (content word/function word). Let O_i and O_j be *ono* words retrieved from the Web. The mutual information $MI(O_i, O_j)$ is defined as:

$$MI(O_i, O_j) = \log \frac{S_{all} \times f(O_i, O_j)}{S_{O_i} \times S_{O_j}}, \quad (1)$$

$$\text{where } S_{O_i} = \sum_{k \in O_{all}} f(O_i, O_k), \quad (2)$$

$$S_{all} = \sum_{O_i \in O_{all}} S_{O_i}. \quad (3)$$

In Eq. (1), $f(O_i, O_j)$ refers to the frequency of O_i and O_j occurring together, and O_{all} is a set of all *ono* words retrieved from the Web.

2. χ^2 statistic

The $\chi^2(O_i, O_j)$ is defined as:

$$\chi^2(O_i, O_j) = \frac{f(O_i, O_j) - E(O_i, O_j)}{E(O_i, O_j)}, \quad (4)$$

$$\text{where } E(O_i, O_j) = S_{O_i} \times \frac{S_{O_j}}{S_{all}}. \quad (5)$$

S_{O_i} and S_{all} in Eq. (5) refer to Eq. (2) and (3), respectively. A major difference between χ^2 and MI is that the former is a normalized value.

3.2.2 Orthographic Similarity Measure

Orthographic similarity has been widely used in spell checking and speech recognition systems (Damerau, 1964). Our orthographic similarity measure is based on a unit of phonetic sequence. The key steps of the similarity between two *ono* words is defined as:

1. Convert each *ono* word into phonetic sequences.

   The "hiragana" characters of *ono* word are converted into phonetic sequences by a unique rule. Basically, there are 19 consonants and 5 vowels, as listed in Table 1.

Table 1: Japanese consonants and vowels

Consonant	–, N, Q, h, hy, k, ky, m, my, n, ny, r, ry, s, sy, t, ty, w, y
Vowel	a, i, u, e, o

   Consider phonetic sequences "hyu-hyu-" of *ono* word "ひゅーひゅー" (*hum*). It is segmented into 4 consonants "hy", "-", "hy" and "-", and two vowels, "u" and "u".

2. Form a vector in n-dimensional space.

   Each *ono* word is represented as a vector of consonants(vowels), where each dimension of the vector corresponds to each consonant and vowel, and each value of the dimension is frequencies of its corresponding consonant(vowel).

3. Calculate orthographic similarity.

   The orthographic similarity between *ono* words, O_i and O_j is calculated based on the consonant and vowel distributions. We used two popular measures, *i.e.*, the cosine similarity, and α-skew divergence. The cosine measures the similarity of the two vectors by calculating the cosine of the angle between vectors. α-skew divergence is defined as:

$$\alpha div(x, y) = D(y \mid\mid \alpha \cdot x + (1 - \alpha) \cdot y),$$

   where $D(x||y)$ refers to Kullback-Leibler and defined as:

$$D(x||y) = \sum_{i=1}^{n} x_i * \log \frac{x_i}{y_i}. \quad (6)$$

Lee (1999) reported the best results with α = 0.9. We used the same value. We defined a similarity metric by combining co-occurrence based and orthographic similarity measures[2]:

$$Sim(O_i, O_j) = MI(O_i, O_j) \times (Cos(O_i, O_j) + 1) \quad (7)$$

3.3 The Newman Clustering Algorithm

We classified *ono* words collected from the WWW. Therefore, the clustering algorithm should be efficient and effective even in the very high dimensional spaces. For this purpose, we chose a graph-based clustering algorithm, called $Newman$ clustering. The Newman clustering is a hierarchical clustering algorithm which is based on Network structure (Newman, 2004). The network structure consists of nodes within which the node-node connections are edges. It produces some division of the nodes into communities, regardless of whether the network structure has any natural such division. Here, "community" or "cluster" have in common that they are groups of densely interconnected nodes that are only sparsely connected with the rest of the network. To test whether a particular division is meaningful a quality function Q is defined:

$$Q = \sum_i (e_{ii} - a_i^2)$$

where e_{ij} is the sum of the weight of edges between two communities i and j divided by the sum of the weight of all edges, and $a_i = \sum_j e_{ij}$, *i.e.*, the expected fraction of edges within the cluster. Here are the key steps of that algorithm:

1. Given a set of n *ono* words $S = \{O_1, \cdots, O_n\}$. Create a network structure which consists of nodes $O_1, \cdots, O_n$, and edges. Here, the weight of an edge between O_i and O_j is a similarity value obtained by Eq. (7). If the "network density" of *ono* words is smaller than the parameter θ, we cut the edge. Here, "network density" refers to a ratio selected from the topmost edges. For example, if it was 0.9, we used the topmost 90% of all edges and cut the remains, where edges are sorted in the descending order of their similarity values.

2. Starting with a state in which each *ono* word is the sole member of one of n communities, we repeatedly joined communities together in pairs, choosing at each step the join that results in the greatest increase.

3. Suppose that two communities are merged into one by a join operation. The change in Q upon joining two communities i and j is given by:

$$\begin{aligned} \triangle Q_{ij} &= e_{ij} + e_{ji} - 2a_i a_j \\ &= 2(e_{ij} - a_i a_j) \end{aligned}$$

4. Apply step 3. to every pair of communities.

5. Join two communities such that $\triangle Q$ is maximum and create one community. If $\triangle Q < 0$, go to step 7.

6. Re-calculate e_{ij} and a_i of the joined community, and go to step 3.

7. Words within the same community are regarded as semantically similar.

The computational cost of the algorithm is known as $O((m+n)n)$ or $O(n^2)$, where m and n are the number of edges and nodes, respectively.

4 Experiments

4.1 Experimental Setup

The data for the classification of *ono* words have been taken from the Japanese *ono* dictionary (Ono, 2007) that consisted of 4,500 words. Of these, we selected 273 words, which occurred at least 5,000 in the document URLs from the WWW. The minimum frequency of a word was found to be 5,220, while the maximum was about 26 million. These words are classified into 10 classes. Word classes and examples of *ono* words from the dictionary are listed in Table 2.

"Id" denotes id number of each class. "Sense" refers to each sense of *ono* word within the same class, and "Num" is the number of words which should be assigned to each class. Each word

[2]When we used χ^2 statistic as a co-occurrence based similarity, MI in Eq. (7) is replaced by χ^2. In a similar way, $Cos(O_i, O_j)$ is replaced by $max - \alpha div(x, y)$, where max is the maximum value among all $\alpha div(x, y)$ values.

Table 2: Onomatopoetic words and # of words in each class

Id	*Sense*	Num	Onomatopoetic words
1	*laugh*	63	あっはっは (a,Q,h,a,Q,h,a), あはは (a,h,a,h,a), わはは (w,a,h,a,h,a) あはあは (a,h,a,a,h,a), いひひ (i,h,i,h,i), うっしっし (u,Q,s,i,Q,s,i), ···
2	*cry*	34	あーん (a,–,N), うわーん (u,w,a,–,N), あんあん (a,N,a,N), えんえん (e,N,e,N) うるうる (u,r,u,u,r,u), うるるん (u,r,u,r,u,N), うるっ(u,r,u,Q), えーん (e,–,N), ···
3	*pain*	34	いがいが (i,k,a,i,k,a), ひりひり (h,i,r,i,h,i,r,i), がじがじ (k,a,s,i,k,a,s,i) がんがん (k,a,N,k,a,N), ···
4	*anger*	33	かーっ(k,a,–,Q), かちん (k,a,t,i,N), かつん (k,a,t,u,N), かっ(k,a,Q), かっか (k,a,Q,k,a), がみがみ (k,a,m,i,k,a,m,i), かりかり (k,a,r,i,k,a,r,i), かんかん (k,a,N,k,a,N), ···
5	*spook*	31	あわわ (a,w,a,w,a), うぎゃー (u,ky,a,–), がーん (k,a,–,N), ぎく (k,i,k,u) ぎくっ(k,i,k,u,Q), ぎくり (k,i,k,u,r,i), ぎくん (k,i,k,u,N), ···
6	*panic*	25	あくせく (a,k,u,s,e,k,u), あたふた (a,t,a,h,u,t,a), あっぷあっぷ (a,Q,h,u,a,Q,h,u), あわあわ (a,w,a,a,w,a)···
7	*bloodless*	27	かくっ(k,a,k,u,Q), がくっ(k,a,k,u,Q), がっかり (k,a,Q,k,a,r,i), がっくり (k,a,Q,k,u,r,i) かくん (k,a,k,u,N), ぎゃふん (ky,a,h,u,N), ぎゅー (ky,u,–), ···
8	*deem*	13	うっとり (u,Q,t,o,r,i), きゅーん (ky,u,–,N), きゅん (ky,u,N) つくづく (t,u,k,u,t,u,k,u), ···
9	*feel delight*	6	うしうし (u,s,i,u,s,i), きゃぴきゃぴ (ky,a,h,i,ky,a,h,i) うはうは (u,–,h,a,–,u,–,h,a), ほいほい (h,o,i,h,o,i), るんるん (r,u,N,r,u,N), ···
10	*balk*	7	いじいじ (i,s,i,i,s,i), うじうじ (u,s,i,u,s,i), おずおず (o,s,u,o,s,u) ぐだぐだ (k,u,t,a,k,u,t,a), もじもじ (m,o,s,i,m,o,s,i), ···
	Total	273	

marked with bracket denotes phonetic sequences consisting of consonants and vowels.

We retrieved co-occurrences of *ono* words shown in Table 2 using the search engine, Google. We applied Newman clustering to the input words. For comparison, we implemented standard k-means which is often used as a baseline, as it is one of the simplest unsupervised clustering algorithms, and compared the results to those obtained by our method. We used Euclidean distance (L_2 norm) as a distance metric used in the k-means.

For evaluation of classification, we used Precision($Prec$), Recall(Rec), and F-$measure$ which is a measure that balances precision and recall (Bilenko et al., 2004). The precise definitions of these measures are given below:

$$Prec = \frac{\#PairsCorrectlyPredictedInSamecluster}{\#TotalPairsPredictedInSameCluster} \quad (8)$$

$$Rec = \frac{\#PairsCorrectlyPredictedInSameCluster}{\#TotalPairsInSameCluster} \quad (9)$$

$$F - measure = \frac{2 \times Prec \times Rec}{(Prec + Rec)} \quad (10)$$

4.2 Results

The results are shown in Table 3. "Co-occ. & Sounds" in Data refers to the results obtained by our method. "Co-occ." denotes the results obtained by a single measure, co-occurrence based distributional similarity measure, and "Sounds" shows the results obtained by orthographic similarity. "θ" in Table 3 shows a parameter θ used in the Newman clustering.[3] Table 3 shows best performance of each method against θ values. The best result was obtained when we used $phrase$-based search and a combined measure of co-occurrence(MI) and sounds (cos), and F-score was 0.451.

4.2.1 AND vs $phrase$-based search

Table 3 shows that overall the results using $phrase$-based search were better than those of AND search, and the maximum difference of F-score between them was 20.6% when we used a combined measure. We note that AND boolean search did not consider the position of a word in a document, while our assumption was that semantically similar words appeared in similar contexts. As a result, two *ono* words which were not semantically similar were often retrieved by AND boolean search. For example, consider two antonymous words, "a,h,a,h,a" (grinning broadly) and "w,a,–,N" (Wah, Wah). The co-occurrence frequency obtained by AND was 5,640, while that of $phrase$-based search was only one. The observation shows that we find phrase-based search to be a good choice.

[3] In case of k-means, we used the weights which satisfies network density.

Table 3: Classification results

Data	Algo.	Sim (Co-occ.)	Sim (Sounds)	Search method	θ	Prec	Rec	F	# of clusters
Co-occ. & Sounds	k-means	χ^2	cos	AND	.050	.134	.799	.229	10
				Phrase	.820	.137	.880	.236	10
		MI		AND	.050	.134	.562	.216	10
				Phrase	.150	.190	.618	**.289**	10
		χ^2	αdiv	AND	.680	.134	.801	.229	10
				Phrase	.280	.138	.882	.238	10
		MI		AND	.040	.134	.602	.219	10
				Phrase	.140	.181	.677	.285	10
	Newman	χ^2	cos	AND	.170	.182	.380	.246	9
				Phrase	.100	.322	.288	.304	14
		MI		AND	.050	.217	.282	**.245**	13
				Phrase	.080	.397	.520	**.451**	7
		χ^2	αdiv	AND	.130	.212	.328	.258	9
				Phrase	.090	.414	.298	.347	17
		MI		AND	.090	.207	.325	.253	6
				Phrase	.160	.372	.473	.417	8
Co-occ.	k-means	χ^2	–	AND	.460	.138	.644	.227	10
				Phrase	.110	.136	.870	.236	10
		MI		AND	.040	.134	.599	.219	10
				Phrase	.150	.191	.588	.286	10
	Newman	χ^2	–	AND	.700	.169	.415	.240	8
				Phrase	.190	.301	.273	.286	14
		MI		AND	.590	.159	.537	.245	3
				Phrase	.140	.275	.527	**.361**	5
Sounds	k-means	–	cos	–	.050	.145	.321	.199	10
			αdiv	–	.020	.126	.545	.204	10
	Newman	–	cos	–	.270	.151	.365	**.213**	4
			αdiv	–	.350	.138	.408	.206	3

4.2.2 A single vs combined similarity measure

To examine the effectiveness of the combined similarity measure, we used a single measure as a quality function of the Newman clustering, and compared these results with those obtained by our method. As shown in Table 3, the results with combining similarity measures improved overall performance. In the $phrase$-based search, for example, the F-score using a combined measure "Co-occ(MI) & Sounds(cos)" was 23.8% better than the baseline single measure "Sounds(cos)", and 9.0% better a single measure "Co-occ(MI)".

Figure 1 shows F-score by "Co-occ(MI) & Sounds(cos)" and "Co-occ(MI)" against changes in θ. These curves were obtained by $phrase$-based search. We can see from Figure 1 that the F-score by a combined measure "Co-occ(MI) & Sounds(cos)" was better than "Co-occ(MI)" with θ value ranged from .001 to .25. One possible reason for the difference of F-score between them is the edges selected by varying θ. Figure 2 shows the results obtained by each single measure, and a combined measure to examine how the edges selected by varying θ affect overall performance, F-measure. "Precision" in Figure 2 refers to the ratio of correct *ono* word pairs (edges) divided by the total number of edges. Here, correct *ono* word pairs were created by using the Japanese *ono* dictionary, *i.e.*, we extracted word pairs within the same sense of the dictionary. Surprisingly, there were no significant difference between a combined measure "Co-occ(MI) & Sounds(cos)" and a single measure "Co-occ(MI)" curves, while the precision of a single measure "Sounds" was constantly worse than that obtained by a combined measure. Another possible reason for the difference of F-score is due to product of MI and Cos in Eq. (7). Further work is needed to analyze these results in detail.

4.2.3 k-means vs Newman algorithms

We examined the results obtained by standard k-means and Newman clustering algorithms. As can be seen clearly from Table 3, the results with Newman clustering were better than those of the standard k-means at all search and similarity measures, especially the result obtained by Newman clustering showed a 16.2 % improvement over the k-means when we used Co-occ.(MI) & Sounds(cos) & $phrase$-based search. We recall that we used 273 *ono* words for clustering. However, Newman clustering is applicable for a large number of nodes and edges without decreasing accuracy too much, as it does not simply calculate the number of short-

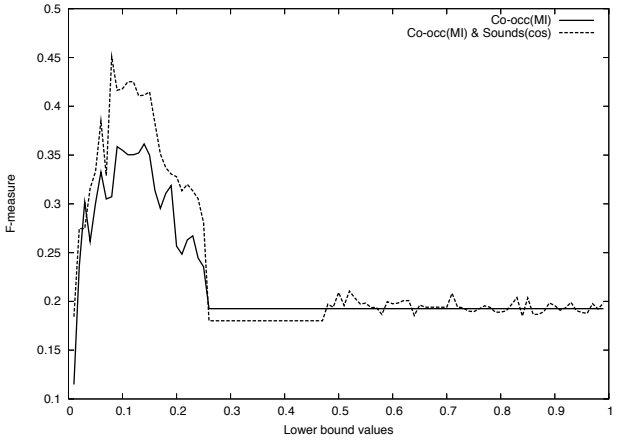


Figure 1: F-score against θ values

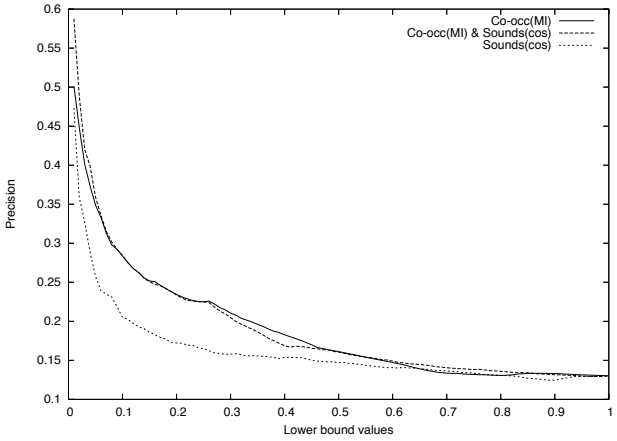


Figure 2: Precision against θ values

est paths between pairs of nodes, but instead calculates a simple quality function. Quantitative evaluation by applying the method to larger data from the Web is worth trying for future work.

4.3 Qualitative Analysis of Errors

Finally, to provide feedback for further development of our classification approach, we performed a qualitative analysis of errors. Consider the following clusters (the Newman output for Co-occ.(MI), Sounds(cos) and $phrase$-based search), where each parenthetic sequences denotes *ono* word:

A1: (t,o,Q) (t,o,Q,t,o) **(t,o,Q,k,i,N,t,o,Q,k,i,N)**
A2: (o,h,o,h,o), (e,h,e,h,e), (h,e,h,e,h,e), **(o,-,o,-)**
A3: (u,s,i,u,s,i), (m,o,s,i,m,o,s,i), **(m,o,s,o,m,o,s,o)**

Three main error types were identified:

1. Morphological idiosyncrasy: This was the most frequent error type, exemplified in **A1**, where "(t,o,Q,k,i,N,t,o,Q,k,i,N)" ($pain$ sense) was incorrectly clustered with other two words ($laugh$ sense) merely because orthographic similarity between them was large, as the phonetics sequences of "(t,o,Q,k,i,N,t,o,Q,k,i,N)" included "t" and "o".

2. Sparse data: Many of the low frequency *ono* words performed poorly. In **A2**, "(o,-,o,-)" (cry sense) was classified with other three words ($laugh$ sense) because it occurred few in our data.

3. Problems of polysemy: In **A3**, "(m,o,s,o,m,o,s,o)" ($pain$ sense) was clustered with other two words ($balk$ sense) of its gold standard class. However, the *ono* word has another sense, $balk$ sense when it co-occurred with action verbs.

5 Conclusion

We have focused on onomatopoetic words, and proposed a method for classifying them into a set of semantically similar words. We used a graph-based clustering algorithm, called Newman clustering with a combined different similarity measures. The results obtained by using the Web data showed a 9.0% improvement over the baseline single distributional similarity measure. There are number of interesting directions for future research.

The distributional similarity measure we used is the basis of the *ono* words, while other content words such as verbs and adverbs are also effective for classifying *ono* words. In the future, we plan to investigate the use of these words and work on improving the accuracy of classification. As shown in Table 2, many of the *ono* words consist of duplicative character sequences such as "h" and "a" of "a,h,a,h,a", and "h" and "i" of "i,h,i,h,i". Moreover, characters which consist of *ono* words within the same class match. For example, the hiragana character "は" (h,a) frequently appears in *laugh* sense class. These observations indicate that integrating edit-distance and our current similarity measure will improve overall performance.

Another interesting direction is a problem of polysemy. It clearly supports the classification of (Ono, 2007) to insist that some *ono* words belong to more than one cluster. For example, "(i,s,o,i,s,o)" has at least two senses, *panic* and *feel delight* sense. In order to accommodate this, we

should apply an appropriate soft clustering technique (Tishby et al., 1999; Reichardt and Bornholdt, 2006; Zhang et al., 2007).

Acknowledgments

We would like to thank the anonymous reviewers for their helpful suggestions. This material is supported in part by the Grant-in-aid for the Japan Society for the Promotion of Science(JSPS).

References

Bilenko, M., S. Basu, and R. J. Mooney. 2004. Integrating Constraints and Metric Learning in Semi-Supervised Clustering. In *Proc. of 21st International Conference on Machine Learning*, pages 81–88.

Chen, K. J. and C. J. Chen. 2000. Automatic Semantic Classification for Chinese Unknown Compound Nouns. In *Proc. of 38th Annual Meeting of the Association for Computational Linguistics*, pages 125–130.

Church, K. and P. Hunks. 1990. Word Association Norms, Mutual Information and Lexicography. In *In Proc. of 28th Annual Meeting of the Association for Computational Linguistics.*, pages 76–83.

Dagan, I., L. Lee, and F. Pereira. 1999. Similarity-based Models of Cooccurrence Probabilities. *Machine Learning*, 34(1-3):43–69.

Damerau, F. 1964. A Technique for Computer Detection and Correction of Spelling Errors. *Communications of the ACM*, 7:171–176.

Galley, M. and K. McKeown. 2003. Improving Word Sense Disambiguation in Lexical Chaining. In *Proc. of 19th International Joint Conference on Artificial Intelligence*, pages 1486–1488.

Geffet, M. and I. Dagan. 2004. Feature Vector Quality and Distributional Similarity. In *Proc. of 20th International Conference on Computational Linguistics*, pages 247–253.

Hindle, D. 1990. Noun Classification from Predicate-argument Structures. In *Proc. of 28th Annual Meeting of the Association for Computational Linguistics*, pages 268–275.

Jannink, J. and G. Wiederhold. 1999. Thesaurus Entry Extraction from an On-line Dictionary. In *Proc. of Fusion'99*.

Lee, L. 1999. Measures of Distributional Similarity. In *Proc. of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 25–32.

Lin, D. 1998. Automatic Retrieval and Clustering of Similar Words. In *Proc. of 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*, pages 768–773.

Matsuo, Y., T. Sakaki, K. Uchiyama, and M. Ishizuka. 2006. Graph-based Word Clustering using a Web Search Engine. In *Proc. of 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP2006)*, pages 542–550.

McLeod, W. T. 1991. *The COLLINS Dictionary and Thesaurus*. HarperCollinsPublishers.

Mihalcea, R. 2005. Unsupervised Large Vocabulary Word Sense Disambiguation with Graph-based Algorithms for Sequence Data Labeling. In *Proc. of the Human Language Technology / Empirical Methods in Natural Language PRocessing Conference*, pages 411–418.

Muller, P., N. Hathout, and B. Gaume. 2006. Synonym Extraction Using a Semantic Distance on a Dictionary. In *Proc. of the Workshop on TextGraphs*, pages 65–72.

Newman, M. E. J. 2004. Fast algorithm for detecting community structure in networks. *In Physics Review E*, (69, 066133).

Ono, M. 2007. *Nihongo Omomatope Jiten (in Japanese)*. Shougakukan.

Reichardt, J. and S. Bornholdt. 2006. Statistical Mechanics of Community Detection. *PHYICAL REVIEW E*, (74):1–14.

Resnik, P. 1995. Using Information Content to Evaluate Semantic Similarity in a Taxonomy. In *Proc. of 14th International Joint Conference on Artificial Intelligence*, pages 448–453.

Sinha, R. and R. Mihalcea. 2007. Unsupervised Graph-based Word Sense Disambiguation Using Measures of Word Semantic Similarity. In *Proc. of the IEEE International Conference on Semantic Computing*, pages 46–54.

Tishby, N., F. C. Pereira, and W. Bialek. 1999. The Information Bottleneck Method. In *Proc. of 37th Annual Allerton Conference on Communication Control and Computing*, pages 368–377.

Weeds, J. and D. Weir. 2005. Co-occurrence Retrieval: A Flexible Framework for Lexical Distributional Similarity. *Computational Linguistics*, 31(4):439–476.

Widdows, D. and B. Dorow. 2002. A Graph Model for Unsupervised Lexical Acquisition. In *Proc. of 19th International conference on Computational Linguistics (COLING2002)*, pages 1093–1099.

Zhang, S., R. S. Wang, and X. S. Zhang. 2007. Identification of Overlapping Community Structure in Complex Networks using Fuzzy C-means Clustering. *PHYSICA A*, (374):483–490.

Affinity Measures based on the Graph Laplacian

Delip Rao
Dept. of Computer Science
Johns Hopkins University
delip@cs.jhu.edu

David Yarowsky
Dept. of Computer Science
Johns Hopkins University
yarowsky@cs.jhu.edu

Chris Callison-Burch
Dept. of Computer Science
Johns Hopkins University
ccb@cs.jhu.edu

Abstract

Several language processing tasks can be inherently represented by a weighted graph where the weights are interpreted as a measure of relatedness between two vertices. Measuring similarity between arbitary pairs of vertices is essential in solving several language processing problems on these datasets. Random walk based measures perform better than other path based measures like shortest-path. We evaluate several random walk measures and propose a new measure based on commute time. We use the psuedo inverse of the Laplacian to derive estimates for commute times in graphs. Further, we show that this pseudo inverse based measure could be improved by discarding the least significant eigenvectors, corresponding to the noise in the graph construction process, using singular value decomposition.

1 Introduction

Natural language data lend themselves to a graph based representation. Words could be linked by explicit relations as in WordNet (Fellbaum, 1989) or documents could be linked to one another via hyperlinks. Even in the absence of such a straightforward representation it is possible to derive meaningful graphs such as the nearest neighbor graphs as done in certain manifold learning methods (Roweis and Saul, 2000; Belkin and Niyogi, 2001). All of these graphs share the following properties:

- They are edge-weighted.
- The edge weight encodes some notion of relatedness between the vertices.
- The relation represented by edges is at least weakly transitive. Examples of such relations include, "is similar to", "is more general than", and so on. It is important that the relations selected are transitive for the random walk to make sense.

Such graphs present several possibilities in solving language problems on the data. One such task is, given two vertices in the graph we would like to know how related the two vertices are. There is an abundance of literature on this topic, some of which will be reviewed here. Finding similarity between vertices in a graph could be an end in itself, as in the lexical similarity task, or could be a stage before solving other problems like clustering and classification.

2 Contributions of this paper

The major contributions of this paper are

- A comprehensive evaluation of various random walk based measures
- Propose a new similarity measure based on commute time.
- An improvement to the above measure by eliminating noisy features via singular value decomposition.

Coling 2008: Proceedings of 3rd Textgraphs workshop on Graph-Based Algorithms in Natural Language Processing, pages 41–48
Manchester, August 2008

3 Problem setting

Consider an undirected graph $G(V, E, \mathbf{W})$ with vertices V, edges E, and $\mathbf{W} = [w_{ij}]$ be the symmetric adjacency weight matrix with w_{ij} as the weight of the edge connecting vertices i and j. The weight, $w_{ij} = 0$ for vertices i and j that are not neighbors and when $w_{ij} > 0$ it is interpreted as an indication of relatedness between i and j. In our case, we consider uniformly weighted graphs, i.e, $w_{ij} = 1$ for neighbors but this need not be the case. Let $n = |V|$ be the order of the graph. We define a relation $\mathsf{sim} : V \times V \rightarrow \mathbb{R}^+$ such that $\mathsf{sim}(i, j)$ is the relatedness between vertices i and j. There are several ways to define sim; the ones explored in this paper are:

- $\mathsf{sim_G}(i, j)$ is the reciprocal of the shortest path length between vertices i and j. Note that this is not a random walk based measure but a useful baseline for comparison purposes.
- $\mathsf{sim_B}(i, j)$ is the probability of a random walk from vertex i to vertex j using all paths of length less than m.
- $\mathsf{sim_P}(i, j)$ is the probability of a random walk from vertex i to vertex j defined via a pagerank model.
- $\mathsf{sim_C}(i, j)$ is a function of the commute time between vertex i and vertex j.

4 Data and Evaluation

We evaluate each of the similarity measure we consider by using a linguistically motivated task of finding lexical similarity. Deriving lexical relatedness between terms has been a topic of interest with applications in word sense disambiguation (Patwardhan et al., 2005), paraphrasing (Kauchak and Barzilay, 2006), question answering (Prager et al., 2001), and machine translation (Blatz et al., 2004) to name a few. Lexical relatedness between terms could be derived either from a thesaurus like WordNet or from raw monolingual corpora via distributional similarity (Pereira et al., 1993). WordNet is an interesting graph-structured thesaurus where the vertices are the words and the edges represent relations between the words. For the purpose of this work, we only consider relations like hypernymy, hyponymy, and synonymy. The importance of this problem has generated copious literature in the past – see (Pedersen et al., 2004) or (Budanitsky and Hirst, 2006) for a detailed review of various lexical relatedness measures on WordNet. Our focus in this paper is not to derive the best similarity measure for WordNet but to use WordNet and the lexical relatedness task as a method to evaluate the various random walk based similarity measures. Following the tradition in previous literature we evaluate on the Miller and Charles (1991) dataset. This data consists of 30 word-pairs along with human judgements which is a real value between 1 and 4. For every measure we consider, we derive similarity scores and compare with the human judgements using the Spearman rank correlation coefficient.

5 Graph construction

For the purpose of evaluation of the random walk measures, we construct a graph for every pair of words for which similarity has to be computed. This graph is derived from WordNet as follows:

- For each word w in the pair (w_1, w_2):
  - Add an edge between w and all of its parts of speech. For example, if the word is coast, add edges between coast and coast#noun and coast#verb.
  - For each word#pos combination, add edges to all of its senses (For example, coast#noun#1 through coast#noun#4.
  - For each word sense, add edges to all of its hyponyms
  - For each word sense, add edges to all of its hypernyms recursively.

In this paper we consider uniform weights on all edges as our main aim is to illustrate the different random walk measures rather than fine tune the graph construction process.

6 Shortest path based measure

The most obvious measure of distance in a graph is the shortest path between the vertices which is defined as the minimum number of intervening edges between two vertices. This is also known as the geodesic distance. To convert this distance measure to a similarity measure, we take the reciprocal of the shortest-path length. We refer to this as the geodesic similarity. This is not a random walk

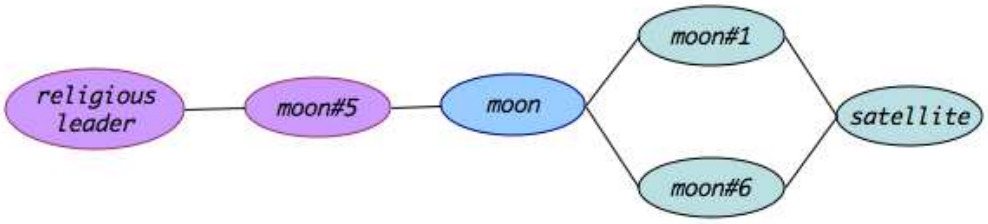


Figure 1: Shortest path distances on graphs

measure but will serve as an important baseline for our work. As can be observed from Table 1, the

Method	Spearman correlation
Geodesic	0.275

Table 1: Similarity using shortest-path measure.

correlation is rather poor for the shortest path measure.

7 Why are shortest path distances bad?

While shortest-path distances are useful in many applications, it fails to capture the following observation. Consider the subgraph of WordNet shown in Figure 1. The term `moon` is connected to the terms `religious_leader` and `satellite`[1]. Observe that both `religious_leader` and `satellite` are at the same shortest path distance from `moon`. However, the connectivity structure of the graph would suggest `satellite` to be "more" similar than `religious_leader` as there are multiple senses, and hence multiple paths, connecting `satellite` and `moon`.

Thus it is desirable to have a measure that captures not only path lengths but also the connectivity structure of the graph. This notion is elegantly captured using random walks on graphs.

7.1 Similarity via Random walks

A random walk is a stochastic process that consists of a sequence of discrete steps taken at random defined by a distribution. Random walks have interesting connections to Brownian motion, heat diffusion and have been used in semi-supervised learning – for example, see (Zhu et al., 2003). Certain properties of random walks are defined for ergodic processes only[2]. In our work, we assume these hold true as the graphs we deal with are connected, undirected, and non-bipartite.

7.1.1 Bounded length walks

As our first random walk measure, we consider the bounded length walk – i.e., all random walks of length less than or equal to a bound m. We derive a probability transition matrix $\mathbf{P}$ from the weight matrix $\mathbf{W}$ as follows:

$$\mathbf{P} = \mathbf{D}^{-1}\mathbf{W}$$

where, $\mathbf{D}$ is a diagonal matrix with $d_{ii} = \sum_{j=1}^{n} w_{ij}$. Observe that:

- $p_{ij} = \mathbf{P}[i,j] \geq 0$, and
- $\sum_{j=1}^{n} p_{ij} = 1$

Hence p_{ij} can be interpreted as the probability of transition from vertex i to vertex j in one step. It is easy to observe that $\mathbf{P}^k$ gives the transition probability from vertex i to vertex j in k steps. This leads to the following similarity measure:

$$\mathbf{S} = \mathbf{P} + \mathbf{P}^2 + \mathbf{P}^3 + ... + \mathbf{P}^m$$

Observe that $\mathbf{S}[i,j]$ derives the total probability of transition from vertex i to vertex j in at most m steps[3]. Given $\mathbf{S}$, we can derive several measures of similarity:

1. Bounded Walk: $\mathbf{S}[i,j]$
2. Bounded Walk Cosine: dot product of rowvectors $\mathbf{S}_i$ and $\mathbf{S}_j$.

When we evaluate these measures on the Miller-Charles data the results shown in Table 2. are observed. For this experiment, we consider all walks that are at most 20 steps long, i.e., $m = 20$. Observe that these results are significantly better than the Geodesic similarity based on shortest-paths.

[1]The `religious_leader` sense of `moon` is due to Sun Myung Moon, a US religious leader.

[2]A stochastic process is ergodic if the underlying Markov chain is irreducible and aperiodic. A Markov chain is irreducible if there exists a path between any two states and it is aperiodic if the GCD of all cycle lengths is one.

[3]The matrix $\mathbf{S}$ is row normalized to ensure that the entries can be interpreted as probabilities.

Method	Spearman correlation
Bounded Walk	0.346
Bounded Walk Cosine	0.365

Table 2: Similarity using bounded random walks ($m = 20$).

7.1.2 How many paths are sufficient?

In the previous experiment, we arbitrarily fixed $m = 20$. However, as observed in Figure 2. , beyond a certain value the choice of m does not affect the result as the random walk converges to its stationary distribution. The choice of m depends on

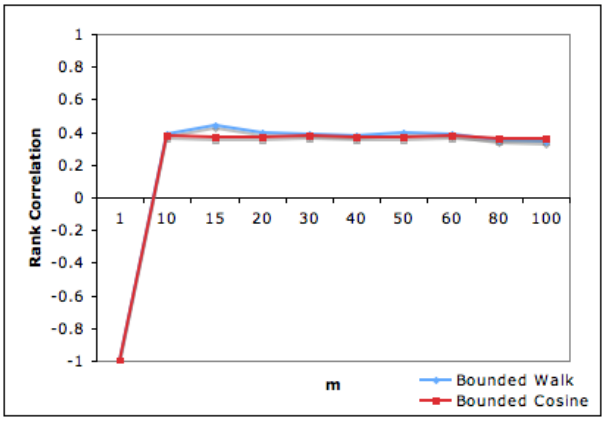


Figure 2: Effect of m in Bounded walk

the amount of computation available. A reasonably large value of m ($m > 10$) should be sufficient for most purposes and one could use lower values of m to derive an approximation for this measure. One could derive an upper bound on the value of m using the mixing time of the underlying Markov chain (Aldous and Fill, 2001).

7.1.3 Similarity via pagerank

Pagerank (Page et al., 1998) is the celebrated citation ranking algorithm that has been applied to several natural language problems from summarization (Erkan and Radev, 2004) to opinion mining (Esuli and Sebastiani, 2007) to our task of lexical relatedness (Hughes and Ramage, 2007). Pagerank is yet another random walk model with a difference that it allows the random walk to "jump" to its initial state with a nonzero probability (α). Given the probability transition matrix $\mathbf{P}$ as defined above, a stationary distribution vector for any vertex (say i) could be derived as follows:

1. Let $\mathbf{e}_i$ be a vector of all zeros with $\mathbf{e}_i(i) = 1$
2. Let $v_0 = \mathbf{e}_i$
3. Repeat until $\|v_t - v_{t-1}\|_F < \epsilon$
   - $v_{t+1} = \alpha v_t \mathbf{P} + (1 - \alpha) v_0$
   - $t = t + 1$
4. Assign v_{t+1} as the stationary distribution for vertex i.

Armed with the stationary distribution vectors for vertices i and j, we define pagerank similarity either as the cosine of the stationary distribution vectors or the reciprocal Jensen-Shannon (JS) divergence[4] between them. Table 3. shows results on the Miller-Charles data. We use $\alpha = 0.1$, the best value on this data. Observe that these results are

Method	Spearman correlation
Pagerank JS-Divergence	0.379
Pagerank Cosine	0.393

Table 3: Similarity via pagerank ($\alpha = 0.1$).

better than the best bounded walk result. We further note that our results are different from that of (Hughes and Ramage, 2007) as they use extensive feature engineering and weight tuning during the graph generation process that we have not been able to reproduce. Hence for simplicity we stuck to a simpler graph generation process. Nevertheless, the result in Table 3. is still useful as we are interested in the performance of the various spectral similarity measures rather than achieving the best performance on the lexical relatedness task. The graphs we use in all methods are identical making comparisons across methods possible.

7.2 Similarity via Hitting Time

Given a graph with the transition probability matrix $\mathbf{P}$ as defined above, the hitting time between vertices i and j, denoted as $h(i, j)$, is defined as the expected number of steps taken by a random walker to first encounter vertex j starting from vertex i. This can be recursively defined as follows:

$$h(i,j) = \begin{cases} 1 + \sum_{k\,:\,w_{ik} > 0} p_{ik} h(k,j) & \text{if } i \neq j \\ 0 & \text{if } i = j \end{cases} \quad (1)$$

[4]The Jensen-Shannon divergence between two distributions p and q is defined as $D(p \parallel a) + D(q \parallel a)$, where $D(. \parallel .)$ is the Kullback-Liebler divergence and $a = (p + q)/2$. Note that unlike KL-divergence this measure is symmetric. See (Lin, 1991) for additional details.

The lower the hitting times of two vertices, the more similar they are. It can be easily verified that hitting time is not a symmetric relation hence graph theory literature suggests another symmetric measure – the commute time.[5] The commute time, $c(i,j)$, is the expected number of steps taken to leave vertex i, reach vertex j, and return back to i. Thus,

$$c(i,j) = h(i,j) + h(j,i) \tag{2}$$

Observe that, the commute time is a metric in that it is positive definite, symmetric, and satisifies triangle inequality. Hence, commute time could be used as a distance measure as well. We derive a similarity measure from this distance measure using the following lemma.

Lemma 1. *For every edge (i,j), $c(i,j) \leq 2l$ where $l = |E|$, the number of edges.*

Proof. This can be easily observed by defining a Markov chain on the edges with probability transition matrix $\mathbf{Q}$ with $2l$ states, such that $Q_{e_1e_2} = 1/degree(e_1 \cap e_2)$. Since this matrix is doubly stochastic, the stationary distribution on this chain will be uniform with a probability $1/2l$. Now $c(i,j) = h(i,j)+h(j,i)$, is the expected time for a walk to start at i, visit j, and return back to i. When the stationary probability at each edge is $1/2l$, this expected time evaluates to $2l$. Hence the commute time can be at most $2l$. □

This lemma allows us to define a similarity measure as follows:

$$sim_C(i,j) = 1 - \frac{c(i,j)}{2l} \tag{3}$$

Observe that the measure defined in Equation 3 is a metric and further its range is defined in $[0,1]$. We now only need a way to compute the commute times to use Equation 3. One could compute the hitting times and hence the commute times from the Equations 1 and 2 using dynamic programming, akin to shortest paths in graphs. In this paper, we instead choose to derive commute times via the graph Laplacian. This also allows us to handle "noise" in the graph construction process which cannot be taken care by naive dynamic programming.

[5]Note that distance measures, in general, need not be symmetric but we interpret distance as proximity which mandates symmetry.

Chandra et. al. (1989) show that the commute time between two vertices is equal to the resistance distance between them. Resistance distance, as proposed by Klein and Randic (1993), is the effective resistance between two vertices in the electrical network represented by the graph, where the edges have resistance $1/w_{ij}$. Xiao and Gutman (2003), show the relation between resistance distances in graphs to the Laplacian spectrum, thus enabling a way to derive commute times from the graph Laplacian in closed form.

We now introduce graph Laplacians, which are interesting in their own right besides being related to commute time. The Laplacian of a graph could be viewed as a discrete version of the Laplace-Beltrami operator on Riemannian manifolds. It is defined as

$$\mathcal{L} = \mathbf{D} - \mathbf{W}$$

The graph Laplacian has interesting properties and a wide range of applications, in semi-supervised learning (Zhu et al., 2003), non-linear dimensionality reduction (Roweis and Saul, 2000; Belkin and Niyogi, 2001), and so on. See (Chung, 1997) for a thorough introduction on Laplacians and their properties. We depend on the fact that $\mathcal{L}$ is:

1. symmetric (since D and W are for undirected graphs)
2. positive-semidefinite : since it is symmetric, all of the eigenvalues are real and by the Greshgorin circle theorem, the eigenvalues must also be non-negative and hence $\mathcal{L}$ is positive-semidefinite.

Throughout this paper we use normalized Laplacians as defined below:

$$\mathbf{L} = \mathbf{D}^{-1/2}\mathcal{L}\mathbf{D}^{-1/2} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$$

The normalized Laplacians preserve all properties of the Laplacian by construction.

As noted in Xiao and Gutman (2003), the resistance distances can be derived from the generalized Moore-Penrose pseudo-inverse of the graph Laplacian($\mathbf{L}^\dagger$) – also called the inverse Laplacian. Like Laplacians, their pseudo inverse counterparts are also symmetric, and positive semi-definite.

Lemma 2. *$L^\dagger$ is symmetric*

Proof. The Moore-Penrose pseudo-inverse is defined as $\mathbf{L}^\dagger = (\mathbf{L}^T\mathbf{L})^{-1}\mathbf{L}^T$. From this definition, it is clear that $(\mathbf{L}^\dagger)^T = (\mathbf{L}^T)^\dagger$. By the symmetry

property of graph Laplacians, $\mathbf{L}^T = \mathbf{L}$. Hence, $(\mathbf{L}^\dagger)^T = \mathbf{L}^\dagger$. □

Lemma 3. *$L^\dagger$ is positive semi-definite*

Proof. We make use of the following properties from (Chung, 1997):

- The Laplacian, $\mathbf{L}$, is positive semi-definite (also shown above).
- If the Eigen-decomposition of $\mathbf{L}$ is $\mathbf{Q}\Lambda\mathbf{Q}^T$, then the Eigen-decomposition of the pseudo-inverse $\mathbf{L}^\dagger$ is $\mathbf{Q}\Lambda^{-1}\mathbf{Q}^T$. If any of the eigenvalues of $\mathbf{L}$ is zero then the corresponding eigenvalue for $\mathbf{L}^\dagger$ is also zero.

Since $\mathbf{L}$ is positive semi-definite, and the eigenvalues of $\mathbf{L}^\dagger$ have the same sign as $\mathbf{L}$, the pseudo inverse $\mathbf{L}^\dagger$ has to be positive semi-definite. □

Lemma 4. *The inverse Laplacian is a gram matrix*

Proof. To prove this, we use the fact that the Laplacian Matrix is symmetric and positive semi-definite. Hence by Cholesky decomposition we can write $\mathbf{L} = \mathbf{U}\mathbf{U}^T$.

Therefore $\mathbf{L}^\dagger = (\mathbf{U}^T)^\dagger\mathbf{U}^\dagger = (\mathbf{U}^\dagger)^T(\mathbf{U}^\dagger)$.

Hence $\mathbf{L}^\dagger$ is a matrix of dot-products or a gram-matrix. □

Thus, from Lemmas 2, 3 and 4, the inverse Laplacian $\mathbf{L}^\dagger$ is a valid Kernel.

7.2.1 Similarity measures from the Laplacian

The pseudo inverse of the Laplacian allows us to compute the following similarity measures.

1. Since $\mathbf{L}^\dagger$ is a kernel, $\mathbf{L}^\dagger_{ij}$ can be interpreted a similarity value of vertices i and j.

2. **Commute time**: This is due to (Aldous and Fill, 2001). The commute time, $c(i,j) \propto (\mathbf{L}^\dagger_{ii} + \mathbf{L}^\dagger_{jj} - 2\mathbf{L}^\dagger_{ij})$. This allows us to derive similarities using Equation 3.

Evaluating the above measures with the Miller-Charles data yields results shown in Table 4. Again, these results are better than the other random walk methods compared in the paper.

Method	Spearman correlation
$\mathbf{L}^\dagger_{ij}$	0.469
Commute Time (sim_C)	0.520

Table 4: Similarity via inverse Laplacian.

7.2.2 Noise in the graph construction process

The graph construction process outlined in Section 5 is not necessarily the best one. In fact, any method that constructs graphs from existing data incorporates "noise" or extraneous features. These could be spurious edges between vertices, missing edges, or even improper edge weights. It is however impossible to know any of this a priori and some noise is inevitable. The derivation of commute times via the pseudo inverse of a noisy Laplacian matrix makes it even worse because the pseudo inverse amplifies the noise in the original matrix. This is because the largest singular value of the pseudo inverse of a matrix is equal to the inverse of the *smallest* singular value of the original matrix. A standard technique in signal processing and information retrieval to eliminate noise or handle missing values is to use singular value decomposition (Deerwester et al., 1990). We apply SVD to handle noise in the graph construction process.

For a given matrix A, SVD decomposes A into three matrices U, S, and V such that $A = USV^T$, where S is a diagonal matrix of eigenvalues of A, and U and V are orthonormal matrices containing the left and the right eigenvectors respectively. The top-k eigenvectors and eigenvalues are computed using the iterative method by Lanczos-Arnoldi (using LAPACK) and the product of these matrices represents a "smoothed" version of the original Laplacian. The pseudo inverse is then computed on this smooth Laplacian. Table 5., shows the improvements obtained by discarding bottom 20% of the eigenvalues.

Method	Original	After SVD
$\mathbf{L}^\dagger_{ij}$	0.469	**0.472**
Commute Time (sim_C)	0.520	**0.542**

Table 5: Denoising graph Laplacian via SVD

Figure 3. shows the dependence on the number of eigenvalues selected. As can be observed in both curves there is a reduction in performance by adding the last few eigenvectors and hence may be safely discarded. This observation is true in other text processing tasks like document clustering or classification using Latent Semantic Indexing.

8 Related Work

Apart from the related work cited throughout this paper, we would also like to note the paper by Yen

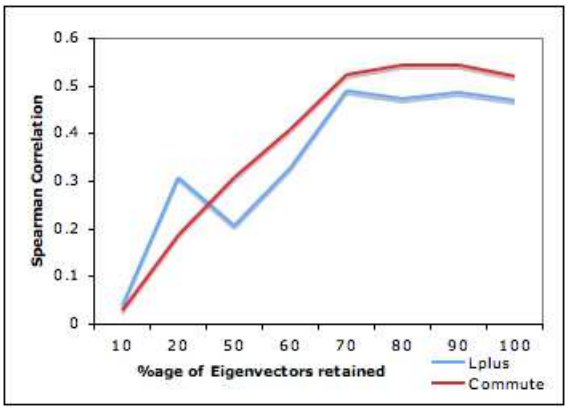


Figure 3: Noise reduction via SVD.

et al (2007) on using sigmoid commute time kernel on a graph for document clustering but our work differs in that our goal was to study various random walk measures rather than a specific task and we provide a new similarity measure (ref. Eqn 3) based on an upper bound on the commute time (Lemma 1). Our work also suggests a way to handle noise in the graph construction process.

9 Conclusions and Future Work

This paper presented an evaluation of random walk based similarity measures on weighted undirected graphs. We provided an intuitive explanation of why random walk based measures perform better than shortest-path or geodesic measures, and backed it with empirical evidence. The random walk measures we consider include bounded length walks, pagerank based measures, and a new measure based on the commute times in graphs. We derived the commute times via pseudo inverse of the graph Laplacian. This enables a new method of graph similarity using SVD that is robust to the noise in the graph construction process. Further, the inverse Laplacian is also interesting in that it is a kernel by itself and could be used for other tasks like word clustering, for example.

Acknowledgements

The authors would like to thank David Smith and Petros Drineas for useful discussions and to Fan Chung for the wonderful book on Spectral Graph theory.

References

Aldous and Fill. 2001. *Reversible Markov Chains and Random Walks on Graphs*. In preparation.

Belkin, Mikhail and Partha Niyogi. 2001. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Proceedings of the NIPS*.

Blatz, John, Erin Fitzgerald, George Foster, Simona Gandrabur, Cyril Goutte, Alex Kulesza, Alberto Sanchis, and Nicola Ueffing. 2004. Confidence estimation for machine translation. In *Proceeding of the COLING*.

Budanitsky, Alexander and Graeme Hirst. 2006. Evaluating wordnet-based measures of lexical semantic relatedness. *Computational Linguistics*, 32(1):13–47.

Chandra, Ashok, Prabhakar Raghavan, Walter Ruzzo, Roman Smolensky, and Prasoon Tiwari. 1989. The electrical resistance of a graph captures its commute and cover times. In *Proceedings of the STOC*.

Chung, Fan. 1997. Spectral graph theory. In *CBMS: Conference Board of the Mathematical Sciences, Regional Conference Series*.

Deerwester, Scott, Susan Dumais, George Furnas, Thomas Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41.

Erkan, Günes and Dragomir Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research (JAIR)*, 22:457–479.

Esuli, Andrea and Fabrizio Sebastiani. 2007. Pageranking wordnet synsets: An application to opinion mining. In *Proceedings of the ACL*, pages 424–431.

Fellbaum, Christaine, editor. 1989. *WordNet: An Electronic Lexical Database*. The MIT Press.

Hughes, Thad and Daniel Ramage. 2007. Lexical semantic relatedness with random graph walks. In *Proceedings of the EMNLP*.

Kauchak, David and Regina Barzilay. 2006. Paraphrasing for automatic evaluation. In *Proceedings HLT-NAACL*.

Klein, D. and M. Randic. 1993. Resistance distance. *Journal of Mathematical Chemistry*, 12:81–95.

Lin, Jianhua. 1991. Divergence measures based on the shannon entropy. *IEEE Transactions on Information Theory*, 37(1).

Miller, G. and W. Charles. 1991. Contextual correlates of semantic similarity. In *Language and Cognitive Process*.

Page, Larry, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1998. The pagerank citation ranking: Bringing order to the web. *Technical report, Stanford University, Stanford, CA*.

Patwardhan, Siddharth, Satanjeev Banerjee, and Ted Pedersen. 2005. Senserelate:: Targetword-A generalized framework for word sense disambiguation. In *Proceedings of the ACL*.

Pedersen, Ted, Siddharth Patwardhan, and Jason Michelizzi. 2004. Wordnet::similarity - measuring the relatedness of concepts. In *Proceedings of the AAAI*.

Pereira, Fernando, Naftali Tishby, and Lillian Lee. 1993. Distributional clustering of english words. In *Proceedings of the ACL*.

Prager, John M., Jennifer Chu-Carroll, and Krzysztof Czuba. 2001. Use of wordnet hypernyms for answering what-is questions. In *Proceedings of the Text REtrieval Conference*.

Roweis, Sam and Lawrence Saul. 2000. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290:2323–2326.

Xiao, W. and I. Gutman. 2003. Resistance distance and laplacian spectrum. *Theoretical Chemistry Association*, 110:284–289.

Yen, Luh, Francois Fouss, Christine Decaestecker, Pascal Francq, and Marco Saerens. 2007. Graph nodes clustering based on the commute-time kernel. In *Proceedings of the PAKDD*.

Zhu, Xiaojin, Zoubin Ghahramani, and John Lafferty. 2003. Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the ICML*.

Semantic structure from Correspondence Analysis

Barbara McGillivray
Dipartimento di Linguistica
Università di Pisa
Pisa, Italy
barbara.mcgillivray@aksis.uib.no

Christer Johansson
Dept. of Linguistics
University of Bergen
Bergen, Norway
christer.johansson@uib.no

Daniel Apollon
Text Technology Lab.
Aksis, UNIFOB
Bergen, Norway
daniel.apollon@aksis.uib.no

Abstract

A common problem for clustering techniques is that clusters overlap, which makes graphing the statistical structure in the data difficult. A related problem is that we often want to see the distribution of factors (variables) as well as classes (objects). Correspondence Analysis (CA) offers a solution to both these problems. The structure that CA discovers may be an important step in representing similarity. We have performed an analysis for Italian verbs and nouns, and confirmed that similar structures are found for English.

1 Introduction

Over the past years, distributional methods have been used to explore the semantic behaviour of verbs, looking at their contexts in corpora (Landauer and Laham, 1998; Redington and Finch, 1998; Biemann, 2006, inter al.). We follow a general approach suggested already by Firth (1957), to associate distributional similarity with semantic similarity.

One question concerns the syntax-semantics interface. Results using distributions of verbs in context had an impact on verb classification (Levin, 1993), automatic verb clustering (Schulte im Walde, 2003), and selectional preference acquisition (Resnik, 1993; Li and Abe, 1995; McCarthy, 2001; Agirre and Martinez, 2001, inter al.).

In automatic verb clustering, verbs are represented by vectors of a multidimensional space whose dimensions (variables) are identified by some linguistic features, ranging, for example, from subcategorization frames to participation in diathesis alternations and lexical selectional preferences. The verbs cluster on co-occurrence with the features chosen, and such information provide a generalisation over the verbs with respect to the variables.

In the case of selectional preference acquisition, a verb (or a verb class) is associated to a class of nouns that can be the lexical fillers of a case frame slot for the verb. This allows us to calculate the association strength between the verb and its filler nouns. The generalisation step is performed for the case frame instances (observations) and produces more abstract noun classes that can be applied to unseen cases. This often utilizes hierarchies of existing thesauri or wordnets.

We propose a method that uses Correspondence Analysis (CA) to study the distribution (and associated semantic behaviour) of a list of verbs with nouns occurring in a particular syntactic relation, for example their subjects. This is collected from a corpus, and reflects usage in that corpus. Unlike clustering methods, this technique does not imply an exclusive choice between a) classifying verbs on the basis of the noun fillers in their syntactic frame, or b) associating noun classes to verbs (sometimes mediated by a semantic hierarchy). Instead, this approach yields a geometric representation of the relationships between the nouns and the verbs in a common dual space (biplot). CA aims to find an overall structure (if any) of the data. The method emphasizes unusual observations, as deviance from the expected is what creates the axes of the analysis. CA generalizes over the actual occurrences of verb-noun pairs in the corpus, and visualizes the shape of the correspondence space.

When associating verbs with nouns, CA takes as input a contingency table (here rows correspond to the verbs, and columns correspond to their subject fillers). Each verb is a row point in the multidimensional noun space, and each noun is a column point in the multidimensional verb space. The CA goals

Coling 2008: Proceedings of 3rd Textgraphs workshop on Graph-Based Algorithms in Natural Language Processing, pages 49–52
Manchester, August 2008

are to reduce the dimension of the dual original space, and to find an optimal subspace that is the closest to this cloud of points in the χ^2-metric. The best subspace is determined by finding the smallest number of orthogonal axes that describe the most variance from the original cloud.

Finally the coordinates of both row and column points of the χ^2 contingency table are projected onto this optimal subspace, simultaneously displaying row and column points. If we consider those points that are well represented, the closer they are in this geometric representation, the more similar their original distributions are. In this way, we can detect not only that there is a relationship between the verb (e.g. *explode*) and the noun (e.g. *bomb*), but also how each word relates to each *other* word.

2 Correspondence Analysis

CA is a data analytic technique developed by Benzécri in the 1960s, which has been widely used in describing large contingency tables and binary data. At the heart of CA is Singular Value Decomposition (SVD), from which many other methods were derived (Biplot, Classical Principal Component Analysis, PCA and more).

Compared to usual clustering methods, CA gives a more fine-grained view of the spread of the input points. Benzécri (1973) points out that CA is more efficient than clustering in terms of decomposition of variance. Secondly, CA represents possible regions in space with varying density, and produces a flexible "compound clustering" on both objects and variables. Verb-nouns association profiles may not cluster in distinct space regions, but may be evenly distributed, follow a gradient-like distribution, or show overlapping clusters. In such difficult cases for clustering, CA is able to offer a representation of the geometry of the input profiles. Finally, CA offers the possibility of reconstructing the original space from the output subspace.

Let us consider a data matrix M whose size is (r, c), the $(i, j)^{th}$ entry of M containing the number of occurrences of verb j with noun i as its subject in a corpus. We calculate the relative frequencies by dividing each entry $M(i, j)$ by the sum of row i, i. e. the frequency of noun i, to get the matrix of row profiles $R(i, j)$. Therefore, the more similar two row profiles i_1 and i_2 are, the more these two nouns can be considered as distributional synonyms.

The next step implies comparing the row profiles with the average distribution where each entry (i, j) is the product of the frequency of noun i by the frequency of verb j divided by the grand total N of the table. This comparison is calculated using the χ^2-distance (i.e. a weighted Euclidean distance), which eliminates effects of high frequency alone. The next formula shows calculations for rows. Calculations for columns are analogous.

$$\delta^2(i_1, i_2) = \sum_{j=1}^{c} \frac{(R(i_1, j) - R(i_2, j))^2}{\sum_{i=1}^{r} M(i, j)}$$

The χ^2-distance between a profile point and the average profile (barycentre) is called inertia of the profile point and the total inertia measures how the individual profiles p_i are spread around the barycentre:

$$Inertia = \frac{1}{N} \sum_{i=1}^{r} \sum_{j=1}^{c} M(i, j)\delta^2(p_i, \bar{p})$$

CA then searches for the optimal subspace S that minimises the distance from the profile points. Once specified its dimension $k \leq min(r - 1, c - 1)$, S is found by applying the Singular Value Decomposition (SVD) to matrix $R - 1\bar{p}$, which decomposes it as the product $N \cdot D \cdot M$: where D is a diagonal matrix with positive coefficients $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_k$ (singular values) and N and M are orthonormal matrices ($N^T N = M^T M = I$). The rows of M are the orthonormal basis vectors that define S (called principal axes of inertia) and the rows of matrix $F = N \cdot D$ are the projections of the row profiles onto S. For $k = 2$, this allows us to plot the new coordinates in a two-dimensional space and get the correspondence analysis of the row profiles.

The total inertia is decomposed into the direction of the principal axes of inertia. The first axis represents the direction where the inertia of the cloud is the maximum; the second axis maximises the inertia among all the directions orthogonal to the first axis, and so on.

The geometry of column profiles can be analysed similarly, because the two problems are directly linked and two transition formulae can be used to pass from one coordinate system to the other, explaining the French name "analyse des correspondances".

As a result, both analyses decompose the same inertia into the same system of principal axes. This allows us to merge the two representations in one single geometric display showing at the same time

the projections of row and column points in the subspace.

In addition to this dual space representation, CA gives a system of diagnostic measures for each of the two dual spaces:

• contributions of the rows (and columns) to the axes, i. e. the inertia of the points projected onto the axes, which contributes to the principal inertia;

• contributions of the axes to the row (and column) points;

• quality of representation (cumulative sum of contributions of the axes for each point); this highlights *well represented* points.

3 Explorations

We performed a CA using the Matlab Analytica Toolbox developed by Daniel Apollon. We tested this technique first on the Italian newspaper corpus LA REPUBBLICA, which consists of 450 million word tokens. This corpus was syntactically parsed using the MALT dependency parser (Nivre, 2006). A list of 196 verbs was compiled following the list of German verbs contained in (Schulte im Walde, 2003) and adapting it to Italian. Looking at the syntactic analyses of the corpus where the verbs of the list showed a subcategorization frame containing a subject slot, their lexical subject fillers were automatically extracted. The matrix M, whose 2553 row entries correspond to the nouns extracted as subject fillers, was then used as input for the CA ($|M| = 196 \times 2553 = 500388$).

Starting from the quality of representation scores of this analysis, we isolated a set of points with increasingly good representation, ending with an extremely faithful and low dimensional representation. We called this method "incremental pruning". Figure 1 shows the dual display of the analysis for the Italian data in a two dimensional space, after filtering out those points showing a quality of representation below a threshold of 30%.

We can conceptualize the data set C after a CA as the cumulative effect of three different underlying phenomena: K, R and E.

K can be seen as a reduction of the latent structure of C; it contains its *core* structure as it has been underlined by the analysis and left after pruning.

R refers to the *residual* variance, not included in the core analysis. It contains the most predictable points[1], which are plotted near to origin (barycentre of the data cloud). These points give a small contribution to the inertia of the principal axes.

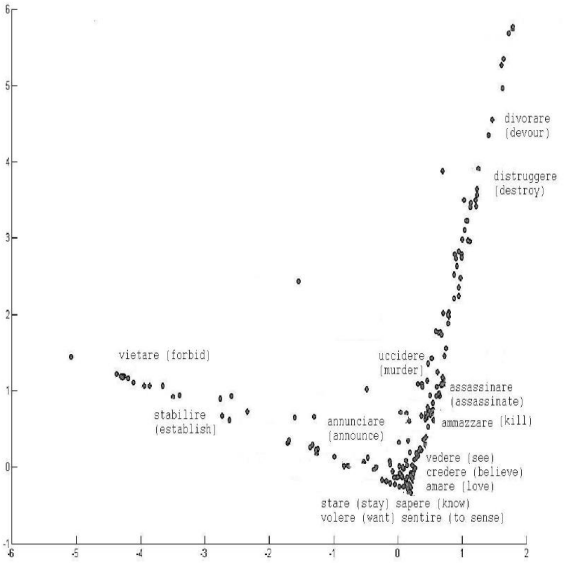


Figure 1: Correspondence graph for Italian

E contains the *error* in the representation, as well as badly represented points.

Points far from the origin display strong structure; they may correspond to rare words used in special contexts. Figure 1 shows that words related to destruction[2] are aligned in the same direction, whereas the second vector is mainly constituted by nouns and verbs that have to do with the political and legal area[3]. The first principal axis accounts for nearly 16% of the total inertia, whereas the second axis accounts for 12%. The first six axes accounts for over 70% of variation. Many words were not well represented, but contribute to variance.

We confirmed our method on English, using the British National Corpus[4]. A similar structure was found. We restrict ourselves to reproduce the graph for Italian.

[1] In our data: pronouns *she, I, he, every-, no-, some-body, who*, nouns with partly pronominal qualities *husband, wife, friend, sir, son, father, mother, fact, event*.

[2] Along the y-axis from top down to the middle, we find the nouns *flame, extend, stick of dynamite, excavator, chemotherapy, effusion, blaze, seism, demon, dynamite, fire, aviation, earthquake, artificer, explosive device, insect, gas, landslide, virus, rain, bulldozer, hurricane, wave, speculation, artillery, remorse, bomb, missile, violence, revolution*, etc.

[3] Along the x-axis we find, from left to the middle, the nouns *order, regulations, norm, code, legislation, rules, treaty, constitution, circular letter, system, directive, law, decree, article, judgement, amendment, court*, etc.

[4] via sketchengine http://www.sketchengine.co.uk

4 Conclusion

CA detects a structure for Italian verb-noun correspondences in LA REPUBBLICA ($\sim$ 450 million words). A similar structure was confirmed using BNC for English. Both global and local structures are found, which gives possibilities to represent lexical units with reference to both principal axes and word similarity. The main dimensions of the Italian corpus are topical (crime related vs. natural catastrophes, and laws vs. political institutions). Semantic relatedness were observed in closely mapped words. Both global and local structure is found, and we can speculate that this helps representing lexical units in semantic labeling (Giuglea and Moschitti, 2006) for machine learning tasks. We can conceptualize text graphs in two distinct usages: knowledge re-presenting (e.g. FrameNet) and visualizing relations in a data set. Our method belongs in the second category.

Acknowledgements

The first author is a MULTILINGUA fellow at Uni. Bergen, financially supported by a Marie Curie action (European Commission). We wish to thank the anonymous reviewers who gave important leads to future research.

References

Agirre, Eneko and David Martinez. 2001. Learning class-to-class selectional preferences. In *Proc. of the ACL/EACL Workshop on Computational Natural Language Learning*, pages 1–8, Toulouse, France.

Benzécri, Jean-Paul. 1973. *L'Analyse des Données*, volume 1. Dunod.

Biemann, Chris. 2006. Chinese Whispers – an Efficient Graph Clustering Algorithm and its Application to Natural Language Processing Problems. In *Proc. of the HLT-NAACL-06 Workshop on Textgraphs-06*, pages 73–80, New York, USA.

Firth, John R., 1957. *Studies in Linguistic Analysis*, chapter A synopsis of linguistic theory 1930-1955. Philological Society.

Giuglea, Ana-Maria and Alessandro Moschitti. 2006. Semantic Role Labeling via FrameNet, VerbNet and PropBank. In *Proc. of the 21st Int. Conf. on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 929–936, Sydney, Australia.

Landauer, Thomas K., Peter W. Foltz and Darrell Laham. 1998. Introduction to Latent Semantic Analysis. *Discourse Processes*, 25:259–284.

Levin, Beth. 1993. *English Verb Classes and Alternations*. The University of Chicago Press.

Li, Hang and Naoki Abe. 1995. Generalizing case frames using a thesaurus and the MDL principle. In *Proc. of Recent Advances in Natural Language Technology*, pages 239–248.

McCarthy, Diana. 2001. *Lexical Acquisition at the Syntax-Semantics Interface: Diathesis Alternations, Subcategorization Frames and Selectional Preferences*. Ph.D. thesis, University of Sussex.

Nivre, Joakim. 2006. *Inductive Dependency Parsing*. Springer.

Redington, Martin, Nick Chater and Steven Finch. 1998. Distributional information: A powerful cue for acquiring syntactic categories. *Cognitive Science*, 22:425–469.

Resnik, Philip. 1993. *Selection and Information: A Class-Based Approach to Lexical Relationships*. Ph.D. thesis, University of Pennsylvania.

Schulte im Walde, Sabine. 2003. *Experiments on the Automatic Induction of German Semantic Verb Classes*. Ph.D. thesis, Institut für Maschinelle Sprachverarbeitung, Universität Stuttgart.

Concept-graph based Biomedical Automatic Summarization using Ontologies

Laura Plaza Morales
Alberto Díaz Esteban
Pablo Gervás
Universidad Complutense de Madrid
C/Profesor José García Santesmases, s/n, Madrid 28040, Spain
lplazam@pas.ucm.es, albertodiaz@fdi.ucm.es,pgervas@sip.ucm.es

Abstract

One of the main problems in research on automatic summarization is the inaccurate semantic interpretation of the source. Using specific domain knowledge can considerably alleviate the problem. In this paper, we introduce an ontology-based extractive method for summarization. It is based on mapping the text to concepts and representing the document and its sentences as graphs. We have applied our approach to summarize biomedical literature, taking advantages of free resources as UMLS. Preliminary empirical results are presented and pending problems are identified.

1 Introduction

In recent years, the amount of electronic biomedical literature has increased explosively. Physicians and researchers constantly have to consult up-to date information according to their needs, but the process is time-consuming. In order to tackle this overload of information, text summarization can undoubtedly play a role.

Simultaneously, a big deal of resources, such as biomedical terminologies and ontologies, have emerged. They can significantly benefit the development of NLP systems, and in particular, when used in automatic summarization, they can increase the quality of summaries.

In this paper, we present an ontology-based extractive method for the summarization of biomedical literature, based on mapping the text to concepts in UMLS and representing the document and its sentences as graphs. To assess the importance of the sentences, we compute the centrality of their concepts in the document graph.

2 Previous Work

Traditionally, automatic summarization methods have been classified in those which generate *extracts* and those which generate *abstracts*. Although human summaries are typically abstracts, most of existing systems produce extracts.

Extractive methods build summaries on a superficial analysis of the source. Early summarization systems are based on simple heuristic features, as the position of sentences in the document (Brandow et al., 1995), the frequency of the words they contain (Luhn, 1958; Edmundson, 1969), or the presence of certain cue words or indicative phrases (Edmundson, 1969). Some advanced approaches also employ machine learning techniques to determine the best set of attributes for extraction (Kupiec et al., 1995). Recently, several graph-based methods have been proposed to rank sentences for extraction. LexRank (Erkan and Radev, 2004) is an example of a centroid-based method to multi-document summarization that assess sentence importance based on the concept of eigenvector centrality. It represents the sentences in each document by its *tf*idf* vectors and computes sentence connectivity using the cosine similarity. Even if results are promising, most of these approaches exhibit important deficiencies which are consequences of not capturing the semantic relations between terms (synonymy, hyperonymy, homonymy, and *co-occurs* and *associated-with* relations).

We present an extractive method for summarization which attempts to solve this deficiencies. Unlike researches conducted by (Yoo et al., 2007; Erkan and Radev, 2004), which cluster sentences to identify shared topics in multiple documents, in this work we apply clustering to identify groups

Coling 2008: Proceedings of 3rd Textgraphs workshop on Graph-Based Algorithms in Natural Language Processing, pages 53–56
Manchester, August 2008

of concepts closely related. We hypothesize that each cluster represents a *theme* or topic in the document, and we evaluate three different heuristics to ranking sentences.

3 Biomedical Ontologies. UMLS

Biomedical ontologies organize domain concepts and knowledge in a system of hierarchical and associative relations. One of the most widespread in NLP applications is UMLS[1] (Unified Medical Language System). UMLS consists of three components: the *Metathesaurus*, a collection of concepts and terms from various vocabularies and their relationships; the *Semantic Network*, a set of categories and relations used to classify and relate the entries in the Metathesaurus; and the *Specialist Lexicon*, a database of lexicographic information for use in NLP. In this work, we have selected UMLS for several reasons. First, it provides a mapping structure between different terminologies, including MeSH or SNOMED, and thus allows to translate between them. Secondly, it contains vocabularies in various languages, which allows to process multilingual information.

4 Summarization Method

The method proposed consists of three steps. Each step is discussed in detail below. A preliminary system has been implemented and tested on several documents from the corpus developed by *BioMed Central*[2].

As the preprocessing, text is split into sentences using GATE[3], and generic words and high frequency terms are removed, as they are not useful in discriminating between relevant and irrelevant sentences.

4.1 Graph-based Document Representation

This step consists in representing each document as a graph, where the vertices are the concepts in UMLS associated to the terms, and the edges indicate the relations between them. Firstly, each sentence is mapped to the UMLS Metathesaurus using *MetaMap* (Aronson, 2001). MetaMap allows to map terms to UMLS concepts, using n-grams for indexing in the ULMS Metathesaurus, and performing disambiguation to identify the correct concept for a term. Secondly, the UMLS concepts are extended with their hyperonyms. Figure 1 shows the graph for sentence *"The goal of the trial was to assess cardiovascular mortality and morbidity for stroke, coronary heart disease and congestive heart failure, as an evidence-based guide for clinicians who treat hypertension."*

Next, each edge is assigned a weight, which is directly proportional to the deep in the hierarchy at which the concepts lies (Figure 1). That is to say, the more specific the concepts connected are, the more weight is assigned to them. Expression (1) shows how these values are computed.

$$\frac{|\alpha \cap \beta|}{|\alpha \cup \beta|} = \frac{|\beta|}{|\alpha|} \tag{1}$$

where α is the set of all the parents of a concept, including the concept, and β is the set of all the parents of its immediate higher-level concept, including the concept.

Finally, the sentence graphs are merged into a document graph, enriched with the *associated-with* relations between the semantic types in UMLS corresponding to the concepts (Figure 1). Weights for the new edges are computed using expression (1).

4.2 Concept Clustering and Theme Recognition

The second step consists of clustering concepts in the document graph, using a *degree-based method* (Erkan and Radev, 2004). Each cluster is composed by a set of concepts that are closely related in meaning, and can be seen as a theme in the document. The most central concepts in the cluster give the sufficient and necessary information related to its theme. We hypothesize that the document graph is an instance of a *scale-free network* (Barabasi, 1999). Following (Yoo et al., 2007), we introduce the *salience* of vertices. Mathematically, the salience of a vertex (v_i) is calculated as follows.

$$salience(v_i) = \sum_{e_j | \exists v_k \wedge e_j conecta(v_i, v_k)} weight(e_j) \tag{2}$$

Within the set of vertices, we select the *n* that present the higher salience and iteratively group them in *Hub Vertex Sets* (HVS). A HVS represents a group of vertices that are strongly related to each other. The remaining vertices are

[1] NLM Unified Medical Language System (UMLS). URL: http://www.nlm.nih.gov/research/umls

[2] BioMed Central: http://www.biomedcentral.com/

[3] GATE (Generic Architecture for Text Engineering): http://gate.ac.uk/

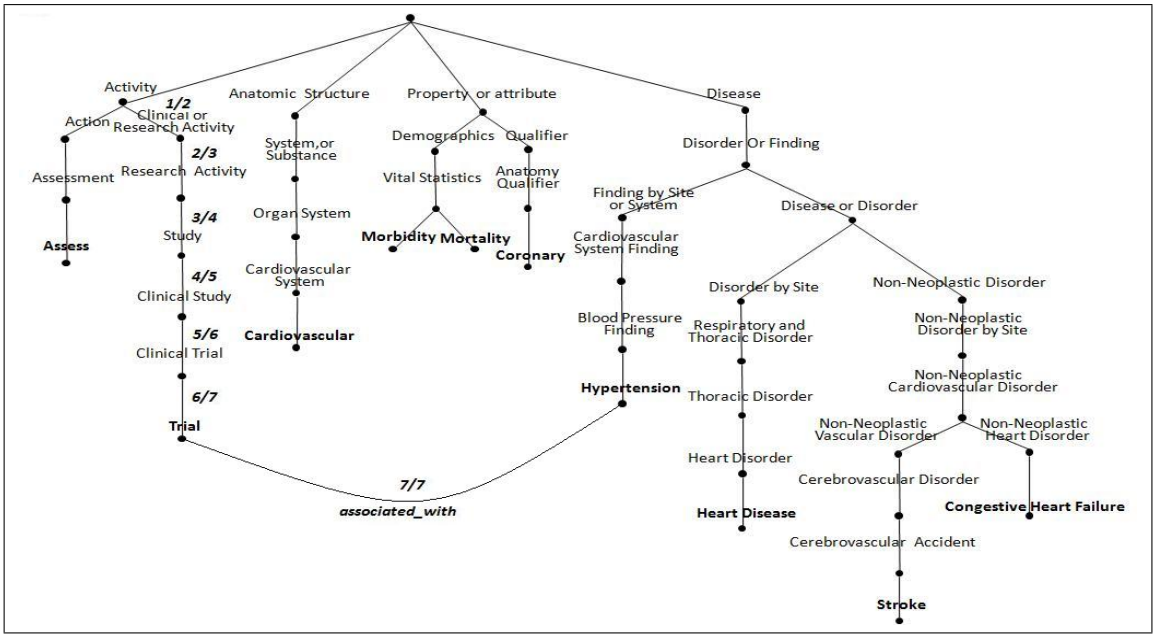


Figure 1: Sentence graph

assigned to that cluster to which they are more connected.

Finally, we assign each sentence to a cluster. To measure the similarity between a cluster and a sentence graph, we use a vote mechanism (Yoo et al., 2007). Each vertex (v_k) of a sentence (O_j) gives to each cluster (C_i) a different number of votes ($p_{i,j}$) depending on whether the vertex belongs to HVS or non-HVS (3).

$$similarity(C_i, O_j) = \sum_{v_k | v_k \in O_j} w_{k,j} \qquad (3)$$

where $\begin{cases} w_{k,j}{=}0 \; si \; v_k \notin C_i \\ w_{k,j}{=}1.0, si \; v_k \in HVS(C_i) \\ w_{k,j}{=}0.5, si \; v_k \notin HVS(C_i) \end{cases}$

4.3 Sentence Selection

The last step consists of selecting significant sentences for the summary, based on the similarity between sentences and clusters. We investigated three alternatives for this step.

- **Heuristic 1:** For each cluster, the top n_i sentences are selected, where n_i is proportional to its size.
- **Heuristic 2:** We accept the hypothesis that the cluster with more concepts represents the main theme in the document, and select the top N sentences from this cluster.
- **Heuristic 3:** We compute a single score for each sentence, as the sum of the votes assigned to each cluster adjusted to their sizes, and select the N sentences with higher scores.

5 Results and Evaluation

In order to evaluate the method, we analyze the summaries generated by the three heuristics over a document[4] from the BioMed Central Corpus, using a compression rate of 20%. Table 1 shows the sentences selected along with their scores.

Although results are not statistically significant, they show some aspects in which our method behaves satisfactorily. Heuristics 1 and 3 extract sentence 0, and assign to it the higher score. This supports the positional criterion of selecting the first sentence in the document, as the one that contains the most significant information. Sentence 58 represents an example of sentence, situated at the end, which gathers the conclusions of the author. In general, these sentences are highly informative. Sentence 19, in turn, evidences how the method systematically gives preference to long sentences. Moreover, while summaries by heuristics 1 and 3 have a lot of sentences in common (9 out of 12), heuristic 2 generates a summary considerably different and ignores important topics in the document. Finally, we have compared these summaries with the author's abstract. It can be observed that heuristics 1 and 3 cover all topics in the author's abstract (see sentences 0, 4, 15, 17, 19, 20 and 25).

[4]BioMed Central: www.biomedcentral.com/content/download/xml/cvm-2-6-254.xml

Sentences	0	4	19	58	7	28	25	20	21	8	43	15
Heuristic 1	99.0	20.0	19.0	18.5	17.0	16.5	16.0	15.5	15.5	13.5	13.5	12.0
Heuristic 2	19.0	16.5	15.5	12.5	12.0	10.5	9.0	9.0	7.5	7.0	7.0	7.0
Heuristic 3	98.8	18.7	17.9	16.3	15.3	14.5	13.4	13.0	13.0	12.7	12.7	12.2

Table 1: Results

As far as heuristic 2 is concerned, it does not cover adequately the information in the abstract.

6 Conclusions and Future Work

In this paper we introduce a method for summarizing biomedical literature. We represent the document as an ontology-enriched scale-free graph, using UMLS concepts and relations. This way we get a richer representation than the one provided by a vector space model. In section 5 we have evaluated several heuristics for sentence extraction. We have determined that heuristic 2 does not cover all relevant topics and selects sentences with a low relative significance. Conversely, heuristics 1 and 3, present very similar results and cover all important topics.

Nonetheless, we have identified several problems and some possible improvements. Firstly, as our method extracts whole sentences, long ones have higher probability to be selected, because they contain more concepts. The alternative could be to normalise the sentences scores by the number of concepts. Secondly, concepts associated with general semantic types in UMLS, as *functional concept*, *temporal concept*, *entity* and *language*, could be ignored, since they do not contribute to distinguish what sentences are significant.

Finally, in order to formally evaluate the method and the different heuristics, a large-scale evaluation on the BioMed Corpus is under way, based on computing the ROUGE measures (Lin, 2004).

Acknowledgements

This research is funded by the Ministerio de Educación y Ciencia (TIN2006-14433-C02-01), Universidad Complutense de Madrid and Dirección General de Universidades e Investigación de la Comunidad de Madrid (CCG07-UCM/TIC 2803).

References

Aronson A. R. Effective Mapping of Biomedical Text to the UMLS Metathesaurus: The MetaMap Program. 2001. *In Proceedings of American Medical Informatics Association*.

Barabasi A.L. and Albert R. Emergence of scaling in random networks. 1999. *Science*,286–509.

Brandow R. and Mitze K. and Rau L. F. Automatic Condensation of Electronic Publications by Sentence Selection. 1995. *Information Processing and Management*,5(31):675–685.

Edmundson H.P. New Methods in Automatic Extracting. 1969. *Journal of the Association for Computing Machinery*,2(16):264–285.

Erkan G. and Radev D. R. LexRank: Graph-based Lexical Centrality as Salience in Text Summarization. 2004. *Journal of Artificial Intelligence Research (JAIR)*,22:457–479.

Kupiec J. and Pedersen J.O. and Chen F. A Trainable Document Summarizer. 1995. *In Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*,68–73.

Lin C-Y. ROUGE: A Package for Automatic Evaluation of Summaries. 2004. *In Proceedings of Workshop on Text Summarization Branches Out, Post-Conference Workshop of ACL 2004, Barcelona, Spain*.

Luhn H.P. The Automatic Creation of Literature Abstracts. 1958. *IBM Journal of Research Development*,2(2):159–165.

Sparck-Jones K. Automatic Summarizing: Factors and Directions. 1999. *I. Mani y M.T. Maybury, Advances in Automatic Text Summarization*. The MIT Press.

Yoo I. and Hu X. and Song I.Y. A coherent graph-based semantic clustering and summarization approach for biomedical literature and a new summarization evaluation method. 2007. *BMC Bioinformatics*,8(9).

Random Graph Model Simulations of Semantic Networks for Associative Concept Dictionaries

Hiroyuki Akama
Tokyo Institute of Technology
2-12-1 O-okayama Meguro-ku
Tokyo 152-8550, Japan
akama@dp.hum.titech.ac.jp

Jaeyoung Jung
Tokyo Institute of Technology
2-12-1 O-okayama Meguro-ku
Tokyo 152-8550, Japan
catherina@dp.hum.titech.ac.jp

Terry Joyce
Tama University
802 Engyo Fujisawa-shi
Kanagawa-ken 252-0805, Japan
terry@tama.ac.jp

Maki Miyake
Osaka University
1-8 Machikaneyama-cho Toyonaka-shi
Osaka 560-0043, Japan
mmiyake@lang.osaka-u.ac.jp

Abstract

Word association data in dictionary form can be simulated through the combination of three components: a bipartite graph with an imbalance in set sizes; a scale-free graph of the Barabási-Albert model; and a normal distribution connecting the two graphs. Such a model makes it possible to simulate the complex features in degree distributions and the interesting graph clustering results that are typically observed for real data.

1 Modeling background

Associative Concept Dictionaries (ACDs) consist of word pair data based on psychological experiments where the participants are typically asked to provide the semantically-related response word that comes to mind on presentation of a stimulus word. Two well-known ACDs for English are the University of South Florida word association, rhyme and word fragment norms (Nelson et al., 1998) and the Edinburgh Word Association Thesaurus of English (EAT; Kiss et al., 1973). Two ACDs for Japanese are Ishizaki's Associative Concept Dictionary (IACD) (Okamoto and Ishizaki, 2001) and the Japanese Word Association Database (JWAD) (Joyce, 2005, 2006, 2007).

While there are a number of practical applications for ACDs, three are singled out for mention here. The first is in the area of artificial intelligence, where ACDs can contribute to the development of intelligent information retrieval systems for societies requiring increasingly sophisticated navigation methods. A second application is in the field of medicine, where ACDs could be used in developing systems that seek to prevent dementia by checking higher brain functions with a brain dock. Finally, within educational settings, ACDs can greatly facilitate language learning through the manifestation of inherent cultural modes of thinking.

The typical format of an ACD is to list the stimulus words (cue words) and their response words together with some statistics relating to the word pairing. The stimulus words are generally basic words determined in advance by the experimenter, while the response words are semantically associated words provided by respondents on presentation of the stimulus word. The statistics for the word pairing include, for example, measured or calculated indices of distance or perhaps some classification of the semantic relationship between the pair of words.

In order to mathematically analyze the structure of ACDs, the raw association data is often transformed into some form of graph or complex network representation, where the vertices stand for words and the edges indicate an associative relationship (Joyce and Miyake, 2007). However, to our knowledge, there have been no attempts at mathematically simulating an ACD as a way of determining in advance the architectural design of a dictionary. One reason is that it is a major challenge to compute maximum likelihood estimations (MLEs) or Monte-Carlo simulations for graph data (Snijder, 2005). Thus, it is extremely difficult to predict dependences for unknown

Coling 2008: Proceedings of 3rd Textgraphs workshop on Graph-Based Algorithms in Natural Language Processing, pages 57–60
Manchester, August 2008

factors such as the lexical distribution across a predetermined and controllable dictionary framework starting simply from a list of basic words. Accordingly, we propose an easier and more basic approach to constructing an ACD model by combining random graph models to simulate graph features in terms of degree distributions and clustering results.

2 Degree distributions for ACDs

2.1 Typical local skew

It is widely known that Barabási and Albert (1999) have suggested that the degree distributions of scale-free network structures correspond to a power law, expressed as $P(x = d) = d^{-r}$ (where d stands for degree and r is a small number, such as 2 or 3). This type of distribution is also known as Zipf's law describing the typical frequency distribution of words in a document and plots on a log scale as a falling diagonal stroke. However, in the degree distribution of ACDs, there is always a local skew, as a local peak or bump with a low hemline. Figure 1 presents two degree distributions; for the IACD (upper) (r = 1.8) and the JWAD (lower) (r = 2.3).

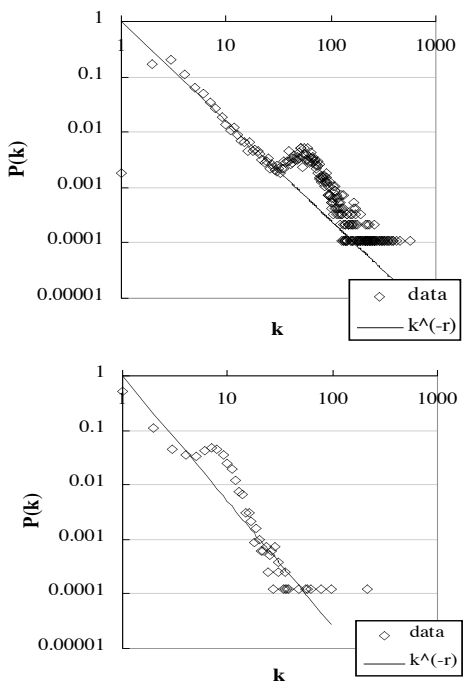


Figure 1. Degree distributions for actual data

The plots indicate a combination of heterogeneous distributions, consisting of a single degree distribution represented as a bell form with a steep slope on the right side. However, what is most interesting here is that throughout the distribution range the curves remain regular and continuous, with an absence of any ruptures or fractures both before and after the local peaks.

When actual ACD data is examined, one finds that as response words are not linked together, almost all the words located in the skewed part are stimulus words (which we refer to as peak words in this study), while the items before the local peak are less frequent response words that have a strong tendency to conform to a decaying distribution. It is therefore relatively natural to divide all word pairs into two types of graph: either a bipartite graph for new response words that are not already part of the stimulus list and a graph that conforms to Zipf's law for the frequencies of response words that are already present in the stimulus list. For the first type, new response words are represented as nodes only with incoming links, generating a bipartite graph with two sets of different sizes. This bipartite graph would exhibit the decaying distribution due to low-frequency response words prior to the local peak. In the second type of graph, response words are represented as nodes with both incoming and outgoing links. This second type is similar to a scale-free graph, such as that incorporated within the Barabási-Albert (BA) model.

2.2 Bipartite Graph and BA Model

A bipartite graph is a graph consisting of vertices that are divided into two independent sets, S and R, such that every edge connects to one S vertex and one R vertex. The graph can be represented by an adjacency matrix with diagonal zero submatrices, where the values of the lower right submatrices would all be zero were it not for the appearances of some stimulus words as response words. The lower right section is exactly where the extremely high degrees of hubs are produced, which far exceed the average numbers of response words.

Thus, we adopt an approach to generating a scale-free graph that reflects Zipf's law for frequency distributions. According to the BA model, the probability that a node receives an additional link is proportional to its degree. Here, we implement the principle of *preferential attachment* formulated by Bollobás (2003):

$$P(x \in N_{x+1}) = m d_t(x) / \sum_{T=1}^{t} d_t(T) \quad (1),$$

with the addition of one condition that is specific to ACDs, which we explain below. The BA model starts with a small number, m_0 of vertices, and at each time step, T, a new vertex with m edges is added and linked to m different vertices that already exist in the graph. N_{t+1} represents a random set of m early vertices, $d_t(i)$ the degree of vertex i in the process at time t. The probability that a new vertex will be connected to a vertex i depends on the connectivity of that vertex, as expressed by Equation (1). However, we specifically assume that m is a random natural number that is smaller than m_0, because in actual data the ratio of stimulus words among all responses words for each stimulus word is obviously far from constant.

Moreover, the graph for the BA model here should be regarded as being a directed graph, because the very reason that hubs emerge within semantic network representations of ACDs is that the number of *incoming* edges is much larger than the expected number of nodes for each possible *in-degree*. In contrast, *out-degree* is limited by the number of responses for each stimulus word i, which is represented as $c(i)$. Let $c(i)$ follow a normal distribution with a mean c_m and a small variance value σ^2 (which is not constant but nearly so) to smoothly combine the distribution of the bipartite graph and the power distribution. If a *directed* adjacency matrix for the network exclusively between stimulus words is expressed as $D(B_{ij})$, then the sum of the non-zero values for each row in a random bipartite graph introducing new response words will be $C(i) - \sum_{i*} D(B_{ij})$ (The vertices of stimulus words with the subscript j are linked with the vertex of the stimulus word i). Thus, new response words—words that are not stimulus words—will be randomly allocated within a bipartite graph according to Equation (2):

$$P((i,l)=1) = r^{-1}(c(i) - \sum_{l*} D(B_{ij})) \qquad (2),$$

where r is the approximate number of such words. Equation (2) will yield the lower left and the upper right sections of the complete adjacency matrix A for the ACD model. The subsequent sub-matrix P^t refers to the transposition of the prior sub-matrix P. The adjacency matrix in Equation (3) represents a pseudo bipartite graph structure where the upper left section is a zero sub-matrix (because there are no interconnections among new response words), but the lower right section is not. Here, B_{ij} (not $D(B_{ij})$, but the undirected counterpart to it), which corresponds to the BA model, is taken as a subsection of the adjacency matrix that must be *non-directed* for the whole composition.

$$A = \begin{pmatrix} O & P^t \\ P & B_{ij} \end{pmatrix} \qquad (3)$$

The key to understanding Equation (3) is to realize that P is conditionally dependent on B_{ij}, because we assume a normal distribution for the number of non-zero values at each row in the lower section of A.

2.3 Simulation Results

Taking into account the approximate numbers of possible new response words, in other words, the balance in sizes between the two sets in the bipartite graph, we built a composition of partial random graphs that could represent an adjacency matrix of the ACD model. Figure 2 presents one of the results obtained for the following conditions:

$t = 90, m_0 = 10, m = 3, c_m = 5, \sigma = 1, r = 3000$.

As the Figure shows, the local peak and the accompanying hemline in the degree distribution are clearly simulated by the complex combination of random graphs.

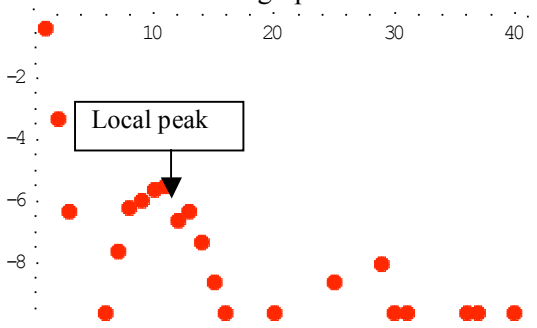


Figure 2. Degree distribution of an ACD model

The degree distribution for the artificial network is consistent with the features observed for actual ACD data, where more than 96% of the stimulus words in each data set are distributed across the peak section of the degree distribution, which is why we have referred to them as peak words. Moreover, it is easy to verify that without the assumption of a normal distribution for $c(i)$, distinct fractures emerge in the artificial curve where new response words in the bipartite struc-

ture would be distinguished from stimulus words located at initial points of the local peak.

3 Markov Clustering of ACDs

3.1 MCL

This section introduces the graph clustering method that is applied to both the real and artificial ACD data in order to compare them. Markov Clustering (MCL) proposed by Van Dongen (2001) is well known as a scalable unsupervised cluster algorithm for graphs that decomposes a whole graph into small coherent groups by simulating the probability movements of a random walker across the graph. It is believed that when MCL is applied to semantic networks, it yields clusters of words that share certain similarities in meaning or appear to be related to common concepts.

3.2 MCL Results

The clustering results for the ACD model created by combining random graphs reveal that each of the resultant clusters contains only one stimulus word surrounded by several response words. This result is somewhat strange because there are dense connections between stimulus words, which would lead us to assume that clusters would have multiple stimulus word. However, the results of applying MCL clustering to the graph for the ACD model are in reality highly influenced by the sub-structure of the bipartite graph and less dependent on the scale-free structure.

Nevertheless, the result is quite similar to results observed with real data. On examining MCL clustering results for different ACD semantic networks, we have observed that MCL clusters tend to consist of one word node with a relatively high degree and some other words with relatively low degrees. On closer inspection of the graph, it is possible to see several supporter nodes that gather around one leader node, forming a kind of small conceptual community. This suggests that the highest degree word for each cluster becomes a representative for that particular cluster consisting of some other low degree words. In short, MCL clustering is executed based on such high degree words that tend to have relatively low curvature values (Dorow, 2005) compared to their high average degree values.

4 Conclusion

In this paper, we have proposed a basic approach to simulating word association dictionary data through the application of graph methodologies. This modeling is expected not only to provide insights into the structures of real ACD data, but also to predict, by manipulating the model parameters, possible forms for future ACDs. Future research will focus on constructing an exponential random graph model for ACDs based on Markov Chain Monte Carlo (MCMC) methods.

References

Barabási, Albert-László and Réka Albert. 1999. *Emergence of scaling in random networks*, Science. 286:509-512.

Bollobás, Béla. 2003. *Mathematical Results on Scale-free Random Graphs*, http://www.stat.berkeley.edu/~aldous/Networks/boll1.pdf

Dorow, Beate et al. 2005. *Using Curvature and Markov Clustering in Graphs for Lexical Acquisition and Word Sense Discrimination*, MEANING-2005,2nd Workshop organized by the MEANING Project, February,3rd-4th.

Joyce, Terry and Maki Miyake. 2007. *Capturing the Structures in Association Knowledge: Application of Network Analyses to Large-Scale Databases of Japanese Word Associations*, Large-Scale Knowledge Resources. Construction and Application, Springer Verlag:116-131.

Kiss, G.R., Armstrong, C., Milroy, R., and Piper, J. 1973. *An associative thesaurus of English and its computer analysis*, In Aitken, A.J., Bailey, R.W. and Hamilton-Smith, N. (Eds.), The Computer and Literary Studies, Edinburgh University Press.

Nelson, Douglas L., Cathy L. McEvoy, & Thomas A. Schreiber. 1998. *The University of South Florida word association, rhyme, and word fragment norms*, Retrieved August 31, 2005, from http://www.usf.edu/FreeAssociation

Okamato, Jun and Shun Ishizaki. 2001. *Associative Concept Dictionary and its Comparison Electronic Concept Dictionaries*. PACLING2001-4th Conference of the Pacific Association for Computational Linguistics:214-220.

Snijders, Tom A.B., Philippa E. Pattison, Garry L.Robins, Mark S. Handcock, 2005. *New Specifications for Exponential Random Graph Models*, http://stat.gamma.rug.nl/SnijdersPattisonRobinsHandcock2006.pdf

Steyvers, Mark and Josh Tenenbaum. 2005. The Large-Scale Structure of Semantic Networks, Statistical Analyses and a Model of Semantic Growth, Cognitive Science. 29 (1):41-78.

Author Index