Linggle: a Web-scale Linguistic Search Engine for Words in Context

Joanne Boisson[+], **Ting-Hui Kao**[*], **Jian-Cheng Wu**[*], **Tzu-His Yen**[*], **Jason S. Chang**[*]
[+]Institute of Information Systems and Applications
[*]Department of Computer Science
National Tsing Hua University
HsinChu, Taiwan, R.O.C. 30013
{joanne.boisson, maxis1718, wujc86, joseph.yen, jason.jschang}
@gmail.com

Abstract

In this paper, we introduce a Web-scale linguistics search engine, *Linggle*, that retrieves lexical bundles in response to a given query. The query might contain keywords, wildcards, wild parts of speech (PoS), synonyms, and additional regular expression (RE) operators. In our approach, we incorporate inverted file indexing, PoS information from BNC, and semantic indexing based on Latent Dirichlet Allocation with Google Web 1T. The method involves parsing the query to transforming it into several keyword retrieval commands. Word chunks are retrieved with counts, further filtering the chunks with the query as a RE, and finally displaying the results according to the counts, similarities, and topics. Clusters of synonyms or conceptually related words are also provided. In addition, *Linggle* provides example sentences from *The New York Times* on demand. The current implementation of *Linggle* is the most functionally comprehensive, and is in principle language and dataset independent. We plan to extend *Linggle* to provide fast and convenient access to a wealth of linguistic information embodied in Web scale datasets including *Google Web 1T* and *Google Books Ngram* for many major languages in the world.

1 Introduction

As a non-native speaker writing in English, one encounters many problems. Doubts concerning the usage of a preposition, the mandatory presence of a determiner, the correctness of the association of a verb with an object, or the need for synonyms of a term in a given context are issues that arise frequently. Printed collocation dictionaries and reference tools based on compiled corpora offer limited coverage of word usage while knowledge of collocations is vital to acquire a good level of linguistic competency. We propose to address these limitations with a comprehensive system aimed at helping the learners "know a word by the company it keeps" (Firth, 1957). *Linggle* (**linggle.com**). The system based on Web-scaled datasets is designed to be a broad coverage language reference tool for English Second Language learners (ESL). It is conceived to search information related to word usage in context under various conditions.

First, we build an inverted file index for the *Google Web 1T* n-grams to support queries with RE-like patterns including PoS and synonym matches. For example, for the query "$V $D +important role", *Linggle* retrieves 4-grams that start with a verb and a determiner followed by a synonym of *important* and the keyword *role* (e.g., *play a significant role* 202,800). A natural language interface is also available for users who are less familiar with pattern-based searches. For example, the question "*How can I describe a beach?"* would retrieve two word chunks such as "*sandy beach* 413,300" and "*rocky beach* 16,800". The n-gram search implementation is achieved through filtering, re-indexing, populating an HBase database with the Web 1T n-grams and augmenting them with the most frequent PoS for words (without disambiguation) derived from the British National Corpus (BNC).

The n-grams returned for a query can then be linked to examples extracted from the New York Times Corpus (Sandhaus, 2008) in order to provide full sentential context for more effective learning.

In some situations, the user might need to search for words in a specific syntactic relation (e.g., *Verb-Object collocation*). The query *absorb $N* in n-grams display mode returns all the nouns that follow the verb ordered by decreasing n-gram counts. Some of these nouns might not be objects of the verb *absorb.* In contrast, the same

Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, pages 139–144,
Sofia, Bulgaria, August 4-9 2013.

query in cluster display mode will control that two words have been labeled *verb-object* by a parser. Moreover, n-grams grouped by object topic/domain give the learner an overview of the usage of the verb. For example the verb *absorb* takes clusters of objects related to the topics *liquid*, *energy, money*, *knowledge*, and *population*.

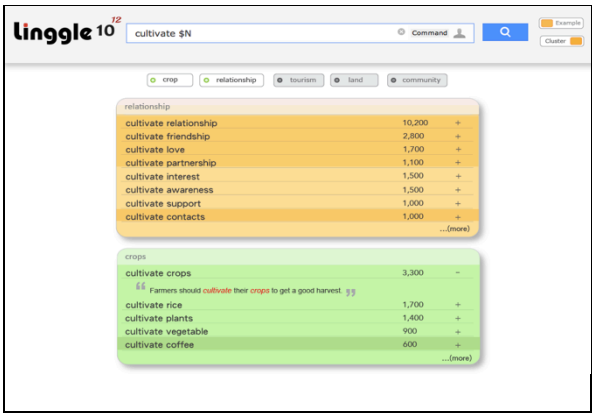

Figure 1. An example Linggle search for the query "absorb $N."

This tendency of predicates to prefer certain classes of arguments is defined by Wilks (1978) as selectional preferences and widely reported in the literature. Erk and Padó (2010) extend experiments on selectional preference induction to inverse selectional preference, considering the restriction imposed on predicates. Inverse sectional preference is also implemented in *linggle* (e.g. "$V apple").

Linggle presents clusters of synonymous collocates (adjectives, nouns and verbs) of a query keyword. We obtained the clusters by building on Lin and Pantel's (2002) large-scale repository of dependencies and word similarity scores. Using the method proposed by Ritter and Etzioni (2010) we induce selectional preference with a Latent Dirichlet Allocation (LDA) model to seed the clusters.

The rest of the paper is organized as follows. We review the related work in the next section. Then we present the syntax of the queries and the functionalities of the system (Section 3). We describe the details of implementation including the indexing of the n-grams and the clustering algorithm (Section 4) and draw perspective of development of Web scale search engines (Section 5).

2 Related work

Web-scale Linguistic Search Engine (LSE) has been an area of active research. Recently, the state-of-the-art in LSE research has been reviewed in Fletcher (2012). We present in this paper a linguistic search engine that provides a more comprehensive and powerful set of query features.

Kilgarriff et al. (2001) describe the implementation of the linguistic search engine Word Sketch (2001) that displays collocations and dependencies acquired from a large corpus such as the BNC. Word Sketch is not as flexible as typical search engines, only supporting a fixed set of *queries*.

Recently, researchers have been attempting to go one step further and work with Web scale datasets, but it is difficult for an academic institute to crawl a dataset that is on par with the datasets built by search engine companies. In 2006, Google released the *Web 1T* for several major languages of the world (trillion-word n-gram datasets for English, Japanese, Chinese, and ten European languages), to stimulate NLP research in many areas. In 2008, Chang described a prototype that enhances *Google Web 1T* bigrams with PoS tags and supports search in the dataset by wildcards (wild-PoS), to identify recurring collocations. Wu, Witten and Franken (2010) describe a more comprehensive system (FLAX) that combines filtered Google data with text examples from the BNC for several learning activities.

In a way similar to Chang (2008) and Wu, Witten and Franken (2010), Stein, Potthast, and Trenkmann (2010) describe the implementation and application of *NetSpeak*, a system that provides quick access to the Google Web 1T n-gram with RE-like queries (alternator "|", one arbitrary word "*", arbitrary number of words between two specified words "…"). In contrast to *Linggle*, *NetSpeak* does not support PoS wildcard or conceptual clustering.

An important function in both *Linggle* and *NetSpeak* is synonym query. *NetSpeak* uses WordNet (Fellbaum 2010) synsets to support synonym match. But WordNet synsets tend to contain very little synonyms, leading to poor coverage. Alternatively, one can use the distributional approach to similarity based on a very large corpus. Lin and Pantel (2002) report efforts to build a large repository of dependencies extracted from large corpora such as Wikipedia, and provide similarity between words (demo.patrickpantel.com). We use these results both for handling synonym queries and to organize the n-grams into semantic classes.

More recently, Ritter and Etzioni (2010) propose to apply an LDA model (Blei et al. 2003) to

the problem of inducing selectional preference. The idea is to consider the verbs in a corpus as the documents of a traditional LDA model. The arguments of the verb that are encountered in the corpus are treated as the words composing a document in the traditional model. The model seems to successfully infer the semantic classes that correspond to the preferred arguments of a verb. The topics are semi-automatically labeled with WordNet classes to produce a repository of human interpretable class-based selectional preference. This choice might be due to the fact that if most LDA topic heads are usually reasonable upon human inspection, some topics are also incoherent (Newman 2010) and lower frequency words are not handled as successfully. We control the coherence of the topics and rearrange them into human interpretable clusters using a distributional similarity measure.

Microsoft Sempute Project (Sempute Team 2013) also explores core technologies and applications of semantic computing. As part of Sempute project, *NeedleSeek* is aimed at automatically extracting data to support general semantic Web searches. While *Linggle* focuses on n-gram information for language learning, *NeedleSeek* also uses LDA to support question answering (e.g., *What were the Capitals of ancient China?*) .

In contrast to the previous research in Web scale linguistic search engines, we present a system that supports queries with keywords, wildcard words, POS, synonyms, and additional regular expression (RE) operators and displays the results according the count, similarity, and topic with clusters of synonyms or conceptually related words. We exploit and combine the power of both LDA analysis and distributional similarity to provide meaningful semantic classes that are constrained with members of high similarity. Distributional similarity (Lin 1998) and LDA topics become two angles of attack to view language usage and corpus patterns.

3 Linggle Functionalities

The syntax of *Linggle* queries involves basic regular expression of keywords enriched with wildcard PoS and synonyms. *Linggle* queries can be either pattern-based commands or natural language questions. The natural language queries are currently handled by simple string matching based on a limited set of questions and command pairs provided by a native speaker informant.

3.1 Natural language queries

The handling of queries formulated in natural language has been implemented with handcrafted patterns refined from a corpus of questions found on various websites. Additionally, we asked both native and non-native speakers to use the system for text edition and to write down all the questions that arise during the exercise.

Linggle transforms a question into commands for further processing based on a set of canned texts (e.g., "How to describe a beach?" will be converted to "$A beach"). We are in the process of gathering more examples of language-related question and answer pairs from **Answers.com** to improve the precision, versatility, and coverage.

3.2 Syntax of queries

The syntax of the patterns for n-grams is shown in Table 1. The syntax supports two types of query functions: basic keyword search with regular expression capability and semantic search.

Basic search operators enable the users to query zero, one or more arbitrary words up to five words. For example, the query "*set off ... $N*" is intended to search for all nouns in the right context of *set off*, within a maximum distance of three words.

In addition, the "?" operator in front of a word represents a search for n-grams with or without the word. For example, a user wanting to determine whether to use the word *to* between *listen* and *music* can formulate the query "*listen ?to music*."

Yet another operation "|" is provided to search for information related to word choice. For example the query "build | construct ... dream" can be used to reveal that people *build* a dream much more often than they *construct* a dream.

A set of PoS symbols (shown in Table 2) is defined to support queries that need more precision than the symbol *. More work might be needed to resolve PoS ambiguity for n-grams. Currently, any word that has been labeled with the requested PoS in the BNC more than 5% of the time is displayed.

The "+" operator is provided to support semantic queries. Placed in front of a word, it is intended to search for synonyms in the context. For example the query "+sandy beach" would generate *rocky beach*, *stony beach*, *barren beach* in the top three results. The query "+abandoned beach" generates *deserted, destroyed and empty beach* at the top of the list. To support conceptual clustering of collocational n-grams, we need to

identify synonyms related to different senses of a given word. Table 3 shows an example of the result obtained for the ambiguous word *bank* as a unigram query. We can see the two main senses of the word (*river bank* and *institution*) as clusters.

Operators	Description
*	Any Word
?	With/without the word
…	Zero or more words
|	Alternator
$	Part of speech
+	Synonyms

Table 1: Operators in the Linggle queries

Part of speech	Description
N	Noun
V	Verb
A	Adjective
R	Adverb
PP	Preposition
NP	Proper Noun
PR	Pronoun
D	Determiner

Table 2: Part-of-speech in the Linggle queries

A *cluster* button on the interface activates or cancels conceptual clustering. When *Linggle* is switched into a cluster display mode, adjective-nouns, verb-objects and subject-verb relations can be browsed based on the induced conceptual clusters (see Figure 1).

The New York Times Example Base

In order to display complete sentence examples for users, the New York Times Corpus sentences are indexed by word. When the user searches for words in a specific syntactic relation, morphological query expansion is performed and patterns are used to increase both the coverage and the precision of the provided examples. For example, the bi-gram *kill bacteria* will be associated with the example sentence "*The bacteria are killed by high temperatures*.".

3.3 Semantic Clusters

Two types of semantic clusters are provided in Linggle: selectional preference and clusters of synonyms. Selectional preference expresses for example that an *apple* is more likely to be *eaten* or *cooked* than to be *killed* or *hanged.* Different classes of arguments for a predicate (or of predicates for an argument) can be found automatically. The favorite class of objects for the verb *drink* is LIQUID with the noun *water* ranked at the top. Less frequent objects belonging to the same class include *liquor* in the tail of the list. We aim at grouping arguments and predicates into semantic clusters for better readability.

valley mountain river lake hill bay plain north ridge coast city district town area community municipality country village land region
route highway road railway bridge crossing canal railroad junction
stream creek tributary

organization business institution company industry organisation agency school department university government court board
channel network affiliate outlet
supplier manufacturer distributor vendor retailer investor broker provider lender owner creditor shareholder customer employer

Table 3: First two level-one clusters of synonyms for the word "*bank*"

We produce clusters with a two-layer structure. Level one represents loose topical relatedness roughly corresponding to broad domains, while level two is aimed at grouping together closely similar words. For example, among the objects of the verb *cultivate*, the nouns *tie* and *contact* belong to the same level-two cluster. *Attitude* and *spirit* belong to another level-two cluster but both pairs are in the same level-one cluster. The nouns *fruit* and *vegetable* are clustered together in another level-one cluster. This double-layer representation is a solution to express at once close synonymy and topic relatedness. The clusters of symonyms displayed in Table 3 follow the same representation.

4 Implementation of the system

In this section, we describe the implementation of *Linggle*, including how to index and store n-grams for a fast access (Section 4.1) and construction of the LDA models (Section 4.2). We will describe the clustering method in more details in section 5.

4.1 N-grams preprocessing

The n-grams are first filtered keeping only the words that are in WordNet and in the British National Corpus, and then indexed by word and position in the n-gram, in a way similar to the rotated n-gram approach proposed by Lin et. al. (2010). The files are then stored in an Apache

HBase NoSQL base. The major advantages of using a NoSQL database is the excellent performance in querying the ability of storing large amounts of data across several servers and the capability to scale up when we have additional entries in the dataset, or additional datasets to add to the system.

4.2 LDA models computations

Two types of LDA models are calculated for *Linggle*. The first type is a selectional preference model between heads and modifiers. Six models are calculated in total for the subject-verb, the verb-object and the adjective-noun relations done in a similar way to Ritter and Etzioni's (2010) model with binary relations instead of triples. The second is a word/synonyms model in which a word is considered as a document in LDA and its synonyms as the words of the document. This second model has the effect of splitting the synonyms of a word into different topics, as shown in Table 3.

Seeds parameter: s_1
1. Consider the m first topics for a verb v according to the LDA per document-topic distribution (θ) 2. Consider $S = o_1,...,o_n$, a set of n objects of v. 3. Split S into m classes $C_1,..,C_m$ according to their LDA per topic-word probability: o_i is assigned to the topic in which it has the highest probability. 4. For each class C_i, move every object o_j that is not similar to any other o_k *of* C_i, *according to a* similarity threshold s_1 into a new created class.
Level 2 parameter: s_2
While ($Argmax_{ci,cj}\ Sim(c_i, c_j) > s_2$): Merge $Argmax_{ci,cj}\ Sim(c_i, c_j)$ into one class.
Level 1 parameter: s_3
While ($Argmax_{ci,cj}\ Sim(c_i, c_j) > s_3$): Group $Argmax_{ci,cj}\ Sim(c_i, c_j)$ under the same level 1 cluster.

Table 4: Clustering Algorithm for the object of a given verb

The hyperparameters *alpha, eta,* that affect the sparsity of the document-topic (*theta*) and the topic-word (*lambda*) distributions are both set to 0.5 and the number of topics is set to 300. More research would be necessary to optimize the value for the parameters in the perspective of the clustering algorithm, as quickly discussed in the next section.

Sim (c_i, c_j):
1. Build the Cartesian product $C = c_i \times c_j$ 2. Get P the set of the similarity between all word pairs in C 3. Return Sim(c_i,c_j) the mean of the scores in P

Table 5: Similarity between two classes t_i and t_j

5 Clustering algorithm

The clustering algorithm combines topic modeling results and a semantic similarity measure. We use Pantel's dependencies repository to compute LDA models for subject-verbs, verbs-objects and adjective-nouns relations in both directions. Currently, we also use Pantel's similarity measure. It has a reasonable precision partly because it relies on parser information instead of bag of words windows. However the coverage of the available scores is lower than what would be needed for Linggle. We will address this issue in the near future by extending it with similarity scores computed from the n-grams.

We combine the two distributional semantics approaches in a simple manner inspired by clustering by committee algorithm (CBC). The similarity measure is used to refine the LDA topics and to generate finer grain clusters. Conversely, LDA topics can also be seen as the seeds of our clustering algorithm.

This algorithm intends to constrain the words that belong to a final cluster more strictly than LDA does in order to obtain clearly interpretable clusters. The exact same algorithm is applied to synonym models, for synonyms of nouns, adjectives and verbs (shown in Table 3).

Table 4 shows the algorithm for constructing double layer clusters for a set S of objects of a verb v. The objects are first roughly split into classes, attributing a single topic to every object o_i. The topic of a word o_i is determined according to its per topic-word probability. More experiments could be done using the product of the per document-topic and the per topic-word LDA probabilities instead, in order to take into account the specific verb when assigning a topic to the object. Such a way of assigning topics should also be more sensitive to the LDA hyperparameters.

At this stage, some classes are incoherent and that low frequency words that do not appear in the head of any topic are often misclassified. Words are rearranged between the classes and create new classes if necessary using the similarity measure. If any word of a class is not simi-

lar to any other word in this class (the threshold is set to $s_1 = 0.09$), a new class is created for it.

Any two classes are then merged if their similarity (computer accordingly to Table 5) is above s_2=0.06, forming the level 2 clusters. Classes are then grouped together if the similarity between them is above $s_3 = 0.02$ forming the level 1 clusters.

Finally, the classes that contain less than three words are not displayed in *Linggle* and the predicate-arguments counts in the *Web 1T* are retrieved using a few hand crafted RE and morphological expansion of the nouns and the verbs.

This algorithm appears to generate interpretable semantic classes and to be quite robust regarding the threshold parameters. More tests and rigorous evaluation are left to future work.

6 Conclusion

There are many different directions in which Linggle will be improved. The first one is to allow users to work with word forms and with multiword expressions. The second one concerns the extension of the coverage of the example base with several large corpora such as Wikipedia and the extension of the coverage of the similarity measure. The third direction concerns the development of automatic suggestions for text edition, such as suggesting a better adjective or a different preposition in the context of a sentence. Finally, Linggle is currently being extended to Chinese.

We presented a prototype that gives access to Web Scale collocations. Linggle displays both word usage and word similarity information. Depending on the type of the input query, the results are displayed under the form of lists or clusters of n-grams. The system is designed to become a multilingual platform for text edition and can also become a valuable resource for natural language processing research.

References

David Blei, A. Ng, and M. Jordan. 2003. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, January 2003.

Jason S. Chang, 2008. Linggle: a web-scale language reference search engine. *Unpublished manuscript*.

Katrin Erk and Sebastian Padó. 2010. A Flexible, Corpus-Driven Model of Regular and Inverse Selectional Preferences. *In Proceedings of ACL 2010.*

Christiane Fellbaum. 2010. WordNet. *MIT Press*, Cambridge, MA.

John Rupert Firth. 1957. The Semantics of Linguistics Science. *Papers in linguistics* 1934-1951. London: Oxford University Press.

William H Fletcher. 2012. Corpus analysis of the world wide web." In The Encyclopedia of Applied Linguistics.

Adam Kilgarriff , and David Tugwell. 2001. Word sketch: Extraction and display of significant collocations for lexicography. In *Proceedings of COLLOCTION: Computational Extraction, Analysis and Exploitation workshop, 39th ACL and 10th EACL*, pp. 32-38.

Dekang Lin. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of the 17th international conference on Computational linguistics*, volume 2. Association for Computational Linguistics, pp. 768-774.

Dekang Lin, and Patrick Pantel. 2002. Concept Discovery from Text**.** In Proceedings of Conference on Computational Linguistics (COLING-02)**.** pp. 577-583. Taipei, Taiwan.

Dekang Lin, Kenneth Ward Church, Heng Ji, Satoshi Sekine, David Yarowsky, Shane Bergsma, Kailash Patil, Emily Pitler, Rachel Lathbury, Vikram Rao, Kapil Dalwani, Sushant Narsale. 2010. New tools for web-scale n-grams. In Proceedings of LREC.

David Newman, Jey Han Lau, Karl Grieser and Timothy Baldwin (2010). Automatic Evaluation of Topic Coherence. In *Proceedings of Human Language Technologies, 11th NAACL HLT*, Los Angeles, USA, pp. 100—108.

Evan Sandhaus. 2008. "New york times corpus: Corpus overview." LDC catalogue LDC2008T19.

Sempute Team. 2013. What is NeedleSeek? http://needleseek.msra.cn/readme.htm

Benno Stein, Martin Potthast, and Martin Trenkmann. 2010. Retrieving customary Web language to assist writers. Advances in Information Retrieval. Springer Berlin Heidelberg, pp. 631-635.

Martin Potthast, Martin Trenkmann, and Benno Stein. Using Web N-Grams to Help Second-Language Speakers .2010. SIGIR 10 Web N-Gram Workshop, pages 49-49.

Alan Ritter, Mausam, and Oren Etzioni. 2010. A Latent Dirichlet Allocation method for Selectional Preferences. *In Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics* (July 2010), pp. 424-434.

Yorick Wilks. 1978. Making preferences more active. *Artificial Intelligence* 11(3), pp. 197-223.

Shaoqun Wu, Ian H. Witten and Margaret Franken (2010). Utilizing lexical data from a web-derived corpus to expand productive collocation knowledge. ReCALL, 22(1), 83–102.