Improving the Recognizability of Syntactic Relations for Information Retrieval Interfaces

First Author
Affiliation / Address line 1
Affiliation / Address line 2
Affiliation / Address line 3
email@domain

Second Author
Affiliation / Address line 1
Affiliation / Address line 2
Affiliation / Address line 3
email@domain

Abstract

A common task in the humanities and social sciences, as well as in qualitative data analysis, is to characterize the usage of a word or concept. The ability to issue queries over syntactic relations between words is useful in these situations, as it allows users to see other words that have significant relationships with a query word. Previous interfaces for searching over syntactic structures require programming-style queries and formal linguistic terminology, but programming experience and linguistic expertise are scarce. Interfaces for non-experts must therefore be able to show options for searching for different syntactic relations in ways that allow the user to recognize the relation that is being queried. We therefore investigated how to present syntactic relations between words in a recognizable way. An experiment with 400 participants found that syntactic relations are recognized with 34% higher accuracy when contextual examples are shown, than a baseline of naming the relations alone. This indicates that more user-friendly query interfaces for syntactic search should augment the shown options with contextual examples.

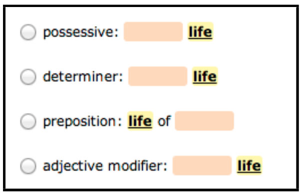
1 Introduction

The ability to search over grammatical relationships between words could be useful in many fields. For example, a social scientist trying to characterize different perspectives on immigration might ask how adjectives applying to 'immigrant' have changed in the last 30 years. A scholar interested in gender might search a collection to find out whether different nouns enter into possessive relationships with 'his' and 'her' (Anonymous and Anonymous, 2013). In other fields, grammatical queries can be used to develop patterns for recognizing entities in text, such as medical terms (Hirschman et al., 2005; MacLean and Heer, 2013), and products and organizations (Culotta and McCallum, 2005), and for coding qualitative data such as survey results.

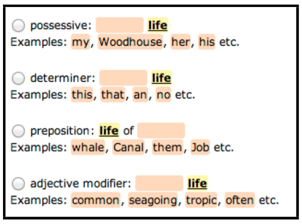
Most existing interfaces for *syntactic search* (querying over grammatical and syntactic structures) require complex program-like syntax. For example, the popular Stanford Parser includes Tregex, which allows for sophisticated regular expression search over syntactic tree structures. and Tsurgeon, which allows for manipulation of the trees extracted with Tregex (Levy and Andrew, 2006). The Finite Structure Query tool for querying syntactically annotated corpora requires its queries to be stated in first order logic (Kepser, 2003). In the Corpus Query Language (Jakubicek et al., 2010), a query is a pattern of attribute-value pairs, where values can include regular expressions containing parse tree nodes and words. Several approaches have adopted XML representations and the associated query language families of XPATH and SPARQL. For example, LPath augments XPath with additional tree operators to give it further expressiveness (Lai and Bird, 2010).

However, programming experience among potential users is scarce. One survey found that even though linguists wished to make very technical linguistic queries, 55% of them did not know how to program (Soehn et al., 2008). According to another survey (Gibbs and Owens, 2012), humanities scholars and social scientists are frequently skeptical of digital tools, because they are often difficult to use. This reduces the likelihood that existing structured-query tools for syntactic search will be usable by non-programmers.

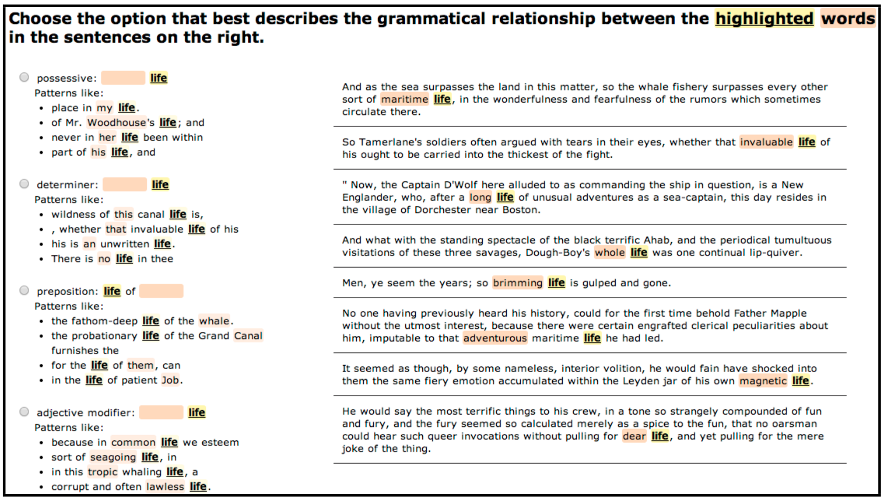
A related approach is the query-by-example work seen in the past in interfaces to database



(a) The options as they appear in the *baseline* condition.



(b) The same options as they appear in the *words* condition.



(c) The same options in the *phrases* condition, shown as they appeared in an identification task for the relationship `amod(life, ___)` (where different adjectives modify the noun 'life'). The correct answer is 'adjective modifier' (4th option), and the remaining 3 options are distractors.

Figure 1: The appearance of the choices shown in the three experiment conditions.

systems (Androutsopoulos et al., 1995). For instance, the Linguist's Search Engine (Resnik et al., 2005) uses a query-by-example strategy in which a user types in an initial sentence in English, and the system produces a graphical view of a parse tree as output, which the user can alter. The user can either click on the tree or modify the LISP expression to generalize the query. SPLICR also contains a graphical tree editor tool (Rehm et al., 2009). According to Shneiderman and Plaisant (Shneiderman and Plaisant, 2010), query-by-example has largely fallen out of favor as a user interface design approach. A downside of QBE is that the user must manipulate an example to arrive at the desired generalization.

At the same time, a related technique, auto-suggest, has become a widely-used approach in search user interfaces with strong support in terms of its usability (Hearst, 2009). A list of selectable options is shown under the search bar, filtered to be relevant as the searcher types. Searchers can recognize and select the option that matches their information need, without having to generate the query themselves.

The success of auto-suggest depends upon showing users options they can recognize. However, we know of no prior work applying auto-suggest to syntactic search. This leaves open the question of how syntactic relationships should be presented to make them recognizable. The problem is that there are no common-language terms for grammatical relationships even though speakers are capable of understanding and using them. Dependency parsers use linguistic terminology to label their outputs, but those names and definitions may not accessible to those outside the field. Take the phrases "he threw the ball" and "the ball was thrown by him". In both cases, it is clear that 'he' is is the one who 'threw' whereas, in grammatical terms, the first phrase is in the active case, and the phrase is in the passive case. The Stanford Dependency Parser (Klein and Manning, 2003), for example, outputs two different variants of the verb-subject relation: `nsubj(he, threw)` and `nsubjpass(he, threw)`. A more user-friendly grammatical search system would have to bridge the gap between the way that relations are recognized by people and the way that relations are represented in a parse.

One current presentation (not used with auto-suggest) is to name the relation and show blanks where the words that satisfy it would appear as in *X is the subject of Y* (Anonymous and Anonymous, 2013); we used this as the baseline presentation in our experiments because it employs the relation definitions found in the Stanford Dependency Parser's manual. Following the principle of recognition over recall, we hypothesized that showing contextualized usage examples would make the relations more recognizable.

We gave participants a series of identification tasks. In each task, they were shown a list of sentences containing a particular syntactic relationship between highlighted words. They were asked to identify the relationship type from a list of four options. We presented the options in three different ways, and compared the accuracy.

We chose Amazon's Mechanical Turk (MTurk) crowdsourcing platform as a source of study participants. The wide range of backgrounds provided by MTurk is desirable because our goal is to find a representation that is understandable to most people, not just linguistic experts or programmers.

Our results confirm that showing examples in the form of words or phrases significantly improves the accuracy with which grammatical relationships are recognized over a standard baseline. Our findings also showed that clausal relationships, which span longer distances in sentences, benefited significantly more from example phrases than either of the other treatments.

These findings suggest that a query interface in which a user enters a word of interest and the system shows candidate grammatical relations augmented with examples from the text will be more successful than the baseline of simply naming the relation and showing gaps where the participating words appear.

2 Experiment

Our hypothesis was:

> Grammatical relations are identified more accurately when shown with examples of contextualizing words or phrases than without.

To test it, participants were given a series of identification tasks. In each task, they were shown a list of 8 sentences, each containing a particular relationship between highlighted words. They were asked to identify the relationship from a list of 4 choices. Additionally, one word was chosen as a *focus word* that was present in all the sentences, to make the relationship more recognizable ("life" in Figure 1).

The choices were displayed in 3 different ways (Figure 1). The **baseline** presentation (Figure 1a) named the linguistic relation and showed a blank space with a pink background for the varying word in the relationship, the focus word highlighted in yellow and underlined, and any necessary additional words necessary to convey the relationship (such as "of" for the prepositional relationship "of", the third option).

The **words** presentation showed the baseline design, and in addition beneath was the word "Examples:" followed by a list of 4 example words that could fill in the pink blank slot (Figure 1b). The **phrases** presentation again showed the baseline design, beneath which was the phrase "Patterns like:" and a list of 4 example phrases in which fragments of text including both the pink and the yellow highlighted portions of the relationship appeared (Figure 1c).

We used a between-subjects design. The task order and the choice order were not varied: the only variation between participants was the presentation of the choices. To avoid the possibility of guessing the right answer by pattern-matching, we ensured that there was no overlap between the list of sentences shown, and the examples shown in the choices as words or phrases. Not every relationship shown in the distractors was tested, and distractors were chosen to be ones intuitively most likely to be mistaken for the relationship shown.

The tasks were generated using the Stanford Dependency Parser (De Marneffe et al., 2006) on the text of *Moby Dick* by Herman Melville. We tested the 12 most common grammatical relationships in the novel in order to cover the most content and to be able to provide as many real examples as possible. These relationships fell into two categories, listed below with examples.

Clausal or long-distance relations:

– Adverbial clause: ***I walk*** *while* ***talking***
– Open clausal complement: ***I love*** *to* ***sing***
– Clausal complement: *he* ***saw*** *us* ***leave***
– Relative clause modifier: *the* ***letter I wrote*** *reached*

Non-clausal relations:

– Subject of verb: ***he threw*** *the ball*
– Object of verb: *he* ***threw*** *the* ***ball***
– Adjective modifier ***red ball***
– Preposition (in): *a* ***hole*** *in a* ***bucket***
– Preposition (of): *the* ***piece*** *of* ***cheese***
– Conjunction (and) ***mind*** *and* ***body***
– Adverb modifier: *we* ***walk slowly***
– Noun compound: ***Mr. Brown***

We tested each of these relations 4 times, with 2 different focus words in each role. For example, the *Subject of Verb* relation was tested in the following forms:

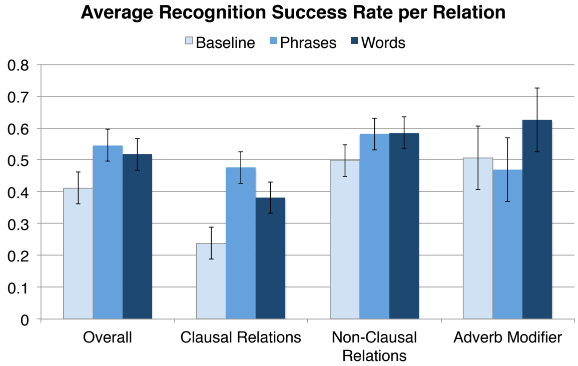


Figure 2: Recognition rates for different types of relations under the 3 experiment conditions, with 95% confidence intervals.

– `(Ahab, ___)`: the sentences each contained 'Ahab', highlighted in yellow, as the subject of different verbs highlighted in pink.
– `(captain, ___)`
– `(___, said)`: the sentences each contained the verb 'said', highlighted in yellow, but with different subjects, highlighted in pink.
– `(___, stood)`

To maximize coverage, yet keep the total task time reasonable (average 6.8 minutes), we divided the relations above into 4 task sets of 3 relations each. Each relation was tested with 4 different words, making a total of 12 tasks per participant.

2.1 Participants

400 participants completed the study distributed randomly over the 4 task sets and the 3 presentations. Participants were paid 50c (U.S.) for completing the study, with an additional 50c bonus if they correctly identified 10 or more of the 12 relationships. They were informed of the possibility of the bonus before starting.

To gauge their syntactic familiarity, we also asked them to rate how familiar they were with the terms 'adjective' (88% claimed they could define it), 'infinitive' (43%), and 'clausal complement' (18%). To help ensure the quality of effort from participants, we included a multiple-choice screening question, "What is the third word of this sentence?" Those that answered incorrectly were eliminated.

2.2 Results

The results (Figure 2) confirm our hypothesis. Participants in conditions that showed examples (**phrases** and **words**) were significantly more accurate at identifying the relations than participants in the **baseline** condition. We used the Wilcoxson signed-rank test, an alternative to the standard T-test that does not assume samples are normally distributed. The average success rate in the **baseline** condition was 41%, which is significantly less accurate than **words**: 52%, (p=0.00019, W=6136), and **phrases**: 55%, (p=0.00014, W=5546.5).

Clausal relations operate over longer distances in sentences, and so it is to be expected that showing longer stretches of context would perform better in these cases; that is indeed what the results showed. Phrases significantly outperformed words and baseline for clausal relations. The average success rate was 48% for **phrases**, which is significantly more than **words**: 38%, (p=0.017 W=6976.5) and **baseline**: 24%, (p=1.9×10^{-9} W=4399.0), which was indistinguishable from random guessing (25%). This is a strong improvement, given that only 18% of participants reported being able to define 'clausal complement'.

For the non-clausal relations, there was no significant difference between **phrases** and **words**, although they were both overall significantly better than the baseline (words: p=0.0063 W=6740, phrases: p=0.023 W=6418.5). Among these relations, adverb modifiers stood out (Figure 2), because evidence suggested that **words** (63% success) made the relation more recognizable than **phrases** (47% success, p=0.056, W=574.0) – but the difference was only almost significant, due to the smaller sample size (only 96 participants encountered this relation). This may be because the words are the most salient piece of information in an adverbial relation – adverbs usually end in 'ly' – and in the phrases condition the additional information distracts from recognition of this pattern.

3 Discussion

The results imply that auto-suggest interfaces for syntactic search should show candidate relationships augmented with a list of phrases in which they occur. A list of phrases is the most recognizable presentation for clausal relationships (34% better than the baseline), and is as good as a list of words for the other types of relations. A mockup of such a search interface is shown in Figure 3. Selecting the choice will return all sentences that contain the search term and match the relation.

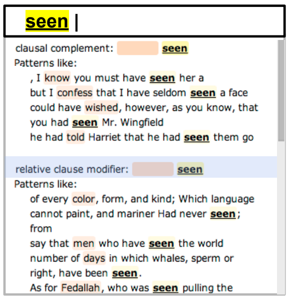


Figure 3: Mockup of auto-suggest for syntactic search on the word ‘seen’, showing clausal relations with example phrases.

There is a tradeoff between recognizability and space required for scrolling through the choices, although it is important to keep in mind that because the suggestions are populated with phrases from the collection itself, they are informative. Further, the suggestions can be ordered by frequency of occurrence in the collection, or by an interestingness measure given the search word. As the user becomes more familiar with a given relation, it may be expedient to shorten the cues shown, and then re-introduce them if a relation has not been selected after some period of time as elapsed.

The best strategy, **phrases**, had an overall success rate of only 55%, although the intended user base may have more familiarity with grammatical relations than the participants did, and therefore may perform better in practice. Nonetheless, there is room for improvement in scores, and it may be that additional visual cues, such as some kind of bracketing, will improve results. Furthermore, the current study did not test three-word relationships or more complex combinations of structures, and those may require improvements to the design.

References

I Androutsopoulos, GD Ritchie, and P Thanisch. 1995. Natural language interfaces to databases–an introduction. *Natural Language Engineering*, 1(01):29–81.

A Anonymous and B Anonymous. 2013. Anonymized for submission. —, –(–):–.

Aron Culotta and Andrew McCallum. 2005. Reducing labeling effort for structured prediction tasks. In *AAAI*, pages 746–751.

Marie-Catherine De Marneffe, Bill MacCartney, Christopher D Manning, et al. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of LREC*, volume 6, pages 449–454.

Fred Gibbs and Trevor Owens. 2012. Building better digital humanities tools. *DH Quarterly*, 6(2).

Marti Hearst. 2009. *Search user interfaces*. Cambridge University Press.

Lynette Hirschman, Alexander Yeh, Christian Blaschke, and Alfonso Valencia. 2005. Overview of biocreative: critical assessment of information extraction for biology. *BMC bioinformatics*, 6(Suppl 1):S1.

Milos Jakubicek, Adam Kilgarriff, Diana McCarthy, and Pavel Rychlỳ. 2010. Fast syntactic searching in very large corpora for many languages. In *PACLIC*, volume 24, pages 741–747.

Stephan Kepser. 2003. Finite structure query: A tool for querying syntactically annotated corpora. In *EACL*, pages 179–186.

Dan Klein and Christopher D Manning. 2003. Accurate unlexicalized parsing. In *Proc. ACL ’03*, volume 1, pages 423–430.

Catherine Lai and Steven Bird. 2010. Querying linguistic trees. *Journal of Logic, Language and Information*, 19(1):53–73.

Roger Levy and Galen Andrew. 2006. Tregex and tsurgeon: tools for querying and manipulating tree data structures. In *Proc. 5th Conference on Language Resources and Evaluation*, pages 2231–2234.

Diana Lynn MacLean and Jeffrey Heer. 2013. Identifying medical terms in patient-authored text: a crowdsourcing-based approach. *Journal of the American Medical Informatics Association*.

Georg Rehm, Oliver Schonefeld, Andreas Witt, Erhard Hinrichs, and Marga Reis. 2009. Sustainability of annotated resources in linguistics: A web-platform for exploring, querying, and distributing linguistic corpora and other resources. *Literary and Linguistic Computing*, 24(2):193–210.

Philip Resnik, Aaron Elkiss, Ellen Lau, and Heather Taylor. 2005. The web in theoretical linguistics research: Two case studies using the linguists search engine. In *Proc. 31st Mtg. Berkeley Linguistics Society*, pages 265–276.

Ben Shneiderman and Catherine Plaisant. 2010. *Designing The User Interface: Strategies for Effective Human-Computer Interaction, 5/e (Fifth Edition)*. Addison Wesley.

Jan-Philipp Soehn, Heike Zinsmeister, and Georg Rehm. 2008. Requirements of a user-friendly, general-purpose corpus query interface. *Sustainability of Language Resources and Tools for Natural Language Processing*, 6:27.