Natural Language Engineering **18** (3): 313–342.
doi:10.1017/S1351324911000210

Exploring patterns in dictionary definitions for synonym extraction

TONG WANG and GRAEME HIRST

Department of Computer Science, University of Toronto, Toronto, ON M5S 3G4, Canada
e-mail: {tong, gh}@cs.toronto.edu

(*Received 7 April 2010; revised 4 March 2011; accepted 8 April 2011; first published online 11 July 2011*)

Abstract

Automatic determination of synonyms and/or semantically related words has various applications in Natural Language Processing. Two mainstream paradigms to date, lexicon-based and distributional approaches, both exhibit pros and cons with regard to coverage, complexity, and quality. In this paper, we propose three novel methods—two rule-based methods and one machine learning approach—to identify synonyms from definition texts in a machine-readable dictionary. Extracted synonyms are evaluated in two extrinsic experiments and one intrinsic experiment. Evaluation results show that our pattern-based approach achieves best performance in one of the experiments and satisfactory results in the other, comparable to corpus-based state-of-the-art results.

1 Introduction

Synonymy is one of the lexical semantic relations (LSRs), which are the relations between meanings of words. By definition, synonyms are 'one of two or more words or expressions of the same language that have the same or nearly the same meaning in some or all senses' (Mish 2003). Despite its importance in various Natural Language Processing (NLP) applications (Mohammad and Hirst 2006; Bikel and Castelli 2008; Mandala, Tokunaga, and Tanaka 1999), the task of synonym extraction remains challenging without satisfactory results in the NLP community.

In this paper, we propose three novel approaches to extracting synonyms from dictionary definitions. In contrast to many existing approaches that try to extract synonyms from free texts (*distributional* methods), our dictionary-based model has the advantage of being computationally efficient, resource-lean, and easily adaptable to various domains of a language or even across languages. In particular, our methods yield the best result for one of the extrinsic evaluations among lexicon-based methods reported to date; in other experiments, we also achieve satisfactory results comparable to some of the state-of-the-art distributional approaches.

We will start with motivations for synonym extraction in Section 1.1, where applications of automatically extracted synonyms are presented. A review of related work on synonym extraction follows in Section 1.2. Details of the proposed approaches will be elaborated in Section 2, followed by evaluation experiments

and results in Section 3. Section 4 will conclude the paper with discussion of possible extensions to the current study.

1.1 Synonymy and its applications

In contrast to other LSRs, synonymy closely associates different lexicalizations of the same concept, which is a unique and useful property in many NLP applications. Mohammad and Hirst (2006), as one of the many successful examples, conflated words into concepts (represented by thesaurus categories) when exploring their cooccurrence patterns. The resulting concept–concept matrix is compacted to approximately 0.01% the size of the lexical cooccurrence matrix and has proven very effective in measuring lexical semantic similarity. Bikel and Castelli (2008) applied synonyms in their event matching system in which each lexical item is augmented by its synonyms from a thesaurus. If a surface-form match fails, the two-tier model will back off to a synonym match to improve coverage at a low cost in accuracy. In information retrieval, *query expansion* can help improve search results by substituting and expanding user queries with synonyms (Mandala *et al.* 1999), since the wording for the same topic varies greatly among users (e.g., *car broker* versus *auto dealer*).

Thesauri are obviously the most common sources for synonyms (e.g., Roget 1911; Fellbaum 1998). While such hand-crafted resources usually guarantee high quality of synonymy among entries, thesauri also exhibit various limitations when used in NLP applications, such as the amount of human effort involved in building a thesaurus, fixed domain coverage, and limited availability. Mandala *et al.* (1999) argued that when used in query expansion, manually constructed thesauri exhibit various problems, such as low convergence and domain incompatibility with the document collection in question. Their study showed that different types of thesauri, manually or automatically constructed, all have their own limitations, but IR system performance is almost doubled when different sources of synonymy are combined, indicating the necessity for automated processes for synonym extraction.

Another application related to synonym extraction is lexical substitution (McCarthy and Navigli 2009), which is useful for many NLP-related tasks, such as question answering, summarization, and paraphrase acquisition. For given instances of words with contexts, synonyms particular to the senses of words in the given contexts are suggested and compared to answers elicited from humans. Thus, in addition to extracting synonyms for a given word, a lexical substitution system must also disambiguate senses of the word according to different contexts,[1] which is a challenge beyond the scope of this study.

1.2 Related work

1.2.1 Distributional approaches

Despite the seemingly intuitive nature of synonymy, it is far from trivial to identify from free text, since synonymous relations, unlike other LSRs, are established more

[1] Similar studies that consider disambiguation using contexts also include that of Shimohata and Sumita (2002).

often by semantics than by syntax. Hyponyms, for example, can be extracted fairly accurately with the syntactic pattern '*A*, such as *B*' (Hearst 1992). As in the sentence 'The bow lute, such as the Bambara ndang, is plucked and . . .', 'Bambara ndang' is a type of (and thus a hyponym of) 'bow lute'. However, there seem to be few, if any, patterns that synonyms tend to follow. Imagine if we contrive to extend the above heuristics for synonym extraction, for example, by taking *B* and *C* as synonyms according to pattern '*A*, such as *B* and *C*'. On a closer examination, the semantic closeness between *B* and *C* is determined by the semantic specificity of *A*, i.e., the more general *A* is in meaning, the more unlikely *B* and *C* are synonyms. This is easy to see from the following excerpt from the British National Corpus in which this rule would establish a rather counterintuitive synonymy relationship between *oil* and *fur*:

> . . . an agreement allowing the republic to keep half of its foreign currency-earning production such as *oil* and *furs*.

Another intuitive approach to extracting synonyms from free texts is to find words sharing similar contexts, under the *distributional hypothesis* that 'similar words tend to have similar contexts' (Harris 1954). Such assumptions, however, are necessary but not sufficient for characterizing synonymy, since there is a fine but distinct line between being similar and being synonymous. In fact, words with similar contexts can represent many LSRs other than synonymy, even including antonymy (Mohammad, Dorr and Hirst 2008). In the work by Lin (1998), for example, the basic idea is that two words sharing more syntactic relations with respect to other words are more similar in meaning. Syntactic relations between word pairs were captured by the notion of *dependency triples* (e.g., (w_1, r, w_2), where w_1 and w_2 are two words and r is their syntactic relation). Semantic similarity is well captured, but this is not equivalent to synonymy (it is shown in Lin's paper that antonyms are abundant in the list for the most similar word pairs).

To address the issue of false positives, Lin *et al.* (2003) devised two methods for identifying antonyms from the related words extracted using the algorithm by Lin (1998). The pattern-based method assumed X and Y to be antonyms if they appeared in patterns, such as *from X to Y* or *either X or Y*. The bilingual dictionary-based method was based on the observation that translations of the same word are usually synonyms. In comparing the classification between synonyms and antonyms to a randomly selected set of synonyms and antonyms from the *Webster's Collegiate Thesaurus* (Mish 2003), performance of the pattern-based method was generally good ($F = 90.5\%$) but the dictionary-based method has very low recall (39.2%) due to the limited coverage of bilingual dictionaries used. In addition, the model can only identify antonyms among the various LSRs mixed in the extraction result.

Several later variants followed the work of Lin (1998). Hagiwara (2008), for example, also used the concept of dependency triples and extended it to *syntactic paths* in order to account for longer dependencies. *Pointwise total correlation* was used as the association ratio for building similarity measures, as opposed to the *pointwise mutual information* used by Lin (1998). Wu and Zhou (2003) used yet another measure of association ratio, i.e., *weighted mutual information* in the same

distributional approach, claiming that weighted mutual information could correct the biased (lower) estimation of low-frequency word pairs in pointwise mutual information.

Some studies also use the syntactic information in context indirectly in synonym extraction. Curran (2002), for example, compiled dependency relations into *contextual vectors*, which were used in a *k-nearest neighbor* model with ensembles to identify synonyms from raw texts. Extraction results were evaluated on a combined thesaurus in a manner similar to our own experiment in Section 3.3 below, but differences in details make the results less commensurate than those we compare with in Section 3.3.

Multilingual approaches can also be found in later studies (Barzilay and McKeown 2001; Shimohata and Sumita 2002; Van der Plas and Tiedemann 2006), hypothesizing that 'words that share translational contexts are semantically related'; the details of these approaches, however, differ in several important ways, such as the resource for computing translation probabilities and the number of languages involved. Wu and Zhou (2003), for example, proposed a semantic similarity model based on *translation probability* (which is calculated from a bilingual corpus) for synonym extraction. Resulting synonym sets are compared to an existing thesaurus (*EuroWordNet*), a setting similar but not comparable to that of Van der Plas and Tiedemann (2006), since both the corpora and the gold standards are different in these two studies.

Another example of distributional approaches is that of Freitag *et al.* (2005), where the notion of context is simply word tokens appearing within windows. Several probabilistic divergence scores were used to build similarity measures and the results were evaluated by solving simulated TOEFL synonym questions, the automatic generation of which is itself another contribution of the study.

1.2.2 Lexicon-based approaches

Lexicons can be viewed as uniquely structured texts, which associate a word with other words that define it. Since lexicons are usually much smaller than text corpora even of modest sizes, it is computationally less expensive to be used in graph-based methods. Compared to free text, definition texts also exhibit stronger structural and syntactic regularity, which allows simple rule-based methods to achieve reasonably good performance.

Dictionaries are among the popular lexicons used for synonym extraction. In the early 1980s, extracting and processing information from machine-readable dictionary definitions was a topic of considerable interest, especially since the *Longman's Dictionary of Contemporary English* (or LDOCE, Procter 1978) had become electronically available. Two special features were particularly helpful in promoting this dictionary's importance in many lexicon-based NLP studies. First, the dictionary uses a *controlled vocabulary* of only 2,178 words to define approximately 207,000 lexical entries. Although the lexicographers' original intention was to facilitate the use of the dictionary by learners of the language, this design later proved to be a valuable computational feature. Second, the *subject code* and

box code label each lexical entry with additional semantic information, such as the domains of usage and selectional preferences/restrictions.

It is debatable whether a learner's dictionary is indeed more suitable for the purpose of machine-based learning for NLP. A controlled vocabulary can also complicate the definition syntax, since there is usually a trade-off between the size of the defining vocabulary and the syntactic complexity of definitions (Barnbrook 2002). Nonetheless, with all the computationally friendly features, LDOCE soon attracted significant research interest. Boguraev and Briscoe (1989) covered various topics in using this machine-readable dictionary, from rendering easier on-line access and browsing (which involved many engineering challenges under the computing environments of the time) to semantic analysis and utilization of the definition texts. The latter is of great relevance to the topics discussed in this paper.

Alshawi (1987) (included in Boguraev and Briscoe 1989) conducted a phrasal analysis of LDOCE definitions by applying a set of successively more specific phrasal patterns on the definition texts. The goal was to mine semantic information from definitions, which is believed to be helpful in 'learning' new words with the knowledge of the controlled vocabulary in LDOCE. Guthrie *et al.* (1991) exploited both the controlled vocabulary and the *subject code* features. The controlled vocabulary was firstly grouped into 'neighborhoods' according to their cooccurrence patterns; the subject codes were then imposed on the grouping, resulting in so-called *subject-dependent neighborhoods*. Such cooccurrence models were claimed to better resemble the polysemous nature of many English words, which, in turn, could help improve word sense disambiguation performance. Unfortunately, no evaluation has ever been published to support this claim.

The work of Chodorow, Byrd and Heidorn (1985) is an example of building a semantic hierarchy by identifying 'head words' (or *genus terms*; see Section 2.1) within definition texts. The basic idea is that genus terms are usually hypernyms of the words they define. If two words share the same head word in their definitions, they are likely to be synonymous siblings under the same parent in the lexical taxonomy. Thus, by grouping together words that share the same hypernyms, not only are synonyms extracted from the definition texts, but they are also, at the same time, organized into a semantic hierarchy.

Particularly, in recent years, one popular paradigm is to build a graph on a dictionary (a *dictionary graph*) according to the defining relationship between words. Vertices correspond to words, and edges point from the words being defined (*definientia*) to words defining them (*definiens*). This idea was first developed by Reichert, Olney and Paris (1969) and has ever since been extensively exploited by later studies as a basis for building dictionary graphs. Given such a dictionary graph, many results from graph theory can then be employed to explore synonym extraction. Blondel and Senellart (2002) applied an algorithm on a weighted graph similar to PageRank (Page *et al.* 1999); weights on the graph vertices would converge to numbers indicating the relatedness between two vertices (words), which are subsequently used to define synonymy. Muller, Hathout and Bruno (2006) built a Markovian matrix on a dictionary graph to model random walks between vertices, which is capable of capturing the semantic relations between words that are not

immediate neighbors in the graph. Ho and Cédrick (2004) employed concepts in information theory, computing similarity between words by their *quantity of information exchanged* through the graph.

Apart from synonym extraction, dictionary graphs have also been applied to other NLP-related tasks, such as word sense disambiguation. Navigli (2009), for example, proposed to disambiguate definiens in dictionary definitions by (quasi-) circle-based scores computed from a dictionary graph. In a broader sense, the topic of a *Wikipedia* article can also be regarded as being 'defined' by the article content, motivating studies, such as Gabrilovich and Markovitch (2007), to develop semantic relatedness metrics based on 'dictionary graphs' using this type of defining relationship.

2 Synonym extraction methods

2.1 Properties of dictionary definitions

We now introduce some lexicographical properties of dictionary definitions as a basis for our extraction methods. We will use some lexicographical terminology: the word being defined in a dictionary entry is called the *definiendum*, and words that define it are called its *definientia*. This subsection discusses special features of these elements in monolingual English dictionaries that facilitate synonym extraction.

Within the definientia, there is usually a word or phrase that is more closely related to the definiendum than the rest of the definition: the *genus term*. Usually, genus terms are either synonyms or hypernyms of the definiendum, as in the example of *automobile*: a motor *car* (synonym) and *summer*: the second and warmest *season* of the year (hypernym). Sometimes a genus term may include a quantifier. Such terms are known as *empty heads*; they preclude simple heuristics for identifying genus terms. An example is the word *type* in the definition *arum*: a tall, white *type* of lily (Guthrie *et al.* 1990). Identifying genus terms or empty heads is itself a useful application in processing dictionary definitions. Nonetheless, the composition of definientia usually exhibits great regularity in terms of syntax, style, and sometimes, vocabulary. Amsler (1980) showed that definitions of nouns and verbs in most dictionaries follow rigid stylistic patterns. In fact, this stylistic regularity goes beyond the definitions of nouns and verbs; as shown in Section 3.3, definitions of adjectives exhibit comparable or even greater regularity than those of nouns.

In monolingual English dictionaries, definition texts can often be decomposed into two parts: the *interpretive part* and the *synonymous part*. The former usually leads a definition in as a relatively lengthy description of the definiendum with simple vocabulary but complex syntax; many of these interpretive parts are followed by one or more synonymous parts, each consisting of a single word or phrase highly synonymous to the definiendum. When appearing together in the same definition, the two parts are usually separated by a special typographical format (e.g., capitalization) or delimiter (e.g., semicolons). Examples include the definition of *look* in Mish (2003) 'to exercise the power of vision upon: EXAMINE', and of *looker-on*: 'one who looks on; a spectator'. Note that in the latter case, the synonymous part after

the semicolon is, instead of being just one word, prefixed by an indefinite article. In terms of length and vocabulary, both examples conform with our observation on the differences between the interpretive and the synonymous parts.

For the interpretive part of a given definition, real synonyms, if any, could be identified only through syntactic and semantic knowledge of the definientia; many attempts have been made to automate such deep analysis of the definition language. It is the simpler cases of the synonymous parts in definitions that are mostly left uninvestigated. Semantically, as is shown in the previous examples, such parts of definitions are highly synonymous to the definienda, whereas terms extracted from the interpretive parts can often be hypernyms (even after successfully avoiding empty heads). Syntactically, synonymous parts can be identified by very simple typographical patterns. Although the patterns are dictionary-specific, their rigid nature usually necessitates minimal human intervention. The semantic relatedness between definienda and definientia validates the hypothesis that synonymy does exist in dictionary definitions, while the regularities in the composition of definientia make it possible to develop algorithms for synonym extraction.

2.2 Inverted index extraction

In this section, we propose a simple baseline algorithm as a first attempt to explore the relationship between definienda and definientia – by building an *inverted index* (hence the name *Inverted Index Extraction* (*IIE*)) on the dictionary.[2] Each line $l = (t, S)$ in the inverted index consists of a *target word* t, of which we want to extract the synonyms, followed by a set $S = \{w : t \in \text{dfn}(w)\}$ of words with t in their definition texts. Here, $\text{dfn}(w)$ refers to the set of words in the definientia of the word w. In IIE, such words are considered semantically related to t, regardless of the importance of t within their definitions. Consequently, semantic relatedness in IIE is built upon the occurrence of one word in the definition of another.

Table 1 shows the IIE result for *look*. Many near-synonyms of the target word *look* are successfully identified: *see* and *watch* as well as troponyms, such as *ogle*, *gaze*, *inspect*, and *glance*. Observe that words, such as *gawp* or *rubberneck*, though as synonymous to *look* as *scrutinize* and *glance*, are not listed in existing thesauri (e.g., *Roget's Thesaurus*, Roget 1911), indicating that the results of IIE could indeed help improve the coverage of existing thesauri. More interestingly, when using *The Macquarie Dictionary* (of Australian origin), colloquial Australian expressions denoting the action of *looking* (such as *dekko* and the rhyming slang *Captain Cook*) appear in the list, which are missing from the results using *Webster's Revised Unabridged Dictionary* (or *WRUD*, of American origin). This partly provides evidence for the claim made in previous studies (e.g., that of Ho and Cédrick 2004) that lexicon-based methods for synonym extraction exhibit portability between different domains and languages. To fully test the claim, however, requires comparison between synonyms extracted from a domain- (or language-)specific dictionary and those compiled in

[2] Similar techniques have also been applied to other resources than dictionary, such as *Wikipedia* (Gabrilovich and Markovitch 2007).

Table 1. *IIE result for the target word* look

air	disdain	get an eyeful of	keep house	retrospect
an optic at	district nurse	get on to	la femme	review
appear	double take	gink	lamp	rubberneck
appearance	easy on the eye	give	leer	scowl
aspect	envisage	glance	lemma	scrutiny
at	evil	glare	letter bomb	search
await	expect	gleam in one's	light-pen	see
babysit	explore	eye	lo	shoofty
bad hair day	expression	glimpse at	load of	show
beam	eye	gloom	look	skew
behold	eyeball	glower	look daggers at	smile
bend one's gaze	eyehole	goggle	lour	snapdragon
on (or upon)	eyesore	Goth	mind	sneer
blink	eyewink	green	nurse	snorkel
blink at	face	have	nut	speck
butchers	face as long as a	have a perv	ogle	squint
candid camera	fiddle	have a screw	optimism	squiz
Captain Cook	faraway	have a sticky	Orpheus	stare
care for	fascinate	have eyes only	overlook	sticky
check	fleer	for	oversee	stony
cherchez	flight control	health visitor	peek	tend
command	front	hold	peep	(the) devil take
contemplate	frown	hook and eye	peer	the hindmost
cook	gander	hope	perv	treat
cop	gawp	horror	phenotype	twig
countenance	gaze	hunt up	pout	view
crane	geek	independent	pry	watch
dekko	get	inspect	quiz	withering
despise		introspect	regard	

a thesaurus in the same domain (or language), which is beyond the scope of this study.

In contrast to the method used by Chodorow *et al.* (1985), who distinguished genus terms from the rest of definition texts, IIE takes all definiens indiscriminately and thus defines a less-strict relatedness. The word *independent*, for example, is related to *look* by its definition '`sufficient to support someone so that they do not have to look for a living`', and *optimism* by '`tendency to look on the bright side of things`'. This is especially problematic for common words, which are more likely to appear in the definition texts of (and thus be related to) many other words. A simple method to deal with false positives of this kind is to specify a part of speech (POS) for each target word. If, for example, we are interested in finding synonyms for the *verb look*, we might want to discard *faraway* : `abstracted or dreamy, as a look` from the IIE result, since we know that the entry *faraway* is an adjective and thus should not be a synonym of *look* (in its verb sense).

Another way to reduce the false positive rate is based on local connectivity of the dictionary graph. The word *fear*, for example, appears in many other words' definitions; among these, many with the suffix *-phobia* are essentially hyponyms rather than synonyms of *fear*. Such words, in contrast to synonyms of *fear*, do not seem to appear in each other's definition (i.e., disconnected in the dictionary graph). Using such difference in connectivity in the dictionary graph, we obtained the two subtables in Table 2.

Table 2. *Discriminating words by connectivity for the target word fear*

(a) *Connected words*

affright	bugbear	funk	shock-horror
alarm	doubt	hair-raiser	shudder
angst	Demogorgon	horror	terror
apprehension	dread	nightmare	thing
awe	emotion	passion	
bugaboo	fright	phobia	

(b) *Isolated words*

aerophobia	courage	Negrophobia	shy
agoraphobia	ergophobia	nosophobia	squeal
Anglophobia	erythrophobia	nyctophobia	superstition
aquaphobia	foetal position	ochlophobia	technophobia
arachnophobia	gasp	pallor	terrorism
attrition	gynophobia	panic	thanatophobia
Bayard	hobgoblin	parliamentary	toxiphobia
biopanic	homophobia	privilege	tremor
blue funk	horripilation	perfect contrition	triskaidekaphobia
cancerophobia	jealousy	persecution complex	wheyface
castration complex	lyssophobia	psychasthenia	xenophobia
coprophobia	necrophobia	scare	zoophobia

It is interesting to see how words in the subtable on top tend to be more synonymous to *fear* than those on the bottom: all hyponyms with the suffix *-phobia* are eliminated, the only antonym (*courage*) is also eliminated, and most synonyms remain in the upper subtable (including *phobia* as a word instead of a suffix).

2.3 Pattern-based extraction

As stated earlier, the number of synonyms extracted by IIE largely depends on the frequency of the target word and is severely diminished if the target word is rare and thus less likely to appear in other words' definitions. Consequently, a new extraction strategy is proposed in this section to alleviate this problem. Specifically, for a given word *w*, the proposed approach tries to identify synonyms within the definition text of *w* according to certain patterns; we refer to this as *Pattern-based Extraction* (PbE). As a result, the frequency of *w* no longer matters, as long as its definition matches any of the extraction patterns. As is true with all existing lexicon-based methods, coverage remains an issue for PbE due to the limited patterns one can find in definition texts, but evaluation on all tasks in Section 3 shows significant improvement on PbE's coverage over that of IIE. The patterns in question are based on the synonymous parts of definitions discussed in Section 2.1. We aim at capturing the features distinguishing such parts from the rest of the definition texts.

In the distribution of definition text lengths of *The Macquarie Dictionary* and *WRUD* (Figure 1), there is a large number of very short definitions: 5,359 consisting of only one word and 9,672 of two words in *The Macquarie Dictionary*, 4,918

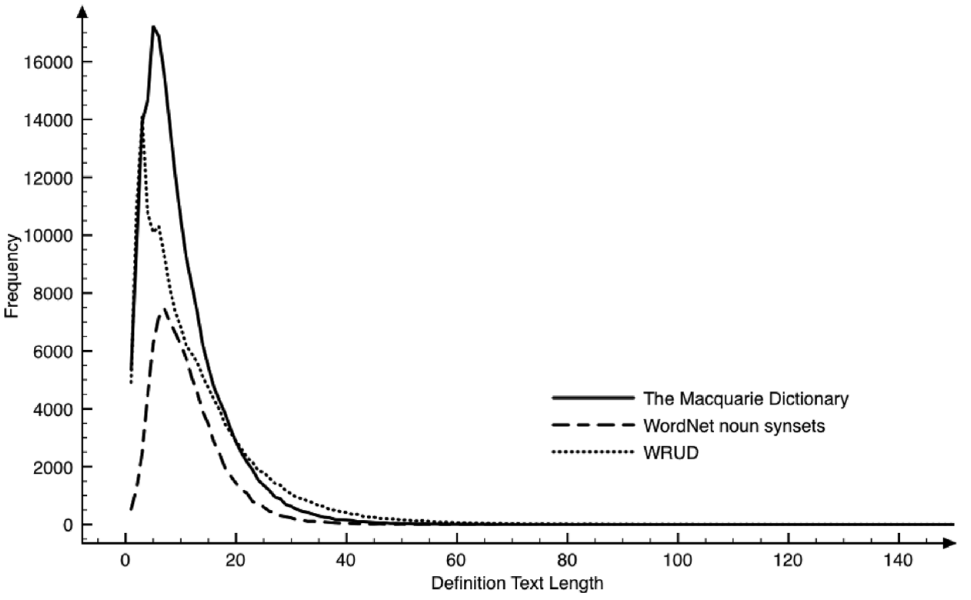


Fig. 1. Distribution of definition text length in *The Macquarie Dictionary*.

and 10,938, respectively, in *WRUD*.[3] The noun synset subset of *WordNet* has different absolute numbers but the distribution is very similar. Some of the one-word definitions are synonyms of the definienda, the other being expansions of abbreviations being defined (which could also be viewed as synonyms in a way). The two-word definitions are more complicated. Sometimes a synonym follows a function word, such as indefinite article in a noun definition (e.g., `jailhouse: a jail`) or infinitive *to* (e.g., `damask: to damascene`) in a verb definition. Sometimes both words are synonyms to the definiendum and are separated by a semicolon in between (e.g., `maculate: spotted; stained`). These two cases constitute a large proportion of the double-word definitions and are both useful for synonym extraction. There are also cases of a hypernym of the definiendum following a certain quantifier (e.g., `madrigal: any song`), with the genus terms being hypernyms rather than synonyms of the definienda.

Note that despite the seemingly large numbers of one- and two-word definitions, they are still far from dominant, considering the size of most dictionaries; the majority of definitions are combinations of synonymous and interpretive parts; sometimes synonymous parts are not present at all. It is therefore necessary to discover patterns that can deal with longer and compounded definitions. Also, not all dictionaries have specific patterns for synonyms in definition texts. Definition texts in *WordNet*, for example, group together synonyms in the form of *synsets*. All

[3] There are originally 27,237 two-word definitions in *WRUD* but a large number of them are inflected forms (e.g., *feared* defined as *of Fear*). Without counting definitions in this form, the number of two-word definitions is then 10,938 (shown in Figure 1).

Table 3. *Notation for regular expressions used in this paper*

`^`	Beginning of line	`$`	End of line
`\w`	Any word	`.`	Any character
`()`	Group of one or more characters	`(?:)`	Unindexed group
`?`	Matching zero or one of the previous character or group	`+`	Matching one or more of the previous character or group
`*`	Matching any number of the previous character or group	`\`	Escape the next character
`|`	Either the previous or the next character or group		

Algorithm 1 Simple Pattern-based Extraction

```
1: resultSet ← {} //holding the extracted synonyms
2: targetWordsSet ← {targetWord} //holding target words for each iteration
3: repeat
4:    newResultsSet ← {} //holding new result from each iteration
5:    for all w in targetWordsSet do
6:       for all def in definitions of w do
7:          for all p in PbEPatterns do
8:             if def matches p on s then
9:                newResultsSet ← temp ∪ {s}
10:            end if
11:         end for
12:      end for
13:   end for
14:   resultSet ← resultSet ∪ newResultsSet//update result set
15:   targetWordsSet ← newResultsSet − targetWordsSet//update target words
16: until newResultsSet is empty//until no new synonyms are extracted
```

of the general-purpose dictionaries we examined[4] explicitly list synonymous parts in their definition texts.

2.3.1 The basic PbE algorithm

Our pattern-based method (Algorithm 1) discovers occurrence patterns of synonyms in definition texts. Given a set of patterns $P = \{p_1, \ldots, p_n\}$, PbE looks at each definition of a target word w and extracts words that follow any one of the patterns as synonyms. In practice, a pattern p_i takes the form of a regular expression, e.g., '`^.*;(\w+).$`'. Table 3 shows the notation for regular expressions used in this paper. If a definition text matches this pattern, the word s corresponding to the regex group '`(\w+)`' will be proposed as a synonym. For example, if the target word is $w =$'*separate*', then one of its definitions '`separate: to disconnect; disunite`' matches a pattern $p =$'`^.*; (\w+)$`' and $s =$'*disunite*' is proposed as a synonym to w. This scenario is to be referred to as '**a definition matching a pattern** p **on a word** s', as in Line 1 in Algorithm 1.

[4] Besides *The Macquarie Dictionary* and *WRUD*, we also examined *Webster's New Collegiate Dictionary*, *LDOCE*, *Merriam Webster Online*, *thefreedictionary.com*, and *wiktionary.org*.

Algorithm 2 Pattern-based Extraction with IIE-style Scanning

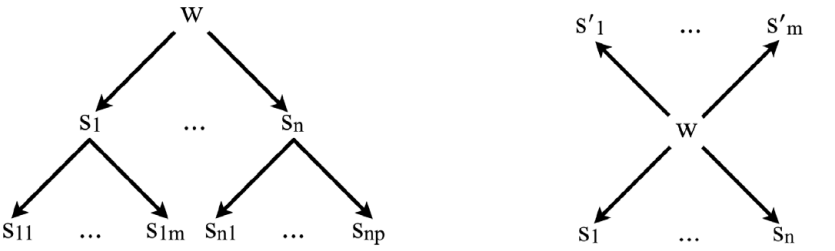
```
1: resultSet ← {}
2: targetWordsSet ← {targetWord}
3: repeat
4:    newResultsSet ← {}
5:    for all w in targetWordsSet do
6:       for all definition d of w do
7:          for all p in PbEPatterns do
8:             if d matches p on s then
9:                newResultsSet ← newResultsSet ∪ {s}
10:            end if
11:         end for
12:      end for
13:      for all w′ in Dictionary do
14:         for all definition d′ of w′ do
15:            for all p in PbEPatterns do
16:               p_s ← plug_w_into_p(p, w)
17:               if d′ matches p_s on w then
18:                  //target word recognize as synonym of w′
19:                  newResultsSet ← newResultsSet ∪ {w′}
20:               end if
21:            end for
22:         end for
23:      end for
24:   end for
25:   resultSet ← resultSet ∪ newResultsSet//update result set
26:   targetWordsSet ← newResultsSet − targetWordsSet//update target words
27: until newResultsSet is empty
```

2.3.2 Incorporating IIE into PbE

As mentioned earlier, there are many definitions that do not contain a synonymous part, which again brings up the problem of diminished sizes of proposed synonym sets and, thus, low coverage of the extraction strategy. To address these two issues, after matching the definientia of w against the patterns, an improved version of PbE (Algorithm 2) scans the entire dictionary and looks at the definientia of other words; if, in this step, any word w' has a definition matching any pattern on w, then w' is extracted as a synonym to w. For example, for the target word $w =$'*separate*', PbE first proposes $s =$'*disunite*' as a result of Algorithm 1; in addition to this, as a result of Line 2 through 2 in Algorithm 2, PbE scans for definitions matching any patterns on the target word $w =$'*separate*'. To do so, PbE first plugs w into the pattern p, resulting in $p_s =$'`^.*; separate$`', and then finds and proposes the word $w'=$'*part*' whose definition '`part: to put or keep asunder...; disunite; separate`' matches the plugged-in pattern p_s on '*separate*' (Line 2, Algorithm 2).

The process of scanning other words' definitions resembles that of IIE in Section 2.2, with a difference that now the algorithm makes distinctions on where and how w appears in the definition of w'.

On top of the synonym set *resultSet* extracted by Algorithm 1 for a target word w, Algorithm 2 can improve the coverage of PbE by the additional pass through the dictionary. Similar improvement can also be achieved by running Algorithm 1 for multiple iterations, but the quality of extracted synonyms will vary. Repeated execution of Algorithm 1 will result in a tree-like growth pattern for synonyms as shown in Figure 2(a), where every element in $S = \{s_1, \ldots, s_n\}$ will serve as a target



(a) Multi-iteration of Algorithm 1

(b) IIE-resembling Scanning in Algorithm 2

Fig. 2. Different growth patterns of result set size in Algorithm 1 and 2.

word; some of the elements in S (e.g., s_1), however, are related to the target word under rare senses, for example, and the offspring of these elements (e.g., $\{s_{11}, \ldots, s_{1m_1}\}$) will be far less synonymous to the original target word w (more details on this in Section 2.3.4). In contrast, the IIE-style procedure in Algorithm 2 starts from the target word w instead of the elements in the extracted set S, and the resulting output of PbE grows 'upwards' as in Figure 2(b). It is obvious that words in $\{s'_1, \ldots, s'_m\}$ are more synonymous to w than those in $\{s_{i1}, \ldots, s_{im_i}\}, i = 1, \ldots, n$.

Here, a pass through the entire dictionary is required for extracting synonyms for every word, but various preprocessing steps can be used to improve the efficiency. For example, the search space could be reduced to only those definitions that follow at least one of the synonymous part patterns, which is less than 16% of the size of the entire dictionary. Algorithm 2 is used for PbE in all the experiments in Section 3.

2.3.3 Pattern bootstrapping

Note that the interpretive and synonymous parts are not intrinsic features of definition texts. Therefore, the number of definitions that follow synonymous-part patterns is limited. In addition, the hand-crafted definition texts usually exhibit many typographical variations: some definitions, for example, end with periods while others do not. It is therefore necessary to devise a pattern-finding mechanism that can both obtain new synonym patterns and accommodate variations with minimal hard-wiring or human intervention.

Bootstrapping can achieve both goals at the same time. Specifically, a bootstrapper is initialized with a word w as well as a seed regex pattern p, which could be some simple pattern for synonymous parts in a given dictionary (e.g., '`.*;(\w+)$`' in *The Macquarie Dictionary*). The output is a set of regex patterns that synonyms follow within definition texts. By applying p on the definientia of w, the bootstrapper gets a set S of synonyms of w. Given the fact that dictionary definitions are often circular (Jurafsky and Martin 2008), it can be assumed that some elements of S are to appear in the definientia of others. If any of these occurrences follows pattern p' other than p, p' is then added to the resulting pattern set. Currently, p' is identified manually, i.e., the bootstrapper would output any of the circular definitions among

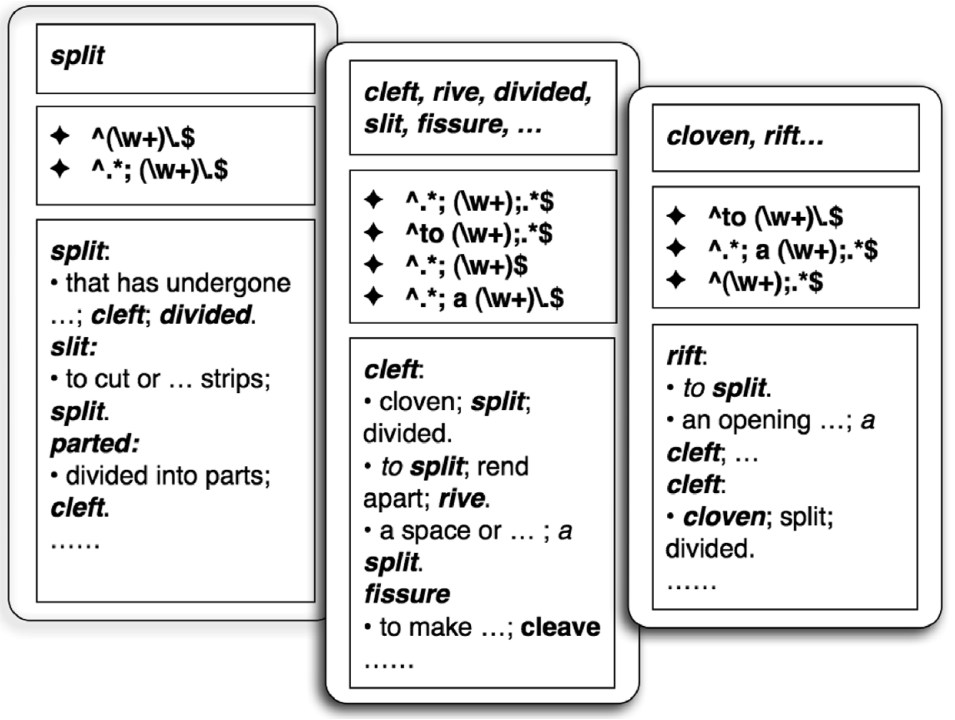


Fig. 3. An example of bootstrapping patterns. The three rounded rectangles in the horizontal layout represent three iterations of bootstrapping; the three vertically distributed rectangles within, from top to bottom, are the extracted synonyms, newly added regular expression patterns, and related definitions.

the set of words S, which provides a human user with potential patterns followed by synonyms.

Figure 3 gives an example of how patterns are bootstrapped from a seed word (*split*) and two seed patterns. Starting from the seed word *split* and the seed patterns '`^(\w+)\.$`' and '`^.*; (\w+)\.$`', *cleft* and *divided* are firstly extracted from the definition of *split*. *Parted* is also added to the synonym set since its inclusion of *cleft* follows one of the seed patterns. As the number of synonyms grows, it becomes more and more likely for some of the synonyms to appear in the definitions of others under patterns different from the seed patterns. In the second iteration, for example, *split* appears in the middle of one definition of *cleft*, resulting in a new pattern '`^*;(\w+);.*$`'. More synonyms could, in turn, be extracted using these new patterns. Examples of bootstrapped patterns are listed in Table 4.

In practice, the resulting synonym and pattern sets converge fairly quickly, since circular definitions within a dictionary are usually limited to a small number of entries. The resulting pattern set captures most of the patterns under which one synonym is used to define another; it also accommodates some of the typographical variations, such as the ending period in definitions (`^.*; (\w+)\.$` versus `^.*; (\w+)$`). It is also worth noting that, since there is no constraint on part of speech in this process, patterns that are apparently for verbs, e.g., '`^to (\w+)\.$`',

Table 4. *A list of patterns manually selected from those suggested by bootstrapping*

```
.*(; (?:a |an |the |or )?(\w+))\.?$
.*(; (?:a |an |the |or )?(\w+)); .*
^((?:a |an |the |or )?(\w+); ).*
^((?:a |an |the )?(\w+)\.?)$
^((?:a |an |the )?(\w+) or )\w+(?:\.|; .*)?$
^((?:a |an |the )?(\w+), )+or \w+\.?$
.*(; (?:to )?(\w+)) ?\.?
.*(; (?:to )?(\w+)) ?; .*
^((?:to |or )?(\w+) ?; ).*
^((?:to |or )?(\w+)(?: \(.*\))? ?\.?)$
```

are mixed together with those for nouns, e.g., '`^.*; a (\w+)\.$`'. This might have some negative effects on the algorithm's efficiency, but not the performance, since POS constraints can always be easily imposed by checking the words' POS tags in the dictionary.

The seemingly simple seed patterns (e.g., '`^.*; (\w+\)$`', '`^.*; (\w+\)\.$`', and '`^.*; \w+\;.*$`') can in fact match as many as 15,228 (7.45%) definition texts in *The Macquarie Dictionary*. By bootstrapping, 16 more patterns have been discovered, doubling the coverage of the seed patterns (32,208 or 15.75% definition texts).

2.3.4 Transitive closure on the dictionary graph

One important feature of synonymy is transitivity, i.e., if word *a* is synonymous to word *b* and *b* to *c*, then it is usually plausible to infer synonymy between *a* and *c*. Considering the common problem of dictionaries' low coverage on synonymy extraction, this property is especially useful since transitivity allows one to take *c*, and even synonyms of *c*, as synonyms of *a*.

On the other hand, the more the rule of transitivity is applied, the less synonymous the resulting synonyms will become to the target word. We can view the output of PbE as a tree structure, with the root of the tree being the target word *w*, and the immediate children the extracted synonyms in first round PbE ($S = \{s_1, \ldots, s_n\}$). When there is more than one iteration of PbE, each synonym $s_i \in S$ is taken to be the root of a subtree, from which sprout more proposed synonyms. The degree of synonymy between *w* and a child node *s* will certainly decrease with the increasing depth of *s*. Nonetheless, due to the circular nature of dictionary definitions, there must be cases in which certain paths in the tree would return to *w* after several iterations. Here, we refer to a nonempty path *p* between *w* and itself as a *transitive closure* on the dictionary graph, and the intuition behind *transitive closure filtering* is that words on such paths should be more synonymous to the target word than those that 'wander off' and never come back. Thus, by finding these circles on the dictionary graph, transitive closure is an intuitively feasible way of dealing with the negative effect of polysemy on synonym transitivity.[5]

[5] In addition to finding closed paths, some colleagues also suggested looking at cliques or subgraphs with certain density threshold. The reason why closed paths are preferred here

fear(noun)
solicitude(noun)
fear(noun), ~~care(noun)~~
anxiety(noun)
~~eagerness(noun)~~, **fear(noun)**
~~care(noun)~~, ~~jump(noun)~~, ~~disease(noun)~~
shock-horror(noun)
fear(noun), **terror(noun)**
terror(noun)
shock-horror(noun)
apprehend(verb (i))
~~understand(verb (t))~~, ~~anticipate(verb (t))~~
fear(verb (i)), ~~conceit(verb (t))~~

Fig. 4. Transitive closure filtering (**bold** for extracted synonyms, ~~strikeout~~ for the filtered words).

Thus, to filter out false positives from PbE, only those words on transitive closures starting from the target word are proposed as synonyms. However, experiments show that transitive closure of only the target word is quite sparse; considering the fact that words extracted in the first round of PbE are usually highly synonymous to the target word, the filtering strategy is relaxed so as to include transitive closures of these words as well. Such relaxation has proved to be successful in that it increases the coverage at little cost to accuracy. In the example of *fear*, the extracted synonyms are: *solicitude*, *anxiety*, *shock-horror*, *terror*, and *apprehend*, while words, such as *care*, *eagerness*, *jump*, *disease*, *understand*, *anticipate*, *conceit*, and *doubt*, are filtered out (Figure 4). Both groups appeal well to intuition. When the extracted synonyms are compared with existing thesauri (Section 3.3), the precision of the filtered results increases by 18.3 percentage points on average, at a recall loss of about 7 percentage points.

2.4 Extraction using maximum entropy

Although PbE exhibits excellent extraction precision (Section 3), coverage is still low due to the limited number of patterns. This motivates general learning methods that treat definition texts in a more generic manner. As an initial attempt at machine learning approaches for synonym extraction from definitions, we formulate the synonym extraction task as a labeling problem: each word in a piece of definition text is a decision point, and a maximum entropy (MaxEnt) classifier is trained to decide whether a word is a synonym of the corresponding definiendum.

The training data consist of 186,954 definition items (definiendum with corresponding definientia) in the *Macquarie Dictionary*. After POS-tagging,[6] any word in a given definition text is labeled as a synonym of the definiendum if the word is (1) of the same POS as the definiendum, and (2) in the same WordNet synset as the definiendum.

again comes from the sparsity of connectivity in the dictionary graph. We observe that the size of a clique seldom exceeds three, and thus, coverage would again become a prominent issue if only cliques were considered synonym sets.

[6] The *Stanford Log-linear Part-of-Speech Tagger* (`http://nlp.stanford.edu/software/tagger.shtml`) is used for POS-tagging the definitions.

We choose the *opennlp.maxent* implementation of the classifier with generalized iterative scaling (GIS) capacity.[7] For each word in the definientia of a given target word, we use lexical features (previous, current, and next word), unigram POS features (previous, current, and next POS), and bigram POS features (previous and next POS bigrams). In addition, another group of features describes the position of each decision point by an integer counter starting from 1 to the length of a definition text. In order to capture the separators discussed in PbE (e.g., semicolons), a second position counter is included which resets to 1 whenever encountering any separators. For example, in the definition of *abbreviation*: '`reduction in length; abridgment`.', the first counter assigns integers 1 to 6 to all definientia (including punctuation ';' and '.'), whereas the second counter assigns 1 to 4 to definientia up to the semicolon but 1 and 2 to *abridgment* and the period.

As we can see, the features of choice are very 'local' to the target word. We tried incorporating larger windows of context as features, but this did not perform as well. This suggests that, unlike in semantic similarity tasks, words from a larger context bring in more noise than useful information in predicting synonyms within definition texts.

Note that in order to make fair comparisons with IIE and PbE in terms of coverage, it is necessary to incorporate the dictionary graph into the MaxEnt method. Specifically, given a target word t, after extracting synonyms from its own definientia, we again go through other words' definitions in the dictionary; if t appears in the definition of another word w and is classified as a synonym, then w is taken as a synonym of t.

2.5 Interpretation of the methods in terms of the dictionary graph

So far in our discussion, the dictionary graph has been assumed to be *undirected*. For IIE, if we are to take the graph as *directed* (with edges pointing from definienda to definientia), then the in-neighbors of a target word are those related by an inverted index, and the out-neighbors are simply all its definientia. We will see how these two types of relatedness perform differently in Section 3.

In contrast, PbE and MaxEnt make fine distinctions about which part of the definitions to relate a target word to. For out-neighbors (words in the definientia of the target word), PbE and MaxEnt choose words that follow specific patterns (either regular expression patterns or, implicitly, patterns learned by a classifier), as opposed to IIE, which takes all definientia indiscriminately; for in-neighbors, IIE relates them all regardless of how or where the target word appears in other words' definitions, while PbE and MaxEnt, again, follow their respective patterns.

3 Evaluation

In this section, we present three experiments for evaluating the extracted synonyms for a given target word. Comparing synonym extraction results with existing

[7] Available at `http://sourceforge.net/projects/maxent/`.

thesauri is straightforward and intuitive. Muller *et al.* (2006), however, observed that 'comparing (extracted synonyms) to (an) already existing thesaurus is a debatable means, when automatic construction is supposed to complement an existing one'. In the same study, it is shown that even thesauri themselves do not correlate well: when several French thesauri were compared against one another, none of them scored over 60% in F-measure.

Another way to look at synonym evaluation is to establish a mapping between synonymy and semantic similarity. This idea has actually been implicitly adopted by many previous studies. Given a similarity measure, the notion of synonymy can be implemented by listing words in nonincreasing order in terms of their similarity scores with respect to a target word or concept. Conversely, once there is a way of extracting synonyms for a target word, a similarity measure can be built, for example, by computing the overlap between the synonym sets of any two words (e.g., the experiment in Section 3.2).

3.1 The machine-readable dictionaries

The machine-readable dictionaries used in this study are an electronic version of *Macquarie Dictionary* (Delbridge 1981) and the *WRUD*.[8] *The Macquarie Dictionary* is contained in an SGML-tagged file of 63 MB with 78 types of tags. Information about each lexical entry, including pronunciation, part(s) of speech, definitions, related phrases, etc., is represented by a tree structure of tags, rooted at a tag named `RECORD`. There are altogether 106,964 such entries in the machine-readable dictionary. Figure 5 shows an example of the tree structure of SGML tags for the entry word *dictionary*.

WRUD has 182,698 entries, each on a separate line in one of the 26 HTML-marked files corresponding to the English letter the entry word starts with. The main structure in each entry consists of the entry word form, its part of speech (if any), and the definition text. Multiple senses of the same word form occupy separate lines (and thus, only 115,773 distinct word forms are defined in the dictionary). Figure 6 shows an excerpt from the *WRUD* for the word *dictionary*.

3.2 Solving TOEFL synonym questions

3.2.1 Experimental setup

Our first evaluation is to use the extracted synonyms to solve TOEFL synonym questions. TOEFL is a standardized test for assessing the English level of nonnative speakers. Part of the test is on synonymy, where each question consists of a question word and four candidates, one of which is a synonym to the question word and therefore the correct answer. Landauer and Dumais (1997) first compiled and used

[8] *WRUD* was published in 1913. In this study, we used a preprocessed version known as *Online Plain Text English Dictionary* (`http://www.mso.anu.edu.au/~ralph/OPTED/`), which, in turn, is based on the Project Gutenberg E-text of *WRUD*.

```
<RECORD id="000020291">
  <HEAD>[dictionary]
  <SORTKEY>[DICTIONARY0990010000]
  <FLAGS>[BIGM N]
  <PRON>
    <PRN>['d1k47nri]
    <PRN TYPE="SAY">['dikshuhnree]
    <PRN>['d1k47n7ri]
    <PRN TYPE="SAY">['dikshuhnuhree]
  <BODY>
    <CHUNK>
      <POS>[noun]
      <INFLECTION>
        <INF NUMBER="PL">[dictionaries]
      <DEF id="322">
        <DTEXT>[a book containing a selection of the words of a
        language, usually arranged alphabetically, with explanations
        of their meanings, pronunciations, etymologies, and other
        information concerning them, expressed either in the same or
        in another language; lexicon; glossary.]
        <THES>[599.04.10]
      <DEF id="157">
        <DTEXT>[a book giving information on particular subjects or a
        particular class of words, names or facts, usually under
        alphabetically arranged headings]
        <IP>[a biographical dictionary.]
  <ETY>[, lit., a word-book, fromword. See]
    <LANG>[Medieval Latin]
    <I>[dicti[omacr ]n[amacr ]rium]
    <LANG>[Late Latin]
    <I>[dictio]
    <LINK>[diction]
  <STERM POS="N" LEMMA="HWD" TYPE="IINF" NUMBER="PL">[dictionaries]
```

Fig. 5. The tree structure of SGML tags for the entry *dictionary* in *The Macquarie Dictionary*.

```
Dictionary (n.) A book containing the words of a language, arranged
alphabetically, with explanations of their meanings; a lexicon; a
vocabulary; a wordbook.

Dictionary (n.) Hence, a book containing the words belonging to any
system or province of knowledge, arranged alphabetically; as, a
dictionary of medicine or of botany; a biographical dictionary.
```

Fig. 6. Entries for the word *dictionary* in *WRUD*.

80 of these questions, which have been frequently used as an evaluation benchmark in later lexical semantics studies.[9]

In essence, this evaluation method is to establish a mapping between synonymy and semantic similarity. For any word pair w_1 and w_2, a similarity measure can be constructed by, for example, computing the overlap between the synonym sets of the pair $S_i = \{s_{i1}, \ldots, s_{in_i}\}$, $i = 1, 2$ (hence, a Jaccard similarity). The semantic similarity between w_1 and w_2 is then given by:

$$sim(w_1, w_2) = \frac{|S_1 \bigcap S_2|}{|S_1 \bigcup S_2|}. \tag{1}$$

[9] http://aclweb.org/aclwiki/index.php?title=TOEFL_Synonym_Questions.

For a TOEFL synonym question, the similarity between the question word and each of the candidates can now be computed, and the candidate with the highest score is the proposed correct answer.

Under the question word *fabricate*, as an example, there are four choices *construct*, *alter*, *select*, and *demonstrate*. Each word is firstly associated with the synonym set proposed by a synonym extraction algorithm, say, IIE; the question word gets the set {*fabricate, coin, trump up, prefabricate, mint, invent, forge, spin*}, the first candidate *construct* gets {*construct, fabricate, cantilever, improvise, laminate, ...*}, and so on. Note that a word is always considered a synonym to itself and thus included in the synonym set. In the above synonym question, the first candidate *construct* is the only one with a set that overlaps with the set of the question word, and consequently, it receives the highest score and is considered the correct answer.

Scores on this experiment are easy to interpret and are positively correlated to the degree of synonymy of the extracted synonyms to their target word and thus the similarity score by (1). However, this evaluation also bears several immediate problems. First, synonymy is sufficient but not necessary for achieving higher similarity scores. Suppose, as an extreme example, that the sets S_i consist of antonyms instead of synonyms to the corresponding target words w_i $(i = 1, 2)$; two synonyms are likely to have similar antonyms, and the two sets S_1 and S_2 can also correlate well and achieve better scores in the same synonym questions. Consequently, higher scores in such tests are only necessary but not sufficient to infer synonymy between the target words and their corresponding extracted sets. Moreover, two choices might have exactly the same Jaccard similarity with the question word.[10] It is also possible that none of the choices overlap with the question word (i.e., none of them share any synonyms with the question word) or that the synonym set for the question word is empty. For IIE, for example, the number of extracted synonyms is very limited when the target word is uncommon. This turns out to be especially problematic in the TOEFL questions, since one of the characteristics of the exam is to test nonnative speakers on an advanced vocabulary with many uncommon words.

In contrast, similarity-based approaches usually assign nonzero scores to most word pairs. Consequently, results reported in such studies are only concerned with how many questions were correctly solved. Here, questions with ties are taken to be 'unsolvable' and no choice would be assigned. Thus, the results later presented borrow the notions of precision and recall from information retrieval. Recall, in this case, denotes the proportion of questions that have nonzero scores, and thus, are solvable by choosing the highest score among the four, while precision denotes how many are solved correctly among these solvable questions.

One way to break ties is to improve coverage by combining IIE or PbE synonym sets with definientia. Specifically, for two target words w_1 and w_2, we use $v_i = S_i \bigcup D_i, i = 1, 2$ as their feature sets, where $D_i = \{d_{i1}, \ldots, d_{im_i}\}$ are words from their respective definientia. To differentiate two types of features, more weights are assigned to synonyms than to ordinary definientia: suppose w_1 and w_2 have two

[10] This situation never happened in the 120 questions used in this experiment, partly because the size of the extracted synonym sets varies widely.

Table 5. *Evaluation of extracted synonyms on TOEFL synonym questions. Each cell contains two numbers from The Macquarie Dictionary and WRUD, respectively, separated by dashes*

	Precision	Recall	F_1	Accuracy
IIE_{out} (Baseline)	0.513–0.590	0.975–0.975	0.672–0.735	0.500–0.575
IIE_{in}	**1.000**–0.800	0.500–0.625	0.667–0.702	0.500–0.500
IIE+Lemma	0.872–0.649	0.975–0.975	0.921–0.779	0.850–0.633
PbE+Lemma	0.936–0.800	0,775–0.938	0.848–0.863	0.725–0.750
PbE+Definiens	0.906–0.719	0.975–**1.000**	**0.939**–0.836	**0.883**–0.719
MaxEnt+Lemma	0.550	0.546	0.548	0.300

common elements v_{1i} and v_{2j} in the ith and jth positions of their feature vectors, and the weight is α for synonyms and β for ordinary definientia. If $v_{1i} \in S_1$ and $v_{2j} \in S_2$ (i.e., both from IIE/PbE results), then this overlapping is weighted by α^2; if $v_{1i} \in S_1$ while $v_{2j} \in D_2$ (or the other way around, i.e., one from IIE/PbE the other from definientia), then the overlapping weight is $\alpha \cdot \beta$; if both are parts of the ordinary definientia, then the weight becomes β^2. In the current implementation, $\alpha = 5$ and $\beta = 1$ (estimated by maximum likelihood using grid search on integer values).

We also compare to a baseline algorithm using only the definientia of each target word, which resembles the Lesk algorithm used in word sense disambiguation (Lesk 1986). The comparison is interesting in that IIE and PbE try to distinguish synonyms from the rest of the definientia, while the baseline uses them all indiscriminately; improvements over the baseline would thus reflect how well the discrimination is made.

3.2.2 Evaluation results

Table 5 shows the results of solving TOEFL synonym questions by the three models and some of their variants. IIE_{in} and IIE_{out} denote variants of *IIE* with in-neighbors only and out-neighbors only, respectively; *IIE* without subscripts corresponds to the original *IIE* method (with both in- and out-neighbors). Due to the low coverage, half of the questions have ties when using IIE. Nonetheless, for the solvable ones, IIE scores 100% in precision. PbE, on the other hand, exhibits higher recall although by a very small margin.

Error analysis on the tied questions reveals that they are due to diminished synonym sets that do not overlap with one another. For the question ⟨`functional: alternate; unknown; original; usable`⟩, for example, although all of the five words have nonempty synonym sets from *The Macquarie Dictionary*, none of them have any word in common, and thus, all four candidates receive a score of zero. For inflected words in the data, using base forms in IIE and PbE improves the coverage (recall) by a large margin (25.0 and 22.5 percentage points, respectively) even through a very simple lemmatization process.

When used 'as-is', *WRUD* has rather poor coverage compared to *The Macquarie Dictionary*. Error analysis reveals that the dictionary has no entries even for some

not-so-rare words, such as *expendable*, *unpredictable*, and *optimal*. Some frequent senses of words, for example, the adjective sense of *key* meaning 'important', are missing from the dictionary too. Most of these words and senses, according to the *Oxford English Dictionary*, have come to frequent use only after the *Webster's Revised Unabridged Dictionary* (and thus entries in *WRUD*) were compiled (in 1913). We thus skipped all the words that are not defined in *WRUD* in the experiment.

The best F_1 score is achieved by the weighted PbE+Definientia approach on *The Macquarie Dictionary*, which, by including the definientia in the feature vectors, significantly improves the recall (20.0 percentage points) at a relatively smaller cost to precision (3.0 percentage points).

The percentage of correctly solved questions is equivalent to 'accuracy' in Table 5. We compare our results to the state-of-the-art results reported on the *ACLweb wiki* (see footnote 9). As of February 2011, the top six results listed there are achieved by either corpus- or Web-based approaches, ranging from 0.975 to 0.813 in accuracy. The best lexicon-based method ranks seventh at 0.788 (Jarmasz and Szpakowicz 2003), which is approximately 10 percentage points lower than PbE with definientia. When using only the extracted synonyms as feature vectors, PbE is still comparable to existing results. Due to its low coverage, however, it is still about 9 percentage points below the best performing hybrid model (0.975, Turney *et al.* 2003).

3.3 Comparison against a combined thesaurus

The experiment discussed in this section resembles that of Wu and Zhou (2003). Target words are first selected from a corpus according to POS and frequency. A thesaurus is then constructed by combining WordNet synsets and an on-line version of *Roget's Thesaurus* (Roget 1911).[11] Note that this version of *Roget's Thesaurus* groups together synonyms and other related words of various POS. We only used those of the same POS as the target word. After synonym extraction algorithms are applied to the target words, the resulting synonym sets are compared against the combined thesaurus.

The corpus for choosing the target words is the 1987–1989 *Wall Street Journal* (WSJ). POS of these target words include nouns (NN), verbs (VB), and adjectives (JJ). Word frequencies range from approximately 8,000 (high) to 1,000 (medium), to 50 (low) occurrences in the corpus. The combined thesaurus is constructed in exactly the same manner as by Wu and Zhou (2003), i.e., given a target word, its corresponding WordNet synsets and synonym sets from *Roget's Thesaurus* are extracted and combined into a larger synonym set. The resulting thesaurus is then used as a gold standard against which the extracted synonyms are compared. The final results are reported in terms of precision, recall, and F_1.

The results for PbE, IIE, and their variants are listed in Table 6. The letters `H`, `M`, and `L` stand for the high, medium, and low frequency of target words in the WSJ, and `P`, `R`, F_1, for precision, recall, and F_1, respectively. Due to the computational

[11] `http://www.bartleby.com/110/`, the same one used by Wu and Zhou (2003).

Table 6. *Evaluation of extracted synonyms on the combined thesaurus*

		NN			JJ			VB		
		P	R	F_1	P	R	F_1	P	R	F_1
IIE	H	.057	.032	.041	.045	.053	.049	.113	.026	.043
	M	.168	.040	.051	.132	.051	.073	.175	.041	.066
	L	.053	.023	.032	.087	.013	.023	.168	.026	.046
PbE	H	.109	.189	.138	.109	.291	.158	.119	.372	.180
	M	.132	.175	.150	.125	.288	.175	.125	.269	.171
	L	.109	.164	.131	.117	.172	.139	.168	.208	.186
PbE_TC	H	.329	.113	.168	.334	.174	.229	**.489**	.181	**.264**
	M	.347	.112	.169	.332	.157	.213	.415	.149	.219
	L	.225	.090	.128	.335	.089	.140	.370	.116	.177
PbE_TC *WRUD*	H	.102	.264	.147	.085	.264	.129	.206	.329	.253
	M	.069	.255	.108	.086	.265	.130	.098	**.379**	.155
	L	.036	.194	.061	.032	.261	.056	.057	.293	.096
PbE_TC + IIE_Filtered	H	.267	.121	.167	.262	.185	.217	.160	.185	.263
	M	.035	.114	.166	.308	.161	.212	.043	.151	.220
	L	.215	.091	.127	.340	.095	.148	.361	.117	.177

intensity of MaxEnt, it is implemented only on nouns to compare with the result by Wu and Zhou (2003), which is shown in Figure 9.

As is shown in Table 6, the output of IIE does not correlate well with the combined thesaurus. Precision is, in general, slightly better than recall, but F_1 seldom exceeds 5%. Also notice that IIE performs best on mid-frequency words, mainly because these words are neither too frequent to appear in many words' definitions (as are high-frequency words and hence their low precision), nor too rare to not appear at all (as are low-frequency words and hence their low recall).

In contrast, PbE exhibits significant improvements over IIE on both precision and recall (on average, twice as high in precision and five times in recall). It also appears more robust to variation in frequency. After the transitive closure filtering, the precision of PbE is almost tripled, which, at a relatively small cost of recall drop (6–7 percentage points), yields the best F_1 score among all proposed methods. A combination of the filtered versions of PbE and IIE is also shown, but the result (the last group of data) is rather disappointing.

Since POS is not of primary interest here, we average the results across the three different POS. Figure 7 shows how IIE compares with PbE across different target word frequencies. On average, PbE has slightly better precision and drastically better recall, resulting in F_1 scores approximately 3–5 times as high as those of IIE. The performance of IIE is apparently 'spiked' at medium target word frequency, confirming our previous hypothesis that IIE would underperform when the target word frequency is too low or too high. In contrast, PbE exhibits 'smoother' performance especially in precision and F_1 score.[12]

[12] Even recall, which seemingly drops drastically as frequency decreases, is still smoother than that of IIE if drawn at equal scale.

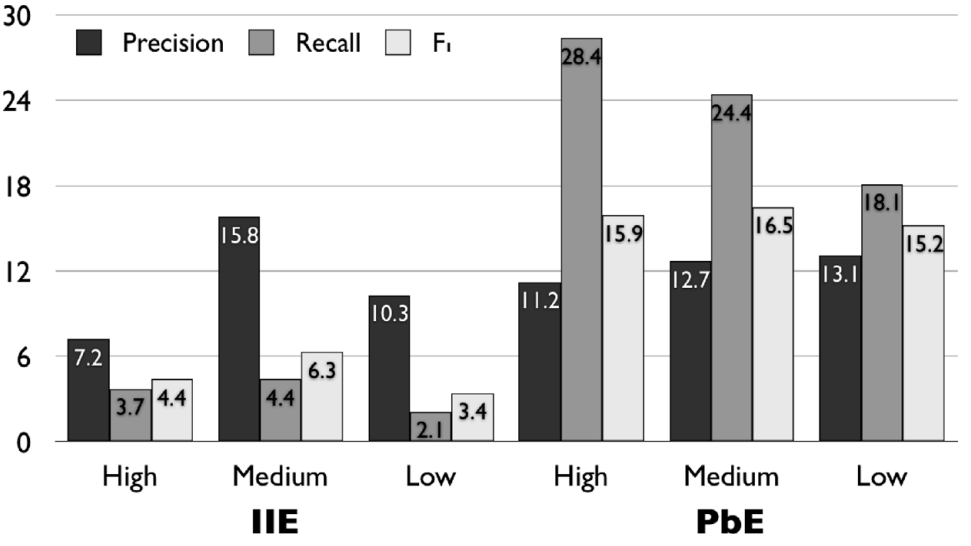


Fig. 7. Inverted index extraction versus pattern-based extraction when compared with existing thesauri. *High*, *Medium*, and *Low* refer to different frequencies of target words in the *Wall Street Journal*.

Precision of PbE increases as target word frequency decreases. We speculate that this is because the degree of polysemy of a word is approximately in proportion to its frequency; high-frequency words, being more polysemous, would have more chance of 'digressing' to various branches of different senses; they also tend to appear in many different words' definitions under different senses. This is especially true when transitivity of synonymy is applied with no constraints. We will show shortly how transitive closure on the dictionary graph helps alleviate this problem.

The drop of recall in PbE with respect to frequency can be explained by different in- and out-degree of target words of different frequencies. Words of higher frequency would not only have a higher out-degree (due to their polysemy), but also a higher in-degree since they are more likely to appear in other words' definitions. In contrast, low-frequency words would have fewer senses and thus smaller numbers of definitions; if they are too infrequent to appear in other words' definitions, then these few definitions of their own would be the only source for synonyms, which would, not surprisingly, result in lower recall.

We also compared PbE with transitive closure calculated using *WRUD*. The trends across POS and frequencies are similar to that obtained from *The Macquarie Dictionary*. Recall is significantly higher for *WRUD* results, but F_1 score, in general, is lower due to very low precision, which we think is due to the following reasons. First, the entries in *WRUD* are sometimes of low quality. Some words are defined in uncommon senses (e.g., *countenance* as '`To make a show of; to pretend`' and '`To encourage; to favor; to approve; to aid; to abet`'), others are spelled in obscure or nonexistent variations (e.g., *counttenance* as '`The face; the features`'). Second, unlike *The Macquarie Dictionary*, *WRUD* does not have usage labels (e.g.,

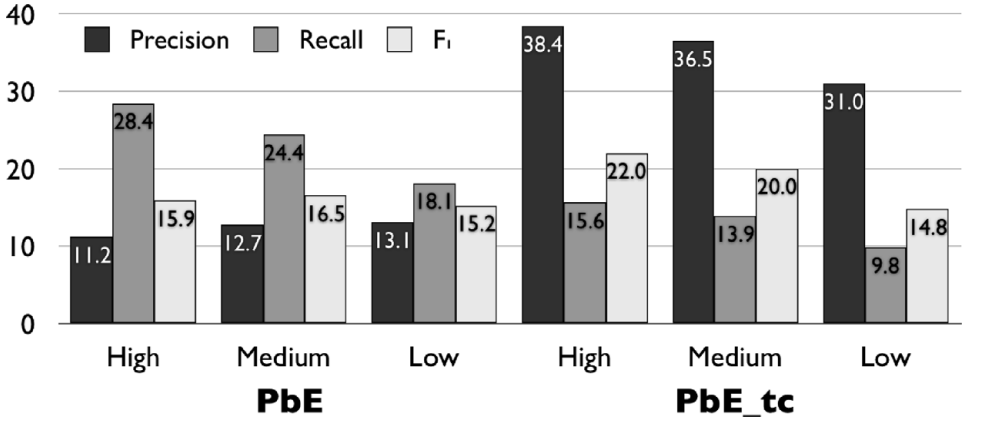


Fig. 8. Performance before and after using transitive closure on pattern-based extraction (denoted *PbE* and *PbE_tc*, respectively). *High*, *Medium*, and *Low* refer to different frequencies of target words in the *Wall Street Journal*.

archaic, *agricultural*) that can suppress paths along nontypical usage of word forms in the construction of transitive closure (and thus, it also explains the high recall).

Figure 8 shows the improvement in PbE performance by finding transitive closure as mentioned in Section 2.3.4. Recall drops to about half of the original values after using transitive closure (denoted PbE_tc in the graph), but meanwhile precision is more than tripled in all frequencies. It is interesting to observe how precision responds differently to frequency change before and after using transitive closure: without transitive closure, precision increases as frequency decreases, while after transitive closure is introduced, it varies in the opposite direction. This indicates that using transitive closure is most helpful for high-frequency target words. This is, again, due to their polysemy and better chances of 'digression', and thus, transitive closure indeed helps to effectively eliminate false positives introduced by such digressions; low-frequency words would already have relatively better precision due to their having fewer senses, and transitive closure appears less helpful in this case.

Figure 9 shows how our methods compare with other published results. IIE is outperformed by all other methods by large margins. PbE has the best precision (32.9%) but falls behind that of Wu and Zhou (2003) in terms of F_1 due to low recall. MaxEnt has better recall than both IIE and PbE, but F_1 score is not as good as that of PbE. The results of Blondel and Senellart (2002) are included as an example of dictionary-based method for comparison, and Lin (1998) as an example of corpus-based approach.[13] Wu and Zhou (2003) combined the methods of Blondel and Senellart (2002) and Lin (1998), as well as a novel method using bilingual resources, achieving the best F_1 score among all methods being compared here. Their experiments did not include adjectives; when comparing the results on nouns and verbs, however, the precision of PbE with transitive closure filtering is, on average, 6 percentage points higher than their best result on nouns and

[13] Results of both Blondel and Senellart (2002) and Lin (1998) are reported by Wu and Zhou (2003).

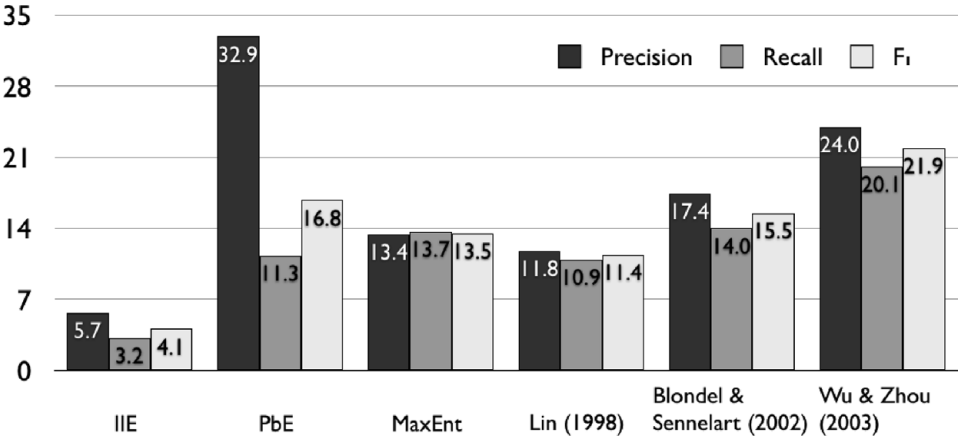


Fig. 9. Comparing with published results on the combined thesaurus experiment.

14.3 percentage points higher on verbs. Coverage, however, still remains a major issue, with recalls of PbE_TC filtering on both nouns and verbs lower than that of Wu and Zhou (2003) (11.3% and 16.1 percentage points, respectively).

As one advantage of using a dictionary as a source for synonym extraction, PbE and IIE both exhibit robustness across different POS.

3.4 Definition text labeling

This section describes an intrinsic task on identifying synonyms within definition texts. Recall that the MaxEnt model labels synonyms in a piece of definition text for a given target word; in fact, PbE and IIE could also be viewed as a synonym labeling processes in definition texts. The basic idea of this third experiment is to see how well each method performs in such a labeling task.

The construction of data was described in Section 2.4; it does not necessitate any human labeling, though at the cost of the quality of synonym labels (to be discussed below). The labeling criteria for the three methods follow the discussion in Section 2.5: IIE takes all definientia as synonyms, while PbE takes only those following prespecified patterns. MaxEnt makes predictions for each defining word based on training. We also introduce a baseline that chooses a defining word as a synonym as long as it has the same POS as that of the definiendum.

The results are presented in Figure 10. The baseline and IIE both have 100% recall by experimental design. IIE and PbE are both outperformed by the baseline. PbE has the highest precision and meanwhile, the lowest recall due to its dependence on specific patterns.

Due to the low quality of the training data, MaxEnt did not perform as well as expected. POS tags have many discrepancies, partly because the tagger is not trained on definition texts. On the other hand, using WordNet to create the gold standard in synonym labels also appears to be error-prone. For example, in the definition of ability (*power or capacity to do or act...*), *power* is labeled as a synonym of *ability* while *capacity* is not, since it is not in the same synset as that of *ability*. There are

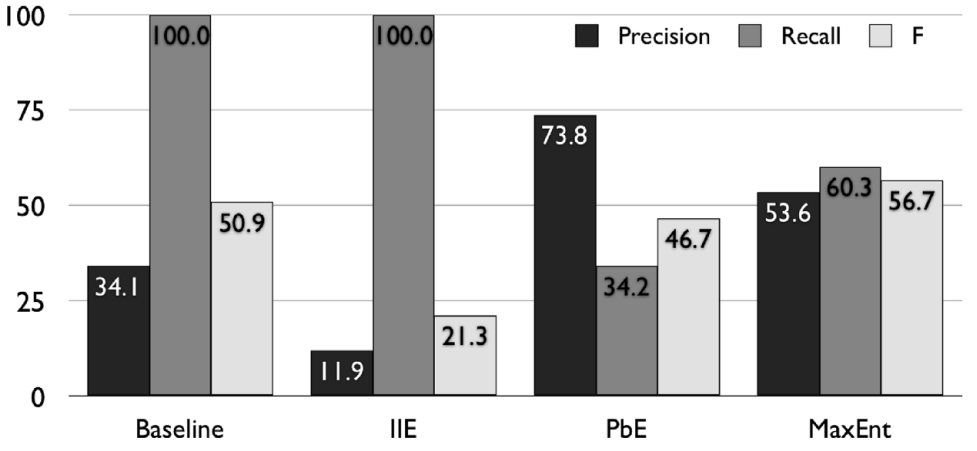


Fig. 10. Performance on synonym labeling in definitions.

also cases where words in insignificant positions within the definition text happen to be in the same synset as that of the definiendum. All such cases will eventually confuse the learning process of MaxEnt.

4 Conclusion and future work

We have presented three novel methods for extracting synonyms from dictionary definitions. Two of them are rule-based systems and one is a machine learning method based on maximum entropy classification. Compared to corpus-based methods, the proposed approaches all have the advantage of lexicon-based methods such as light weight in computational resource and complexity, and easy adaptation across different domains or languages.

Evaluation results show that simple extraction methods can perform fairly well on extrinsic experiments, such as solving TOEFL synonym questions; in fact, our PbE method gives the best performance on this task among all lexicon-based methods reported to date. Our methods also perform comparably to existing corpus-based studies when the extracted synonyms are compared against existing thesauri. Meanwhile, although the proposed methods, in general, exhibit excellent precision in our experiments, coverage of lexicon-based methods is obviously limited to the coverage of the dictionary of choice. The inverted indexing method, for example, yields 100% precision in the TOEFL synonym task, while it fails to extract any synonyms for some words due to low coverage. A similar situation applies also to the pattern-based method when its extraction results are compared to existing thesauri: the method scores the highest precision among all methods compared but the F_1 score is drastically compromised by low recall.

Another problem comes from polysemy. Transitivity of synonymy is not well preserved when synonyms of different senses of a polysemous word are mixed together. Sense-disambiguated definition texts are required to avoid this problem. Improvement of the methods can be made by automating the process of pattern bootstrapping. Although identifying regular expressions from free texts is a

1. administered
   a. managed
   b. recognized
   c. unregulated
   d. justified

Fig. 11. A revised TOEFL synonym question with related but not synonymous choice.

challenging task, the rules used in our pattern-based method are usually well specified and simple enough for an automated process to follow.

There is also a promising outlook for developing new evaluation schemes of synonym extraction tasks. For example, the current version of TOEFL synonym questions used in Section 3.2 does not discriminate between strict synonymy and the more general notion of semantic relatedness because, for a given question, the three incorrect choices are almost always totally irrelevant to the question word. It would be interesting to make the decoys more difficult to distinguish by including semantically related but not synonymous words.

Note that in the example in Figure 11, the third choice has been changed from *opposed* to *unregulated*, which is related but not synonymous to the question word *administered*. Generally speaking, it might be difficult to devise experiments that directly measure the quality of synonyms. However, there are various NLP applications that use synonyms as a component of their systems. Theoretically, all such applications can be used as extrinsic benchmarks for evaluating extracted synonyms, and it would be a meaningful study to investigate their individual characteristics and applicability.

References

Alshawi, H. 1987. Processing dictionary definitions with phrasal pattern hierarchies. *Computational Linguistics* **13**(3–4): 195–202.

Amsler, R. 1980. *The Structure of the Merriam-Webster Pocket Dictionary*. PhD thesis, The University of Texas, Austin, TX.

Barnbrook, G. 2002. *Defining Language: A Local Grammar of Definition Sentences*, Amsterdam, The Netherlands: John Benjamins.

Barzilay, R., and McKeown, K. 2001. Extracting paraphrases from a parallel corpus. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics*, pp. 50–57, Toulouse, France.

Bikel, D., and Castelli, V. 2008. Event matching using the transitive closure of dependency relations. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pp. 145–148, Columbus, Ohio, USA.

Blondel, V., and Senellart, P. 2002. Automatic extraction of synonyms in a dictionary. In *Proceedings of the Society for Industrial and Applied Mathematics Workshop on Text Mining*, pp. 7–13, Arlington, Virginia, USA.

Boguraev, B., and Briscoe, T. 1989. *Introduction to Computational Lexicography for Natural Language Processing*, White Plains, NY: Longman.

Chodorow, M., Byrd, R., and Heidorn, G. 1985. Extracting semantic hierarchies from a large on-line dictionary. In *Proceedings of the 23rd Annual Meeting of the Association for Computational Linguistics*, pp. 299–304, Chicago, Illinois, USA.

Curran, J. 2002. Ensemble methods for automatic thesaurus extraction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 222–229, Philadelphia, Pennsylvania, USA.

Delbridge, A. ed. 1981. *The Macquarie Dictionary*. Australia: Macquarie Library, McMahons Point, NSW.

Fellbaum, C. 1998. *WordNet: An Electronic Lexical Database*. Cambridge, MA: MIT Press.

Freitag, D., Blume, M., Byrnes, J., Chow, E., Kapadia, S., Rohwer, R., and Wang, Z. 2005. New experiments in distributional representations of synonymy. In *Proceedings of the 9th Conference on Computational Natural Language Learning*, pp. 25–32, Ann Arbor, Michigan, USA.

Gabrilovich, E., and Markovitch, S. 2007. Computing semantic relatedness using Wikipedia-based explicit semantic analysis. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, pp. 6–12, Hyderabad, India.

Guthrie, J., Guthrie, L., Aidinejad, H., and Wilks, Y. 1991. Subject-dependent cooccurrence and word sense disambiguation. In *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics*, pp. 146–152, Berkeley, California, USA.

Guthrie, L., Slator, B., Wilks, Y., and Bruce, R. 1990. Is there content in empty heads. In *Proceedings of the 13th Conference on Computational Linguistics*, pp. 138–143, Helsinki, Finland.

Hagiwara, M. 2008. A supervised learning approach to automatic synonym identification based on distributional features. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pp. 1–6, Columbus, Ohio, USA.

Harris, Z. 1954. Distributional structure. *Word* **10**(23): 146–162.

Hearst, M. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th International Conference on Computational Linguistics*, vol. 2, pp. 539–545, Nantes, France.

Ho, N.-D., and Cédrick, F. 2004. Lexical similarity based on quantity of information exchanged – synonym extraction. In *Proceedings of the Research Informatics Vietnam-Francophony, Hanoi, Vietnam*, pp. 193–198.

Jarmasz, M., and Szpakowicz, S. 2003. Roget's thesaurus and semantic similarity. In *Proceedings of International Conference on Recent Advances in Natural Language Processing*, pp. 212–219, Borovets, Bulgaria.

Jurafsky, D., and Martin, J. 2008. *Speech and Language Processing*, 2nd ed. Upper Saddle River, NJ: Pearson Education.

Landauer, T., and Dumais, S. 1997. A solution to Plato's problem: the latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review* **104**(2): 211–240.

Lesk, M. 1986. Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In *Proceedings of the 5th Annual International Conference on Systems Documentation*, pp. 24–26, New York, USA.

Lin, D. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of the 17th International Conference on Computational Linguistics*, pp. 768–774, Montreal, Canada.

Lin, D., Zhao, S., Qin, L., and Zhou, M. 2003. Identifying synonyms among distributionally similar words. In *Proceedings of International Joint Conference of Artificial Intelligence*, pp. 1492–1493, Acapulco, Mexico.

Mandala, R., Tokunaga, T., and Tanaka, H. 1999. Combining multiple evidence from different types of thesaurus for query expansion. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 191–197, Berkeley, California, USA.

McCarthy, D., and Navigli, R. 2009. The English lexical substitution task. *Language Resources and Evaluation* **43**(2): 139–159.

Mish, F. ed. 2003. *Merriam-Webster's Collegiate Dictionary*, 11th ed. Springfield, MA: Merriam-Webster.

Mohammad, S., and Hirst, G. 2006. Distributional measures of concept-distance: a task-oriented evaluation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 35–43, Sydney, Australia.

Mohammad, S., Dorr, B., and Hirst, G. 2008. Computing word-pair antonymy. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pp. 982–991, Honolulu, Hawaii, USA.

Muller, P., Hathout, N., and Bruno, G. 2006. Synonym extraction using a semantic distance on a dictionary. In *Proceedings of TextGraphs: The Second Workshop on Graph Based Methods for Natural Language Processing*, pp. 65–72, New York, USA.

Navigli, R. 2009. Using cycles and quasi-cycles to disambiguate dictionary glosses. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 594–602, Athens, Greece.

Page, L., Brin, S., Motwani, R., and Winograd, T. 1999. The PageRank citation ranking: bringing order to the web. Technical Report 1999-66, Stanford Digital Library Technologies Project.

Procter, P. ed. 1978. *Longman Dictionary of Contemporary English*. London, UK: Longman.

Reichert, R., Olney, J., and Paris, J. 1969. Two dictionary transcripts and programs for processing them – the encoding scheme, Parsent and Conix, vol. 1. DTIC Research Report AD0691098.

Roget, P. 1911. *Roget's Thesaurus of English Words and Phrases*. New York, NY: TY Crowell.

Shimohata, M., and Sumita, E. 2002. Automatic paraphrasing based on parallel corpus for normalization. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation*, pp. 453–457, Canary Islands, Spain.

Turney, P., Littman, M., Bigham, J., and Shnayder, V. 2003. Combining independent modules to solve multiple-choice synonym and analogy problems. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing*, pp. 482–489, Borovets, Bulgaria.

Van der Plas, L., and Tiedemann, J. 2006. Finding synonyms using automatic word alignment and measures of distributional similarity. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pp. 866–873, Sydney, Australia.

Wu, H., and Zhou, M. 2003. Optimizing synonym extraction using monolingual and bilingual resources. In *Proceedings of the 2nd International Workshop on Paraphrasing*, pp. 72–79, Jeju Island, South Korea.