The AAAI 2017 Spring Symposium on
Computational Construction Grammar and Natural Language Understanding
Technical Report SS-17-02

Construction Detection in a Conventional NLP Pipeline

Jesse Dunietz
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15213, USA
jdunietz@cs.cmu.edu

Lori Levin
Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA 15213, USA
lsl@cs.cmu.edu

Miriam R. L. Petruck
International Computer Science Institute
University of California Berkeley
Berkeley, CA 94704, USA
miriamp@icsi.berkeley.edu

Abstract

This paper presents an approach to detecting constructions based on a conventional NLP pipeline: the **"constructions on top" approach** to integrating constructions into NLP, as opposed to "constructions all the way down." The approach is illustrated with the BECauSE corpus of causal language, the BECauSE constructicon, and the Causeway causal language detector, described elsewhere. We argue here that although BECauSE is not a full construction grammar, its lightweight design and compatibility with conventional NLP tools have facilitated progress on and insights into issues related to construction detection in news corpora. The issues we discuss are (1) individuating families of constructions, and (2) dealing with co-present, non-prototypical meanings that may be present alongside the prototypical meaning of a construction. Particularly significant is the observation that the BECauSE constructicon highlights the importance of integrating frame-evoking constructions into frame semantic resources such as FrameNet.

Introduction

Natural language processing (NLP) systems, following the linguistic tradition, conventionally treat linguistic structures as a stack of more or less modular units, from phonemes up through morphemes, words, syntax, and discourse. The problems arising from this approach are well-known: errors at one level propagate to the next, and linguistic patterns in the wild frequently break these abstraction barriers.

Construction grammar (CxG; Fillmore, Kay, and O'Connor, 1988; Fillmore, 1985, 1988) offers enormous potential for resolving these problems. It posits that linguistic patterns at any scale are paired with meanings, and those pairings combine to create utterances and their semantic representations. This obviates the need to define hard lines between morphemes, words, and syntax, allowing multi-word expressions (MWEs), idiosyncratic syntactic constructions, and productive morphology to flourish alongside the usual NLP categories. A construction-based system could handle all of these phenomena in a uniform manner.

Several projects have worked to realize this vision, building full NLP frameworks that are constructional through and through. The three best-known implementations are Embodied Construction Grammar (ECG; Bergen and Chang, 2005), Fluid Construction Grammar (FCG; Steels, 2012), and Sign-Based Construction Grammar (Boas and Sag, 2012), which uses the Head-driven Phrase Structure Grammar (HPSG) framework. Ideas from construction grammar have also been integrated into broad-coverage precision grammars like the English Resource Grammar (Copestake and Flickinger, 2000). All of these systems feature powerful parsing capabilities, and some feature generation, as well.

Copyright © 2017, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

(1) This ruling **opens the way for** broader legal protections.
(2) **For** the markets **to** work, banks can't expect bailouts.
(3) Judy's comments were **so** offensive **that** I left.
(4) We headed out **in spite of** the bad weather.
(5) We value any contribution, **no matter** its size.
(6) Strange **as** it seems, there's been a run of crazy dreams!
(7) You're **as** bad **as** my mom!
(8) **More** boys wanted to play **than** girls.
(9) Andrew is **as** annoying **as** he **is** useless.
(10) I'm poor**er than** I'd like.

Table 1: Examples of causal, concessive, and comparative language, with the tokens participating in the relevant construction bolded.

Despite the promise of CxG, so far these formalisms have seen relatively little uptake in the NLP community. There are many reasons for this state of affairs, including that CxG-based systems have not yet reached the level of maturity and robustness exhibited by conventional linguistic analysis systems. Another factor, though, seems to be the perceived barrier of rebuilding the entire NLP ecosystem based on "constructions all the way down." The NLP community already has mature, well-studied tools that identify words, parse them into syntactic structures, and assign semantic roles to pieces of those structures. Until fully constructional systems offer competitive robustness, researchers who need automated linguistic analysis are unwilling to leave the conventional tools aside. And in the short term, developing constructional tools for well-studied tasks seems like stepping backwards and retreading ground, even if doing so holds the potential for long-term advances.

However, a middle ground exists that leverages the key in-

sights of CxG to improve NLP in both the short and the long term. In several crucial semantic domains, the constructions that carry meaning are difficult to represent and automatically tag using conventional notions of words and grammar. In such cases, simple lexical analysis fails to capture the bulk of the semantics of even unremarkable sentences (Fillmore, Lee-Goldman, and Rhomieux, 2012). These constructions represent low-hanging fruit for incorporating CxG into NLP tools, while also laying the groundwork for further uptake.

For instance, consider the examples in Table 1 of causatives, concessives, and comparatives, all fundamental types of relations. Some are expressed via idiomatic multiword expressions (1, 4), which require treating multiple words as a unit. Others include sublexical elements (10) or consist of gappy patterns (7, 8, 9). And many depend on particular configurations of syntactic relations and slot-fillers (2, 3, 5, 6, 9), placing them closer to the grammatical end of the continuum of lexicon and grammar. Using a non-constructional NLP approach, recovering the semantic relationships that these constructions express is difficult, if not impossible.

Importantly, it is not necessary to rewrite the standard NLP pipeline to apply the key insights of CxG to these domains. We take those central insights to be:

1. Morphemes, words, MWEs, and grammar are all on the same spectrum of linguistic forms.
2. Any aspect or combination of those forms is equally capable of being mapped to meanings.

To accommodate these insights, all that needs to change in NLP is the representation for how meanings are attached to text. The gold-standard scheme for rich, domain-general semantic parsing is FrameNet (Ruppenhofer et al., 2016), which represents semantic frames as indicated by lexical unit (LU) "targets." But this is purely a matter of operational design decisions. As long as at least one relevant lexical or morphological span exists, the targets can be expanded without much trouble to allow richer, more flexible spans that capture constructions like those in Table 1. Such extensions, a longtime goal of FrameNet, have started to come to fruition in the FrameNet Constructicon (Fillmore, Lee-Goldman, and Rhomieux, 2012), a companion repository of grammatical information that characterizes constructions for cases when lexical analysis does not suffice.

With constructions linked directly to semantic frames, automatic taggers can rely on the usual robust part-of-speech taggers, dependency parsers, and so on to determine the presence of a given construction as well as its slot-fillers. This **"constructions on top" approach** to NLP gains much of the representational flexibility of constructions, while still retaining the ability to use existing NLP infrastructure.

In the long term, the "constructions on top" approach will serve as means of moving from current NLP systems to fully constructional tools. Tools built on this approach will enable researchers to more easily explore large corpora to understand what phenomena a full construction grammar will need to explain. Eventually, systems will be able to relax the constraint of requiring a specific lexical or morphological span as a semantics-bearing target, bringing non-lexicalized grammatical constructions within reach. And once constructions are firmly embedded as a lynchpin of semantic analysis, CxG techniques can more easily propagate back through the rest of the NLP pipeline.

This paper argues for the "constructions on top" approach as a short-term methodology for incorporating CxG into NLP, focusing on how it enhances frame-semantic parsing. After briefly reviewing FrameNet's lexicographic annotation practice, the paper discuss a recently created corpus and associated tagger (**(alias?)**) that demonstrate the usefulness of the approach for causal language, where constructional phenomena abound. The paper demonstrates the prevalence in the corpus of super-lexical causal constructions, and show that a pipeline based on conventional NLP tools recovers these constructions reasonably well. Finally, this paper discusses two major lessons learned for "constructions on top" systems, and perhaps for other CxG-based systems, as well.

FrameNet's Lexicographic Annotation Practice

Since the approach presented here derives from the FrameNet Constructicon project, an overview of FrameNet's lexicographic annotation practice (which informed construction annotation in the FN Constructicon) is in order.

FrameNet provides a catalog of semantic frames for English, where each frame describes some real-world concept or scenario (e.g., CAUSATION). Each frame specifies a number of related Frame Elements, or frame-specific roles that participate in the scenario (e.g., CAUSE, EFFECT). Each frame also includes a list of lexical units – i.e., words or word-like phrases – which the frame characterizes. Other "non-thematic roles" may exist in any frame, such as PURPOSE, which labels the intended purpose of the action profiled in a frame). Figure 1 shows an example annotation from the CAUSATION frame.

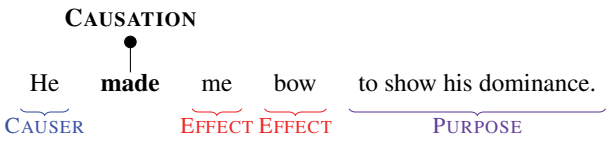

Figure 1: An example sentence annotated with FrameNet semantic annotations for the target *made*.

FrameNet characterizes LUs that are more than just individual lemmas; phrases that behave like words, such as *due to* and *bring about*, also constitute targets. Occasionally, targets are even discontinuous, as in verb/particle constructions like *give X up*. (This does not include prepositions that introduce arguments or adjuncts – e.g., *prevent me **from** going* – which FrameNet would not consider discontinuous target words, but rather part of the arguments that they introduce.) Typically, however, FrameNet does not consider a pattern of multiple content words to be a single target. It also excludes many function words (e.g., some prepositions, or words like *for* and *to* in ***for** market discipline **to** work*).

With the many overlapping frames that a sentence instantiates, FrameNet offers a rich, structured representation of

the sentence's semantic content. Indeed, its inclusion of multiword expressions ("words with spaces") as LUs even allows capturing (1) and (4) from Table 1. However, constructions that rely on discontinuous target words, function words, sublexical elements, or unlexicalized elements still present challenges. In fact, every other construction in Table 1 would either be impossible to capture in FrameNet's lexical unit framework or would require abusing the representation.

In the domain of causality, such constructions are common enough to constitute a serious gap in coverage. A newly developed corpus of causal constructions demonstrates this phenomenon.

Work on Constructions of Causal Language

The BECauSE Causal Language Corpus

Dunietz, Levin, and Carbonell (2015) developed a corpus of causal language, to be released as BECauSE (Bank of Effects and Causes Stated Explicitly; **(alias?)**). Using that corpus, we gathered statistics about the frequency of various types of constructions, focusing on those that FN does not consider LUs.

First, a brief overview of the corpus (see the papers for full details): The scheme specifically concerns language that appeals to psychological notions of cause and effect – i.e., what causal relationships the text asserts, not what causal relationships hold in the real world. For example, *cancer causes smoking* states a false causation, but it would nonetheless be annotated. In contrast, *bacon pizza is delicious* would not be annotated, even though bacon may in fact cause deliciousness, because the causal relationship is not stated as such.

The annotations are similar in spirit to FrameNet. Each instance of causal language is associated with a *connective* (much like FrameNet's lexical units), consisting of the lexical items in the construction that signal the causal relationship (e.g., *because of*). The connective annotation includes all words whose lemmas appear in every instance of the construction. This excludes elements that can be absent or whose lemmas can vary, such as copulas or determiners (e.g., the connective for *as a result* is the words *as* and *result*, because the determiner can vary). In addition, each annotation includes a cause and effect span unless one of the two has been syntactically eliminated, as in passives or infinitives. These spans are generally events or states of affairs, expressed as complete clauses or phrases.

The connective may be discontinuous and arbitrarily complex (e.g., *necessary condition of* or *if __ is to __*). BECauSE considers words that introduce arguments of a construction (e.g., *to* in *cause to*) to be part of the causal construction, rather than part of the arguments.

Annotators together established an informal constructicon to guide their decisions on what patterns are considered connectives. This constructicon contains about 150 constructions (the exact number is debatable, as discussed below). Note that a connective is not synonymous with a construction; rather, it is a lexical indicator of the presence of the construction, an anchor for the annotations. A full description of the construction would include the relevant parts of speech, syntactic relations, semantic constraints, etc.

The corpus itself consists of three sets of exhaustively annotated documents:

- 59 randomly selected articles from the year 2007 in the Washington section of the New York Times corpus (Sandhaus, 2008)
- 47 documents randomly selected from sections 2-23 of the Penn Treebank (Marcus et al., 1994)
- 679 sentences[1] transcribed from Congress' Dodd-Frank hearings, taken from the NLP Unshared Task in PoliInformatics 2014 (Smith et al., 2014)

The corpus contains a total of 4161 sentences, among which are 1099 labeled instances of causal language. 1004 of these, or 91%, include both cause and effect arguments.

An expanded version of the BECauSE corpus is currently under development (discussed further in the conclusion).

Constructional Phenomena are Frequent in the Corpus

To determine the prevalence and practical impact of constructional phenomena, it is instructive to examine how much of the BECauSE corpus is covered by FrameNet. Statistics on FrameNet's coverage of various types of constructions in BECauSE are shown in Table 2. A few notes on how these statistics were computed:

- The number/contiguity of words in a construction are determined by FrameNet standards: for prepositional arguments of the construction, the initial prepositions (e.g., *prevent **from***) do not count, even though BECauSE annotates them as part of the connective.
- In some cases, FrameNet includes a single-word lexical target that participates in the construction, but the FrameNet Constructicon would nonetheless record a construction with multiple "construction-evoking elements." One such example is *enough X for Z to Y*: FrameNet already has an LU for this use of *enough*. These are counted as multi-word constructions.
- The statistics were computed automatically, using sequence of connective tokens as an indicator of the causal construction. However, this method occasionally conflates gappy and non-gappy versions of a construction, such as *X is **sufficient to** Y* and ***sufficient** X **to** Y*. Thus, the numbers should not be taken as extremely precise.
- Constructions included in frames that are often but not always causal were counted as fully represented.
- A construction was counted as partially represented if significant portions of the construction were missing (other than argument-initial prepositions).
- A construction was counted as "represented isomorphically" if the causal relationship could be derived from FrameNet, but a causal frame would not be identified

[1] The remainder of the document was not annotated due to constraints on available annotation effort.

	Construction complexity			
FrameNet status	Single word	Contiguous words	Non-contiguous words	**Total by FN status**
Represented	618	63	16	697
Partially represented	0	17	21	38
Represented isomorphically	182	0	0	182
Missing LU	153	24	0	177
Not representable	0	0	5	5
Total by complexity	953	104	42	1099

Table 2: Instance counts for various types of constructions in BECauSE. See the text for details of how these were computed.

withou extra information linking specific roles to frames. For example, a purpose phrase like *I left **to get food*** would be labeled with a PURPOSE role, but this role is not explicitly connected to the PURPOSE frame.

Not surprisingly, FrameNet's coverage is good for the single-word connectives, representing 65% of them. Many of the remainder are simply LUs that have not yet been documented in FrameNet. (The bulk of the 182 single-word "represented isomorphically" instances are the word *to*, used in its purpose sense.) No single-word connectives are unrepresented or unrepresentable in FrameNet.

With multi-word connectives, however, constructional effects become apparent. At 61%, coverage of contiguous multi-word constructions is almost as good as that of single words. However, 16% of contiguous connectives cannot be represented as LUs without missing significant pieces of the construction. Non-contiguous constructions are relatively uncommon, but fully 62% of them cannot be represented as FrameNet LUs. In total, the instances that are partially or fully unrepresentable without a richer construction representation constitute nearly 4% of all annotations – an underestimate, considering that a significant number of the connectives not (yet) in FrameNet would be represented only partially.

That causation is not neatly or completely characterized with lexical units is no surprise. Indeed, other fundamental relation types exhibit similar behavior: concepts and components of meaning that are central to human thinking are squeezed into language ubiquitously and at all levels of linguistic structure. As noted in the discussion of Table 1, concessives and comparatives are two other areas where complex constructions abound, and doubtless there are many more.

The prevalence of constructional phenomena speaks to the need for enriching FrameNet's representations with a constructicon, as the FrameNet Constructicon has begun to do. The BECauSE lexicographic work could contribute to this, serving as a basis for causality-related entries in the FrameNet Construction. It might first be necessary to reorganize the causality-related frames as suggested by Vieu et al. (2016). Next, lexicographers would need to examine each construction and define its "construct elements." Automating this task would be difficult, since construct elements are defined by prose descriptions of constraints that would not be obvious to a machine, such as "an AP headed by *long*." However, with some human intuition, mapping the BECauSE representation into FrameNet Constructicon entries ought to be fairly straightforward, since both represent an instance of a construction as a sequence of fixed or semi-fixed elements and slot-fillers.

Causeway: A Constructicon-Based System for Tagging Causal Constructions

If a constructicon-based semantic representation could not be integrated with conventional NLP tools, it would be of little help for injecting CxG into NLP. Fortunately, constructions can indeed be tagged using input from conventional tools, as demonstrated by the Causeway system (**(alias?)**). Causeway tags causal language using the BECauSE annotation scheme.

Causeway is implemented as a four-stage pipeline, which is designed to overgenerate and then prune (see the paper for full implementation details):

1. **Pattern-based tentative connective discovery**. At training time, lexico-syntactic patterns[2] for connectives are extracted from the training data. At test time, these patterns are matched against each sentence to find tokens that *may* be participating in a causal construction. Each extracted pattern specifies the connective lemmas and parts of speech. The pattern also specifies the minimal dependency parse subtree that links the connective words and argument heads.
2. **Argument identification** marks the cause and effect spans by expanding argument heads to include most dependents.
3. **A statistical filter** to remove false matches. The filter performs a weighted vote between three probabilistic classifiers: a connective-specific logistic regression classifier, a global logistic regression classifier, and a connective-specific majority-class classifier.
4. **A constraint-based filter** to remove redundant connectives. For example, if *to* (which is causal in sentences like *I left to get lunch*) and *cause X to Y* both matched, the smaller of the two should be ignored.

Causeway Results Dunietz, Levin, and Carbonell (In press) report results from running Causeway with 20-fold

[2]The original Causeway paper reports on two different variants of this pipeline, one that uses lexical patterns for the first stage and one that uses syntactic patterns. The syntactic version achieves better end-to-end results, so we present that version here.

Pipeline	Connectives			Causes			Effects		
	P	**R**	$\mathbf{F_1}$	$\mathbf{S_C}$	$\mathbf{H_C}$	$\mathbf{J_C}$	$\mathbf{S_E}$	$\mathbf{H_E}$	$\mathbf{J_E}$
Pattern matching without classifier	7.3	**71.9**	13.2	65.0	84.3	39.3	30.4	63.0	30.7
Pattern matching + MFS	40.1	37.9	38.6	71.0	87.6	42.0	34.3	64.4	31.9
Pattern matching + MFS + SC filter	**60.9**	36.2	45.1	75.1	92.3	42.9	40.7	75.2	35.8
Pattern matching + voting classifier	51.9	47.6	49.4	68.7	86.9	39.9	38.0	72.5	34.1
Pattern matching + voting classifier + SC filter	57.7	47.4	**51.8**	67.1	84.4	39.0	37.7	70.7	33.4
Baseline	**88.4**	21.4	33.8	74.1	94.7	43.7	48.4	83.3	38.4
Baseline + full pipeline	59.6	**51.9**	**55.2**	67.7	85.8	39.5	39.5	73.1	34.2

Table 3: Experimental results for Causeway. S_C and S_E indicate exact span match for causes and effects, respectively; H_C and H_E indicate percentage accuracy for cause and effect heads; and J_C and J_E indicate cause and effect Jaccard indices, a measure of span overlap. "MFS" indicates a per-connective most-frequent-sense classifier. "SC filter" indicates the filter for smaller overlapping connectives. Note that argument scores between pipelines are not directly comparable, as they are only computed for connectives that were correct.

cross-validation on the BECauSE corpus. Their findings appear in Table 3. The baseline is a simple most-frequent-sense system; see the Causeway paper for details.

The most important implication of these results is that it is possible to tag constructions with a reasonable degree of accuracy based on conventional NLP tools. Although the metrics are not entirely comparable to those of regular semantic parsing tools, Causeway's scores are not terribly far behind. A secondary takeaway is that even fairly simple machine learning methods, based on pattern-matching and common classifiers, can learn much of what is needed to recognize constructions.

Of course, this work is only the first attempt at building a system for this task; future automated construction recognizers will no doubt be more sophisticated and more accurate. To that end, it is worth examining what kinds of errors Causeway makes. The original Causeway paper gives an extensive error analysis; here, we review a few notable points.

The biggest issue facing the system is low end-to-end recall. The initial pattern matching has high recall, but its precision is low (as expected). When the filters eliminate false matches, they do improve the F_1, but the filters are overly aggressive, dragging down recall too far. Examining the classifier scores reveals that the filter is correctly assigning low probability to negative instances. For positive instances, however, rather than clustering the scores on the high end, the classifier's probability estimates are more evenly distributed.

It would be particularly helpful to improve classification for a few simple but highly ambiguous connectives, such as *to*, *if*, *for*, and *so*. These collectively accounted for about two thirds of all misclassifications by the filter.

Interestingly, the system has more trouble with effects than with causes. This seems to be because causes tend to be subjects or nominal modifiers, which are short and therefore easier for the system to guess correctly. In contrast, effects are frequently primary clauses, complements, or direct objects, which tend to be more complex.

Lessons Learned From Constructicon-Based Annotation of Causal Language

In the process of examining the BECauSE corpus, two major issues with constructicon-based annotation became apparent: the problem of individuating constructions, which is particular to this style of applying CxG; and the problem of overlapping meanings, which raises concerns for computational CxG systems in general.

Individuating Constructions

Computational formalizations of CxG typically try to spell out a list of linguistic forms and inheritance relationships. Conceptually, this is similar to building a lexicon of words, except that the units are more structured, with arguments, semantic mappings, and constraints.

This approach does eliminate the problematic abstraction barriers that conventional NLP systems must contend with. However, building a constructicon for NLP use raises an equally serious problem of what to include as a construction.

As an example, consider the following examples of constructions where the extremity of a graded attribute leads to some result (variants of all of these were found in the corpus):

(11) ***too*** *sweet* ***to*** *eat*

(12) ***too*** *sweet* ***for*** *me* ***to*** *eat*

(13) *sweet* ***enough to*** *eat*

(14) *sweet* ***enough for*** *me* ***to*** *eat*

(15) *sweet* ***enough that*** *I can eat it*

(16) ***so*** *sweet* ***that*** *I can't eat it*

(17) ***so*** *sweet I can't eat it*

How many different constructional patterns are in play here? One option is simply to say that each example employs a separate construction. However, this approach is extremely unparsimonious: (13), (14), and (15) share an obvious *enough* ⟨*complement*⟩ structure; (13) is almost identical to (14), but with an optional argument removed, and likewise for (11) and (12); (17) is identical to (16) but without

the optional complementizer; and the *to* tokens in (11), (13), and (14) seems to perform identical functions.

The usual CxG answer is to construct a hierarchy of inheritance relationships that capture just the right generalizations. Such hierarchies have indeed been carefully constructed in other domains (e.g., Hasegawa et al., 2010). At some point, though, a product of multiple interacting constructions starts to become conventionalized, enabling the combined form to take on unique, non-compositional properties. How conventionalized must such a combination be before it is considered its own construction? Does *as a result* appear as a collocation often enough that it deserves its own constructicon entry, even though more compositional variants such as *as one result* occasionally show up?

In theory, the difficulty of individuating constructions should be particularly problematic for the "constructions on top" approach. After all, its entire *raison d'être* is to sidestep a full-fledged analysis of all the underlying constructions. But without the complete analysis, it is impossible to determine where one construction ends and another begins, or where multiple constructions are interacting. This approach avoids the work of the analysis, but it also cannot reap the benefits.

In practice, this issue is not too damning for "constructions on top"; the takeaway here for NLP constructicon developers is simply that the decisions must be made. The lines between constructions may be drawn semi-arbitrarily, but as long as they mostly match the right set of surface patterns, it should not matter whether a pattern is considered one construction or two, the implications for the theory notwithstanding. In fact, the lack of precise definition may even be a benefit: decisions can be made more on the grounds of convenience for machine learning algorithms than rigorous consistency and elegance.

This "quick-and-dirty" style of analysis also allows researchers to discover observations like the patterns in (11)-(17). These are the very observations that a fuller construction grammar will ultimately need to explain. The discovery of such organized observations is precisely the hope that Fillmore expressed in his introduction to the FrameNet Constructicon.

Multiple Overlapping Meanings

The second problem is a familiar one from word sense disambiguation: constructions often suggest meanings from multiple related semantic domains. In some cases, this is simply a matter of one meaning implying another. For instance, example (14) (the *enough for X to Y* construction) would be annotated in BECauSE as causal. However, FrameNet's LU inventory includes it in the SUFFICIENCY frame (under the LU *too*). While the concept of SUFFICIENCY does not link directly to a causation or enablement frame in FrameNet, the concepts of sufficiency and causality are not in conflict; sufficiency for a particular outcome implies enablement. Augmenting the FrameNet lexicon with relations between frames or frame elements would resolve this problem. Indeed, FrameNet already supports such additions.

The trickier cases are the ones displaying semantic drift into non-prototypical meanings. For example, imagine a speaker who wants to express that taking a drink caused her headache to lessen. She might say:

(18) My head was hurting, but taking a drink **made** it feel much better.

This example is clearly causal. But equally felicitous, perhaps even more so, would be to express the thought as:

(19) My head was hurting, but **after** I took a drink, it felt much better.

Obviously, *after* expresses a temporal relationship. However, because temporal and causal relations are so intertwined in our mental models of the world, temporal language is often borrowed to suggest causal relationships. Similar overlaps occur between causation and hypotheticals (***if*** *you touch it, it will fall over*), obligation and permission (*my father* ***let*** *me take a cookie*), creation and termination (*these reports* ***create*** *a perception of higher risk*), and many other relations that overlap with causation in the real world. In FrameNet terminology, each of these constructions can evoke either relevant frame, or both simultaneously.

In principle, multiple frames could be handled by positing a separate version of the construction for each possible combination of meanings, where the meaning side of the construction can be a conjunction if necessary. This is precisely what a constructicon must posit. As a practical matter, the conjunctions could be represented either explicitly, by listing all allowed combinations, or implicitly, by allowing multiple constructions with the same form to be active at once. Which option is preferable is a question of implementation priorities: elaborating each combination explicitly is unwieldy, but the implicit option makes it harder for the constructicon builder to specify what combinations are disallowed.

Whatever the implementation, the downside of treating overlapping meanings as separate constructions is that it still requires making binary decisions about precisely when a given meaning is present. In reality, as with word senses, there are often shades of meaning. The way *if* came to convey causation, for example, was by diachronic semantic drift, which presumably must have begun with hints of causal meaning that gradually became more prominent. Also, it seems profligate to posit a new construction every time a more radial instance of an existing construction appears.

A construction representation more consistent with CxG's cognitive roots would help considerably here. With some formal notion of a semantic space, it would be possible to formalize the concept of prototypes and radial categories (see, e.g., Lewandowska-Tomaszczyk, 2007). The meaning side of a construction would then be a distribution in that space. For example, the meaning of the *create* construction would be concentrated in the creation/termination realm, but with a spread that extends to a lesser degree into causation. The construction could then be instantiated with a meaning anywhere within the distribution, with the strength of the radial aspects determined by contextual factors and constraints. Such a continuous representation would even allow

novel uses of the construction at the boundaries of the meaning distribution.

Conclusions

The constructicon approach presented here is a way to inject CxG principles into NLP even before full-fledged construction grammar tools are ready for widespread use. As the BECauSE corpus and the Causeway tagger demonstrate, working with a constructicon of constructions tied directly to semantic frames offers insight and practical lessons for both constructional analysis and future NLP tools built on it.

An effort is currently underway to revise and extend BECauSE to indicate where relations that overlap with causality are also present. (It will also include about 500 additional sentences, and fix some quirks of the original annotation scheme.) To handle multiple meanings, this effort implements the implicit option of allowing multiple constructions with the same form to be active at once.

In the longer term, we hope that the CxG and NLP communities will work together to define more flexible representations for semantic frames. To that end, the constructicon approach can help with rapid experimentation. Ultimately, with fluidity in the representations of both form and meaning, CxG-based tools should outperform their non-constructional counterparts.

References

Bergen, B., and Chang, N. 2005. Embodied construction grammar in simulation-based language understanding. *Construction grammars: Cognitive grounding and theoretical extensions* 3:147–190.

Boas, H. C., and Sag, I. A. 2012. *Sign-based construction grammar*. CSLI Publications/Center for the Study of Language and Information.

Copestake, A. A., and Flickinger, D. 2000. An open source grammar development environment and broad-coverage English grammar using HPSG. In *Proceedings of the Second conference on Language Resources and Evaluation (LREC-2000)*.

Dunietz, J.; Levin, L.; and Carbonell, J. 2015. Annotating causal language using corpus lexicography of constructions. In *The 9th Linguistic Annotation Workshop held in conjuncion with NAACL 2015*, 188–196.

Dunietz, J.; Levin, L.; and Carbonell, J. In press. Automatically tagging constructions of causation and their slot-fillers. In *Transactions of the Association for Computational Linguistics*.

Fillmore, C. J.; Kay, P.; and O'Connor, M. C. 1988. Regularity and idiomaticity in grammatical constructions: The case of let alone. *Language* 64(3):501–538.

Fillmore, C. J.; Lee-Goldman, R.; and Rhomieux, R. 2012. The FrameNet constructicon. *Sign-Based Construction Grammar* 309–372.

Fillmore, C. J. 1985. Syntactic intrusions and the notion of grammatical construction. In *Annual Meeting of the Berkeley Linguistics Society*, volume 11, 73–86.

Fillmore, C. J. 1988. The mechanisms of construction grammar. In Axmaker, S.; Jaisser, A.; and Singmaster, H., eds., *Proceedings of the Fourteenth Annual Meeting of the Berkeley Linguistics Society*, volume 14, 35–55. Berkeley Linguistics Society.

Hasegawa, Y.; Lee-Goldman, R.; Ohara, K. H.; Fujii, S.; and Fillmore, C. J. 2010. On expressing measurement and comparison in English and Japanese. *Contrastive construction grammar* 169–200.

Lewandowska-Tomaszczyk, B. 2007. Polysemy, prototypes, and radial categories. *The Oxford handbook of cognitive linguistics* 139–169.

Marcus, M.; Kim, G.; Marcinkiewicz, M. A.; MacIntyre, R.; Bies, A.; Ferguson, M.; Katz, K.; and Schasberger, B. 1994. The Penn Treebank: Annotating predicate argument structure. In *Proceedings of the Workshop on Human Language Technology*, HLT '94, 114–119. Stroudsburg, PA, USA: Association for Computational Linguistics.

Ruppenhofer, J.; Ellsworth, M.; Petruck, M. R. L.; Johnson, C. R.; Baker, C.; and Scheffczyk, J. 2016. FrameNet II: Extended theory and practice.

Sandhaus, E. 2008. The New York Times annotated corpus. *Linguistic Data Consortium, Philadelphia*.

Smith, N. A.; Cardie, C.; Washington, A. L.; and Wilkerson, J. 2014. Overview of the 2014 NLP unshared task in poli-informatics. In *Proceedings of the ACL 2014 Workshop on Language Technologies and Computational Social Science*.

Steels, L. 2012. Computational issues in Fluid Construction Grammar. *Lecture Notes in Computer Science* 7249.

Vieu, L.; Muller, P.; Candito, M.; and Djemaa, M. 2016. A general framework for the annotation of causality based on FrameNet. In Calzolari, N.; Choukri, K.; Declerck, T.; Goggi, S.; Grobelnik, M.; Maegaard, B.; Mariani, J.; Mazo, H.; Moreno, A.; Odijk, J.; and Piperidis, S., eds., *Proceedings of LREC 2016*. Paris, France: European Language Resources Association (ELRA).