

# Semantic Hypergraphs

Telmo Menezes<sup>\*1</sup> and Camille Roth<sup>†1,2</sup>

<sup>1</sup>*Computational Social Science Team, Centre Marc Bloch Berlin (CNRS/HU), Friedrichstr. 191, 10117 Berlin, Germany*

<sup>2</sup>*CAMS (Centre Analyse et Mathématique Sociales, UMR 8557 CNRS/EHESS), 54 Bd Raspail, 75007 Paris, France*

## Abstract

Existing computational methods for the analysis of corpora of text in natural language are still far from approaching a human level of understanding. We attempt to advance the state of the art by introducing a model and algorithmic framework to transform text into recursively structured data. We apply this to news titles extracted from a social news aggregation website. We show that a recursive ordered hypergraph is a sufficiently generic structure to represent significant number of fundamental natural language constructs, with advantages over conventional approaches such as semantic graphs. We present a pipeline of transformations from the output of conventional NLP algorithms to such hypergraphs, which we denote as semantic hypergraphs. The features of these transformations include the creation of new concepts from existing ones, the organisation of statements into regular structures of predicates followed by an arbitrary number of entities and the ability to represent statements about other statements. We demonstrate knowledge inference from the hypergraph, identifying claims and expressions of conflicts, along with their participating actors and topics. We show how this enables the actor-centric summarization of conflicts, comparison of topics of claims between actors and networks of conflicts between actors in the context of a given topic. On the whole, we propose a hypergraphic knowledge representation model that can be used to provide effective overviews of a large corpus of text in natural language.

**Keywords:** Knowledge Representation; Knowledge Graphs; Information Extraction; Text Mining; Natural Language Understanding; Computational Social Science.

## 1 Introduction

The large-scale processing of textual corpora relies on two intertwined challenges: the extraction of information and its representation. On the extraction side, the portfolio of Natural Language Processing (NLP) has been enriched over the last few years by cutting-edge software aimed at conducting part-of-speech tagging, extracting

parse trees and carrying out lemmatization. Recent advances in machine learning and the increase in available datasets and computational power have been contributing to an increase in their quality and performance. For example, Google released SyntaxNet [3], which uses a neural network model to implement an NLP dependency parser [5]. This is one amongst several state-of-the-art NLP packages that became recently available. Another example is spaCy [17], that incorporates many of the latest advances available in the literature, and which we use as a basis for the experiments to be presented.

The availability of these tools facilitates the creation of systems capable of analyzing text in semantically rich ways. In turn, the issue of the representation of the extracted information is being dealt with a variety of frameworks. These approaches, whose general goal is to transform natural language documents into structured data that can be more easily analyzed by scholars, are usually referred to by the umbrella term “*text mining*”. They exhibit a wide range of sophistication, from simple numerical statistics to more elaborate machine learning algorithms. Some methods indeed rely essentially on scalar numbers, for instance by measuring text similarity (e.g., with cosine distance [36]) or attributing a valence to text, as in the case of sentiment analysis [28], which in practice may be used to appraise the emotional content of a text (anger, happiness, disagreement, etc.) or public sentiment towards political candidates in social media [42]. Similarly, political positions in documents may be inferred from Wordscores [20] – a popular method in political science that also relies on the summation of pre-computed scores for individual words, and has more refined elaborations, e.g. with Bayesian regularization [24]. Other methods preserve the level of words: such is the case with term and pattern extraction (i.e., discovering salient words through the use of helper measures like term frequency-inverse document frequency (TF-IDF) [30]), so-called “Named Entity Recognition” [26] (used to identify people, locations, organizations, and other entities mentioned in corpora, for example in Twitter streams [32]) and ad-hoc uses of conventional computer science approaches such as regular expressions to identify chunks of text matching against a certain pattern (for example, extracting all p-values from a collection of scientific articles [12]). An-

<sup>\*</sup>menezes@cmb.hu-berlin.de

<sup>†</sup>roth@cmb.hu-berlin.de

other strand of approaches operates at the level of word sets, including those geared at topic detection (such as Latent Dirichlet Allocation (LDA) [8] and TextRank [23], used to extract the topics addressed in a text) or used for relationship extraction (meant at deriving semantic relations between entities mentioned in a text, e.g., *is(Berlin, City)*).

Overall, these techniques provide useful approaches to analyze text corpora at a high level, for example, with regard to their main entities, relationships, sentiment, and topics. However, there is limited support to detect, for instance, more sophisticated claim patterns across a large volume of texts, what recurring statements are made about actors or actions, and what are the qualitative relationships among actors and concepts. This type of goal, for instance, extends semantic analysis to a socio-semantic framework [33] which also takes into account actors who make claims.

The need for such sophistication is all the more pregnant for social sciences. On the one hand, qualitative social science methods of text analysis do not scale to the enormous datasets that are now available. Furthermore, quantitative approaches allow for other types of analysis that are enriching and complementary to qualitative research, yet may simplify extensively the processing in such a way that it hinders their adoption by scholars used to the refinement of qualitative approaches. And the more sophisticated the NLP techniques become, the further they tend to be from being used for large-scale text analysis purposes. Indeed, these systems are fast and accurate enough to form a starting point for more advanced computer-supported analysis in the context of computational social sciences, and they enable approaches that are substantially more sophisticated than the text mining state of the art discussed above. Yet, the results of such systems may seem relatively trivial compared to human-level understanding of natural language.

The literature already features some works which attempt at going beyond simplistic bag-of-words or triplet language models. Statement Map [25] is aimed at mining the various viewpoints expressed around a topic of interest in the web. Here a notion of claim is employed. A statement provided by the user is compared against statements from a corpus of text extracted from various web sources. Text alignment techniques are used to match statements that are likely to refer to the same issue. A machine learning model trained over NLP-annotated chunks of text classifies pairs of claims as “agreement”, “conflict”, “confinement” and “evidence”. The subfield of argumentation mining also makes extensive use of machine learning and statistical methods to extract portions of text correspond to claims, arguments and premises, while generally relying on surface linguistic features that rarely deal with structured and relational data [19]. Already in 2008, van Atteveldt and Kleinnijenhuis [40] proposed a system to extract binary semantic

relationships from Dutch newspaper articles. A recent work [34] presents a system aimed at analysing claims in the context of climate negotiations. It leverages dependency parse trees and general ontologies [39] to extract tuples of the form:

$\langle \text{actor}, \text{predicate}, \text{negotiation\_point} \rangle$

where the actors are stakeholders (e.g., countries), the predicates express agreement, opposition or neutrality and the negotiation point is identified by chunk of text. Similarly, in another recent work [41], parse trees are used to automatically extract source-subject-predicate clauses in the context of news reporting over the 2008-2009 Gaza war, and used to show differences in citation and framing patterns between U.S. and Chinese sources.

These works help demonstrate the feasibility of using parse trees and other modern NLP techniques to identify viewpoints and extract more structured claims from text. Being a step forward from pure bag-of-words analysis, they still leave out a considerable amount of information contained in natural language texts, namely by relying on topic detection, or on pre-defined categories, or on working purely on source-subject-predicate clauses. We propose to use a more sophisticated language model, where all entities participating in a statement are identified, where entities can be described as combinations of other entities, and where statements can be entities themselves, allowing for claims about claims, or even claims about claims about claims. The formal backbone of this model consists of an extended type of hypergraph that is both recursive and directed, thus generalizing semantic graphs and inducing powerful representation capabilities.

In subsequent sections we will discuss how NLP parser outputs and annotations can be used to create formal data structures that are suitable for computer-supported appraisal of large socio-semantic corpora, and enable a viable middle ground between more naive text mining methods and human-level understanding of natural language.

## 2 Semantic hypergraphs – structure and syntax

### 2.1 Structure

We assume that the richness of information provided by modern NLP parsers cannot be fully captured and analyzed using traditional graph-based network methods or distributional frameworks. On one hand, natural language is recursive, allowing for concepts constructed from other concepts as well as statements about statements, and on the other hand, it expresses  $n$ -ary relationships. To address this, we propose a model based on a hypergraph formalism. Combined with machine learning

parsing techniques, we contend that our model is capable of dealing a richer array of concepts than most existing approaches. It takes into account underlying conceptual relationships while remaining simple enough to lend itself to large-scale empirical analysis of human communication.

**Recursive ordered hypergraphs** While a graph  $G = (V, E)$  is based on a set of vertices  $V$  and a set of edges  $E \subset V \times V$  describing dyadic connections, a *hypergraph* [10] generalizes such structure by allowing  $n$ -ary connections. In other words, it can be defined as  $H = (V, E)$ , where  $V$  is again a set of vertices yet  $E$  is a set of hyperedges  $(e_i)_{i \in 1..M}$  connecting an arbitrary number of vertices. Formally,  $e_i = \{v_1, \dots, v_n\} \in E = \mathcal{P}(V)$ .

We further generalize hypergraphs in two ways: hyperedges may be ordered [15] and recursive [18]. Ordering entails that the position in which a vertex participates in the hyperedge is relevant (as is the case with directed graphs). Recursivity means that hyperedges can participate as vertices in other hyperedges. The corresponding hypergraph may be defined as  $H = (V, E)$  where  $E \subset \mathcal{E}_V$ , the recursive set of all possible hyperedges generated by  $V$ , i.e.

$$\mathcal{E}_V = \{(e_i)_{i \in \{1..n\}} \mid n \in \mathbb{N}, \forall i \in \{1..n\}, e_i \in V \cup \mathcal{E}_V\} \quad (1)$$

In this sense,  $V$  configures a set of irreducible hyperedges of size one, which we also denote as *atoms*, similarly to semantic graphs. From here on, we simply call these recursive ordered hyperedges as “hyperedges”, or just “edges”, and we denote the corresponding hypergraph as a “semantic hypergraph”. This type of structure has been proposed as a general solution for knowledge representation [9, 16].

Let us consider a simple example, based on a set  $V$  made of four atoms: the noun “(Berlin)”, the verb “(is)”, the adverb “(very)” and the adjective “(nice)”. They may act as building blocks for both hyperedges “(is Berlin nice)” and “(very nice)”. These structures can further be nested: the hyperedge “(is Berlin (very nice))” represents the sentence “Berlin is very nice”. It illustrates a basic form of recursivity.

## 2.2 Syntax

In a general sense, the hyperedge is the fundamental and unifying construct that carries information within the formalism we propose. We thus further introduce the notion of hyperedge *types*, which simply describe the type of meaning that some hyperedge carries. For instance, hyperedges can be used to represent concepts, predicates or relationships, as in the above examples — respectively (Berlin), (is) and (is Berlin nice).

**Connectors** In all, we distinguish 11 hyperedge types which appear to cover virtually all possible information

representation roles. We will provide extensive details on hyperedge types and their role in the next section. For now, it is enough to know that predicates, in particular and for instance, belong to a larger family of types that are crucial for the construction of hyperedges and that we call *connectors*. In this regard, semantic hypergraphs rely on a syntactic rule that is both simple and universal: the first element in a hyperedge that is not an atom must be a *connector*. In effect, a hyperedge represents information by combining other (inner) hyperedges that represent information. The purpose of the connector is to specify *in which sense* inner hyperedges are connected. Naturally, it can be followed by one or more hyperedges, possibly in a recursive fashion. These subsequent hyperedges always play the role of arguments with respect to the connector. Subsequent arguments are never connectors *as such*, even though they are hyperedges: as such, if they are not atoms, they must also start with a connector themselves.

Let us illustrate this on the simple hyperedge (is Berlin (very nice)). Here, (is) is a predicate that plays the role of connector, while (Berlin) and (very nice) are arguments of the initial hyperedge. (Berlin) is an atomic hyperedge, while (very nice) is itself a hyperedge made of two elements: the connector, (very), an atomic hyperedge, and an argument, (nice), also an atomic hyperedge. Both cannot be decomposed further.

Readers who are familiar with Lisp will likely have noticed that hyperedges are isomorphic to *S-expressions* [21]. This is not purely accidental. Lisp is very close to  $\lambda$ -calculus, a formal and minimalist model of computation that is based on function abstraction and application. The first item of an S-expression specifies a function, the following ones its arguments. One can think of a function as an association between objects. Although hyperedges do not specify computations, connectors are similar to functions at a very abstract level, in that they define associations. The concepts of “race to space” and “race in space” are both associated to the concepts “race” and “space”, but the combination of these two concepts yields different meaning by application of either the connector “in” or “to”. For this reason,  $\lambda$ -calculus has also been applied to dependency parse trees in the realm of question-answering systems [31].

We provide further concrete examples in table 1. This figure and the remaining discussion in this section are meant to illustrate what is gained by using hypergraphs, and specifically in relation to popular triplet-based knowledge representation systems. For example, the *Semantic Web* [6, 35] community tends to use standards such as *RDFa* [1], which represent knowledge as *subject-predicate-object* expressions, and are conceptually equivalent to semantic graphs [4, 38] (similarly, a particular type of hypergraph has been used in [11] to represent tagged resources by users, yet this also reduces to fixed triplet conceptualization). We can see in this ta-

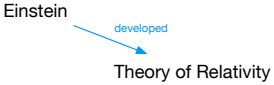
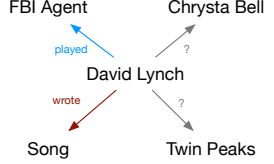
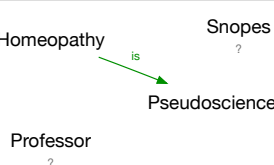
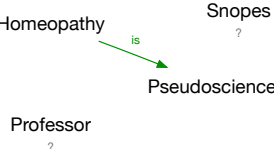
Original sentence	Hypergraphic representation	Graphic representation
"Einstein developed the Theory of Relativity"	<b>( developed Einstein ( of Theory Relativity ) )</b>	
"David Lynch wrote a song with Chrysta Bell"	<b>( wrote David Lynch ( with Song Chrysta Bell ) )</b>	
"David Lynch played an FBI agent in Twin Peaks"	<b>( played David Lynch ( in FBI Agent Twin Peaks ) )</b>	
"Professor claims homeopathy is pseudoscience" (in Snopes)	<b>( published Snopes ( claims Professor ( is Homeopathy Pseudoscience ) ) )</b>	

Table 1: Examples of simple hypergraphs extracted from sentences, and the shortcomings of corresponding semantic graph representations. a) Binary relationship including concept derived from other concepts; b) Intersecting non-binary relationships; c) Claims about claims (two levels); Predicates are represented as the first item of hyperedges or sub-hyperedges, and shown in bold, in both hypergraphs and source sentences.

ble how binary relationships and lack of recursion limit the expressive power of semantic graphs, and how hypergraphs can represent semantic information that is lost in the graphic representation. Concretely, in the graphic representation the concept of "theory of relativity" becomes atomic and is not related to sub-concepts. "David Lynch" is connected to "Chrysta Bell", but the information that this connection is a triplet that also includes the concept of "Song" is not preserved. It is possible to express a proposition connecting "homeopathy" and "pseudoscience", but it is not possible to express a proposition about a proposition.

## 2.3 Types

We now describe a type system that further clarifies the role that each entity plays in a hypergraphic expression. We start by presenting the types that atoms may have and by discussing their use in constructing higher-order entities. We then show how the type of a hyperedge may be recursively inferable from the types of the connector and the subsequent arguments.

In developing this knowledge representation model, we were guided by the *Stanford Universal Dependencies* [14]. We chose this methodology for two reasons: firstly because the Universal Dependencies are meant to be general enough to represent grammatical relationships in all of human languages. Although we only use English sources in this work, we aim to ensure that semantic hypergraphs are sufficiently expressive to represent the full range of concepts and statements that can be communicated in human languages. In B we illus-

trate how hyperedges can be used to represent the various constructs present in the Universal Dependencies framework. The second reason is that, as we will discuss in the next section, we rely on dependency parsers as one of the main components in the process of translating texts in natural language to semantic hypergraphs.

Furthermore, semantic hypergraphs are an attempt to move from grammatical representations to a purely conceptual representation that will lend itself to automating reasoning, e.g.: inference and ontology construction. Later in this text, we will make this more explicit by discussing the practical application of semantic hypergraphs to the analysis of a corpora of news headlines.

**Atoms and concepts** The first, simplest and most fundamental type that atoms can have is that of a *concept*. This corresponds to concepts that can be expressed as a single word in the target language, for example "apple". As could be guessed from the previous subsection, atoms are labeled by this human-readable string of characters stemming from the target language. The nomenclature we propose further indicates the type of an atom by appending a more machine-oriented code after this label, after a slash (/). In the case of a concept, this code is "c":

(apple/c)

As we shall see, these machine-oriented codes remove ambiguity, and facilitate automatic inference as well as other computational methods. Here we will present a simplified version of these machine-oriented codes, focusing only on the eleven main type indicators, as seen



Code	Type	Purpose	Example	Atom	Non-atom
c	concept	Define atomic concepts	<u>apple/c</u>	×	×
p	predicate	Build relations	( <u>is/p</u> berlin/c nice/c)	×	×
a	auxiliary	Modify a predicate	( <u>not/a</u> is/p)	×	
m	modifier	Modify a concept	( <u>red/m</u> shoes/c)	×	×
b	builder	Build concepts from concepts	( <u>of/b</u> capital/c germany/c)	×	
w	meta-modifier	Modify a modifier	( <u>very/w</u> large/m)	×	
x	subpredicate	Auxiliary predicate	( <u>by/x</u> john/c)	×	
t	trigger	Build specifications	( <u>in/t</u> 1994/c)	×	
r	relation	Express facts, statements, questions, orders, ...	( <u>is/p</u> berlin/c nice/c)		×
d	dependent	Relation argument	( <u>by/x</u> scientists/c)		×
s	specifier	Relation specification (e.g. condition, time, ...)	( <u>in/t</u> 1976/c)		×

Table 2: Hyperedge types with use purposes and examples. Connector types are emphasized with a gray background. The rightmost columns specify whether this type may be encountered in atomic or non-atomic hyperedges.

in table 2. A complete description of machine-oriented codes can be found in A.

**Connectors** The second and last role that atoms can play is the role of connector. We then have seven types of connectors, each one with a specific function that relates to the construction of specific types of hyperedges.

The most straightforward connector is the *predicate*, whose code is “p”. It is used to define relations, which are frequently statements (we will see that there are other types of relations, for example questions). Let us revisit a previous example with types:

(is/p berlin/c nice/c)

The predicate (is/p) both establishes that this hyperedge is a relation between the entities that follow it, and gives meaning to the relation.

This case is isomorphic to the typical knowledge graph, where “Berlin” and “nice” would be connected by an edge labeled with “is”. But what if we want to declare the opposite, that Berlin is not nice? Then we can modify the predicate itself with an *auxiliary*, whose code is “a”, such as (not/a), so that:

((not/a is/p) berlin/c nice/c)

There are two atomic connector types that allow for the construction of new concepts from existing ones: *modifiers* (“m”) and *builders* (“b”). Modifiers operate on a single concept. A typical case is adjectivation, for example:

(nice/m shoes/c)

Note here that “nice” is being considered as a modifier, while “nice” was a concept in the previous case: this is due to the fact that (nice/m) and (nice/c) refer to two distinct atoms which share the same human-readable label, “nice”.

Builders combine several concepts to create a new one. For example, the atomic concepts of (capital/c) and (germany/c) can be combined with the builder atom (of/b) to produce the concept of “capital of Germany”:

(of/b capital/c germany/c)

A very common structure in English and many other languages is that of the compound noun (e.g. “guitar player” or “Barack Obama”). To represent these cases, we introduce a special builder atom that we will call (+/b). Unlike what we have seen so far, this is an atom that does not correspond to any word, but indicates that a concept is formed by the compound of its arguments — it is necessary to render such compound structures. Using this builder, the previous examples can be represented respectively as (+/b guitar/c player/c) and (+/b barack/c obama/c).

*Meta-modifiers* (code “w”) allow for the modification of a modifier, in a similar way to how modifiers can make a concept more specific. For example, the modifier (large/m) can be applied to the concept (windows/c) to create the concept (large/m windows/c). But what if we want to specify “very large windows”? In this context, “very” acts as the meta-modifier of the modifier “large”:

((very/w large/m) windows/c)

Meta-modifiers can be infinitely nested, so there is no need for meta-meta-modifiers.

The two remaining cases, *subpredicates* (x) and *triggers* (t), deal with constructs that depend on relations and help to give them meaning. Subpredicates are language constructs that are a logical part of the predicate, but that refer specifically to one of the arguments of the relation. One typical case in the English language is that of a preposition that indicates the agent in a passive sentence, for example “The species was discovered by scientists”:

((was/a discovered/p) (the/m species/c)  
(by/x scientists/c))

Triggers indicate some additional specification of a relationship, for example conditional (“We go if it rains.”), or temporal (“John and Mary traveled to the North Pole in 2015”), or local (“Pablo opened a bar in Spain”), etc. To illustrate with the latter example:

(opened/p pablo/c (a/m bar/c) (in/t  
spain/c))

**Hyperedge type inference** We have seen that atoms have explicit types, and we already hinted at the fact that hyperedges also have types, but these are implicit and inferable from the type of the connector. Given, for example, that (germany/c) is an atom of type concept (c), the hyperedge (of/b capital/c germany/c) is also a concept, and this can be inferred from the fact that its connector is of type builder (b). The same happens when the connector is a modifier (m): the hyperedge (northern/m germany/c) is also a concept (c). When the connector is a meta-modifier, it needs to be followed by exactly one modifier, and yields a modifier, as in: (very/w large/m). When the connector is an auxiliary, it needs to be followed by exactly one predicate, and yields a predicate, as in: (not/a is/p). Builders, by contrast, need to be followed by at least two concepts.

Table 3 lists these inference rules and their respective requirements. This composition table also happens to induce syntactic constraints on the eight types that we evoked so far. Atomic types are entirely covered by these eight types, of which five exclusively concern atoms (auxiliaries, builders, meta-modifiers, subpredicates and triggers).

We can see in this table that there are also three types that only concern non-atomic hyperedges, i.e. they are always defined as the result of a composition of hyperedges: relation (r), dependent (d) and specifier (s). Hyperedges of these types are defined with the help of connectors of the respective types: predicate (p), subpredicate (x) and trigger (t). *Relations* are typically based on a predicate and used to state some facts (even though they

Element types	→	Resulting type
(m c)		c
(b c c+)		c
(w m)		m
(a p)		p
(p [crds]+)		r
(x [cr]+)		d
(t [cr]+)		s

Table 3: Type inference rules. We adopt the notation of regular expressions: the symbol + is used to denote an arbitrary number of entities with the type that precedes it, while square brackets indicate several possibilities (for instance, [cr]+ means “at least one of any of both “c” or r types).

can also be used to represent questions, orders and other things). (is/p Berlin/c nice/c) is an obvious example of relation. In our context, they thus turn out to be a crucial hyperedge type.

*Dependents* and *specifiers* are types that play a more peripheral role, in the proper sense, in that they provide specifications to a concept or a relation. A dependent is a hyperedge argument of a relation starting with a subpredicate connector, as in (by/x scientists/c). A specification is what a trigger produces, for example, the trigger “in/t” can be used to construct the specification: (in/t 1976/c). Specifications, as the name implies, add precisions to relations e.g., when, where, why or in which case something happened.

### 3 Translating natural language into semantic hypergraphs

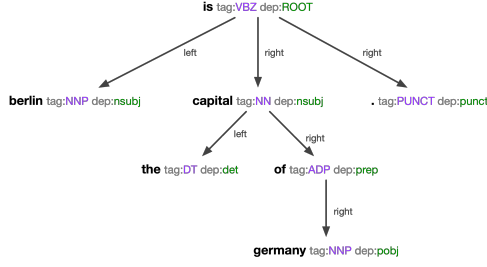
We now discuss the crucial task of translating natural language into this representation. The process starts with the application of conventional NLP methods: segmentation of text into sentences, followed by the creation of dependency parse trees and part-of-speech annotation of each token. For these tasks we used spaCy<sup>1</sup> — an open-source library for NLP in Python which includes convolutional neural network models for tagging, parsing and named entity recognition in multiple languages. A relatively recent comparison of ten popular syntactic parsers found spaCy to be the fastest, with an accuracy within 1% of the best one [13].

As can be seen in figure 1, the translation process consists of two stages. In the first one, sentence tokens (i.e. words annotated with POS tags and dependency labels) are transformed into atoms. The human-readable part of the atom is the word itself, so this process amounts

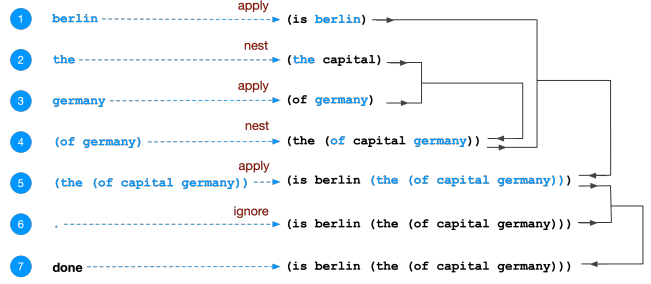
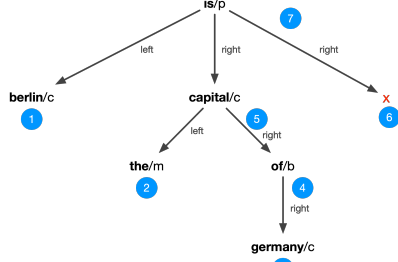
<sup>1</sup>More precisely, we used spaCy v2.0.13 with the language model en\_core\_web\_lg-2.0.0.

“Berlin is the capital of Germany.”

Stage 1: Generate atoms



Stage 2: Generate hyperedges



(is/p berlin/c (the/m (of/b capital/c germany/c)))

Figure 1: The two stages of translation of a sentence in natural language to a semantic hyperedge. In stage 1, conventional NLP tasks are performed (part-of-speech tagging and dependency parsing). POS tags and dependency labels are used to generate atoms from words. In stage 2, the structure of the dependency parse tree is traversed to generate the hyperedge. Numbered circles represent the steps of depth-first, left-to-right traversal. On the right, we show how the hyperedge is recursively built at each step, by combining the current hyperedge at the child with the current hyperedge at its parent.

to mapping POS tags and dependency labels to semantic hypergraph type annotations. In abstract, this task can be modeled as a function  $\alpha(\text{tags}, \text{dep})$ , mapping POS tags and dependency labels (dep) to atom types. To use an example from the figure, if  $\alpha(\text{NNP}, \text{nsbj}) = c$ , then the “berlin” token is translated to the (berlin/c) atom.

After the first step, we are left with atoms organized in a tree structure that represents the plain grammatical roles of the words to which these atoms correspond. We then build the hyperedge by reorganizing these words into recursive lists. By construction, the parse tree is ordered, i.e. child nodes are put in an order that respects the grammatical structure of the original sentence. At first sight, it could be trivial to turn the tree into a hyperedge by performing a depth-first traversal and appending recursively each subtree into increasingly nested hyperedge. For instance, the parse tree of “Berlin is the capital of Germany”, as shown in the figure, would produce “(is/p berlin/c (capital/c the/m (of/b germany/c)))”, which is not a syntactically valid

hyperedge. Instead, the second stage of translation consists precisely in using this grammatical structure, as well as the previously obtained atom types, to infer the hyperedge structure that correctly conveys the same meaning as the original sentence.

As illustrated in the bottom half of figure 1, the hyperedge generation stage consists of traversing the dependency tree depth-first, and recursively building an hyperedge by selecting the appropriate way to merge the parent and child hypergraphical entities at each tree node. Once again, this mechanism can be abstracted as a mapping function  $\beta(e_p, c_p, e_c, c_c)$ , making a correspondence between the parent ( $e_p$ ) and the parent connector types ( $c_p$ ), as well as the child ( $e_c$ ) and the child connector types ( $c_c$ ), to the operation ( $o$ ) that should be employed to combine the two entities. In practice, these fall into three main categories: *apply*, *nest* and *ignore*. It is not necessary to describe the small technical distinctions within these categories to explain the general method, so it is enough to consider the general

ones.<sup>2</sup> To explain what we mean by *apply*, *nest* and *ignore*, it is useful to recall the analogy of connectors as functions (or, consequently, hyperedges as functions applied to lists of arguments). When considering any given parent-child pair in the dependency tree, there are three possible cases: child is an argument to be applied to the connector/hyperedge defined by parent (*apply*); parent is an argument to be applied to the connector/hyperedge defined by child (*nest*); or finally, child should be discarded (*ignore*). Examples for each one of these cases can be found in the “Stage 2” example of figure 1. In step 1, the child atom (`berlin/c`) is applied to the parent atom (`is/p`). Conversely, in step 2, the child atom (`the/m`) is a connector that is nested around the parent atom (`capital/c`). In step 6, the punctuation mark is ignored.

It can now be seen that the translator is ultimately defined by specifying the two functions  $\alpha$  and  $\beta$ . In this work, these functions were implemented by hand, as simple decision trees. This was done by trial-and-error and using simple intuitions. The specific details of this implementation are not particularly interesting, except for readers motivated to dive deep into the details. As we will see in subsequent sections, this translator is of sufficient quality to allow us to infer non-trivial knowledge from text in natural language. It seems very likely that higher-quality translators can be created using supervised machine learning approaches. Both functions can be trivially treated as classification problems with categorical features, amenable to approaches such as random forests or simple feed-forward neural networks. Such improvements are obviously desirable, but outside the scope of the present work, which is meant to present the foundations of the semantic hypergraph approach, while avoiding any further complexities.

Naturally, it could be possible to produce our hypergraphic representation directly from the sentences, using some supervised machine learning approach, without using the dependency parse POS-tags as intermediary steps. There are however good practical reasons to avoid this option, the main one being that large training sets would need to be generated. By relying on traditional NLP methods to perform the “heavy lifting”, we take advantage of the extensive work and training sets already available for these approaches, and of the likely future improvements in precision to be expected, given their popularity.

Even though we have only so far created a translator from English, we expect that it will be only necessary to define language-specific  $\alpha$  mappings to support other languages. Stage 2 works with atom types and parse tree structures. The former are language-independent, the latter might be tractable by a sufficiently generic  $\beta$  func-

tion.

We have already hinted that some complexities were left out of this explanation. This includes subtypes and argument codes, which are detailed in A. The 11 basic types presented in table 2 are the building-blocks of meaning in semantic hypergraphs, but for several tasks it is useful to provide further specifications, i.e. subtypes. As a simple example, a concept can refer to a proper name (e.g., (`berlin/c`)), or to a common concept (e.g., (`capital/c`)). We will introduce a few more details like this where necessary. Overall, the 11 basic types provide the fundamental structure and are meant to be complete. We empirically demonstrate this completeness by showing that they are sufficient to cover all the cases in the Stanford Universal Dependencies (B). The subtypes and additional type information are meant to be extensible where necessary, for future applications. In A we present the complete taxonomy of hyperedge types, subtypes and other specifications used in this work.

## 4 Concepts, ontologies and coreference resolution

We are now in the position to start discussing how this hypergraphic representation makes it possible to infer knowledge using simple symbolic and probabilistic rules. More specifically, we will show how to derive ontologies and perform coreference resolution among concepts. For example, how automated methods can reach the conclusion that “President Obama” is a type of “President”, or that “Obama”, “Barack Obama” and “President Obama” refer to the same external entity, while “Michelle Obama” refers to another one. To proceed, it is necessary to disclose a few more details about concept representation.

### 4.1 More about concepts and implicit taxonomies

Given a concept hyperedge, a key issue is that of inferring its *main concept*, i.e. the concept that can be assumed to be its hypernym. Beyond the simple case of atoms, concept hyperedges may only be formed by connectors that are either modifiers or builders. When the connector is a modifier, finding the hypernym is admittedly trivial.

When the connector is a builder, it is still often possible to infer the main concept among the arguments. With compound nouns ((`+/b`) builder), the translator (and more specifically, the  $\alpha$  function discussed in section 3) can make use of part-of-speech and dependency labels to infer the main concept. Another common situation where finding is quite trivial is the case of relational builders, such as (`of/b`), which express a relationship between the arguments. For example, in (`of/b capital/c germany/c`), the main concept is (`capital/c`). “Capital of Germany” is thus a type of capital. In English and many

<sup>2</sup>The interested reader can inspect the source code at <https://github.com/graphbrain/graphbrain>, and more specifically the English-to-Hypergraph parser module `graphbrain/parsers/parser_en.pyx`



other languages, it is always the case that the first argument after a relational builder is the main concept.

Similarly, hyponyms of a concept can be found by looking for hyperedges where the concept appears either as the main argument of a builder-defined concept or as the argument of a modifier-defined concept. It follows from these structures that the semantic hypergraph representation implicitly builds a taxonomy. More generally, we can talk of an implicit ontology. Beyond the taxonomical relationships that we described, the concepts that form a concept hyperedge are related to it in a non-specified fashion. For example, we know that (germany/c) is related in an unspecified way to (of/b capital/c germany/c). Of course, this is not to say that a more specific relation cannot be inferred by further processing with other methods. Here we are simply highlighting the ontological relations that come “for free” with the hypergraphic representation.

## 4.2 Coreference resolution: co-occurrence graph

A common but challenging task in NLP is that of coreference resolution, i.e. identifying different sequences of n-grams that refer to the same entity (such as “Barack Obama” and “President Obama”). This is an old research topic [37] that has been revived lately with modern machine learning methods [29]. ML approaches such as deep learning require large training sets and tend to provide black box models, where precision/recall can be measured and improved upon, but the exact mechanisms by which the models operate remain opaque. Here we do not mean to provide a complete solution to this problem, but instead show that several cases of coreference resolution can be performed in a simpler and understandable manner through the use of semantic hypergraphs for situations that are nevertheless common and useful, especially in the context of social science research.

We will discuss in the following section several experimental results that we obtained on a dataset of several years of news headlines. This corpus is largely focused on political issues, and it is dominated by reports of actors of various types making claims or interacting with each other. These actors can be people, institutions, countries and so on. In our hypergraphic representation, such actors will very frequently be referred to by hyperedges forming compound nouns, with the use of the +/b connector, as discussed previously.

In figure 2 we can see such a case: a number of compound concept edges with the main atomic concept obama/c refer to actors. How can we group them in sets, such that all the cases in a given set refer to the same entity? Here, we start taking advantage of the hypergraph as a type of network, and of the analysis graphs that we can easily distill from the hypergraph. Seman-

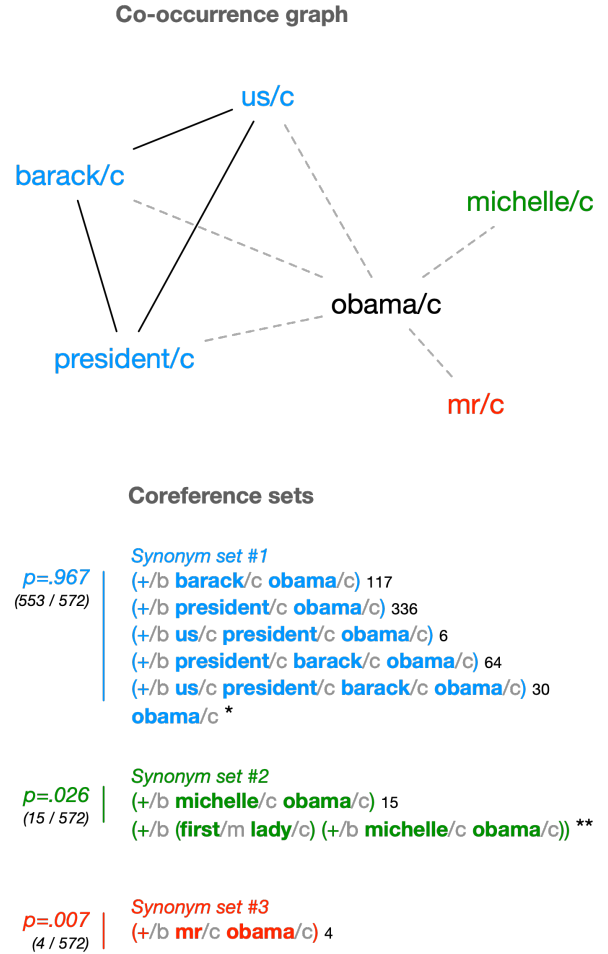


Figure 2: Example of coreference resolution. On top we can see the co-occurrence graph and its components, identified by different colors and leading to corresponding coreference sets on the bottom. The probabilities for each coreference set are shown to their left, including the ratios of total degree of the set to total degree, used to compute them. Individual degrees are shown next to each edge. \* indicates the assignment of the seed to one of the coreference sets. \*\* indicates the recursive nature of the process, with (+/b michelle/c obama/c) taking the role of seed in another instance of this coreference resolution process.

tic graph-based disambiguation has been extensively explored since the mid-2000s, especially emphasizing the importance of centrality and proximity in deciding which sense correspond to a given word in a certain context, and semantic hypergraphs are no exception [22, 27, 2].

We can trivially traverse all the concepts in the hypergraph, finding the subset of concepts that play the role of main concept in the above mentioned compound concept constructs. For each of these *seed concepts*, we can then attempt to find coreference relationships between

the concepts they help build. In the figure, we see an example using the seed concept (obama/c). On the right side of the figure, we see all the compound concepts containing the seed as the main concept (except for the ones marked with \* and \*\*). It is possible then to build a graph of all the auxiliary concepts that appear together with the seed. A connection between two concepts in this graph means that there is at least a compound concept in which they take part together. In the example, we can see that this graph has three maximal cliques, which we identified with different colors. We then apply this simple rule: two compound concepts are assumed to refer to the same entity if all of their auxiliary concepts belong to the same maximal clique. The intuition is that, if auxiliary concepts belong to a maximal clique, then they tend to be used interchangeably along with the seed, which indicates that they are very likely to refer to the same entity. We will show that this intuition is empirically confirmed in our corpus, from where the example in the figure was extracted.

The co-occurrence graph method produces the coreference sets seen on the right of the figure, except for the items marked with \* and \*\*. As can be seen, it correctly groups several variations of hyperedges that refer to Barack Obama (president of the United States during most of the time period covered by our news corpus), and it correctly identifies a separate set referring to Michelle Obama, his wife. It can also be seen that it fails to identify that “Mr. Obama” is also likely to refer to Barack Obama. We will say more about this specific case when we discuss claim analysis, in the next section.

But what about the seed concept itself, in this case (obama/c)? The co-occurrence method is not able to assign it to one of the sets. Here we employ another simple method, this time of a more probabilistic nature. Before tackling this method, we have to make a small detour to discuss the semantic hypergraph from a network analysis perspective.

### 4.3 Simple hypergraph metrics

In a conventional graph, it is common to talk of the degree of a vertex. This refers simply to the number of edges that include this vertex or, in other words, the number of other vertices that it is directly connected to (we assume here an undirected graph without self-loops). With the semantic hypergraph, such measure is not so straightforward, given that an edge can have more than two participants, and that recursivity is permitted.

Let us first define the set  $D_e$ , containing all the edges in with a given edge  $e$  participates:

$$D_e = \{e_i | e_i \in E \wedge e \in e_i\} \quad (2)$$

We define the degree of a hyperedge  $e$  as:

$$d(e) = \sum_{e_i \in D_e} (|e_i| - 1) \quad (3)$$

This is to say, the hypergraphic degree is the number of edges with which a given edge is connected to by outer hyperedges. It is intuitively equivalent to the conventional graph degree.

Another useful metric that we can define is the *deep degree*, which considers edges connected by hyperedges not necessarily at the same level, but appearing recursively at any level of the connecting hyperedge. Let us consider the set  $\Delta_e$ , containing the edges that co-participate in other edges with  $e$  at any level. This set is recursively defined, so we describe how to generate it, in Algorithm 1.

```

Function Generate $\Delta(e)$ 
  Data: An edge  $e$ 
  Result:  $\Delta_e$  neighborhood of edge  $e$ 
   $\Delta_e \leftarrow D_e$ 
  for  $e' \in D_e$  do
     $\Delta' \leftarrow \text{Generate}\Delta(e')$ 
     $\Delta_e \leftarrow \Delta_e \cup \Delta'$ 
  end for
  return  $\Delta_e$ 
end

```

**Algorithm 1:** Generating the neighborhood  $\Delta_e$  of an edge  $e$ .

We can now define the deep degree  $\delta$  as:

$$\delta(e) = \sum_{e_i \in \Delta_e} (|e_i| - 1) \quad (4)$$

To provide a more intuitive understanding of these metrics, let us consider the following edge:

(is/p berlin/c (of/b capital/c  
germany/c))

Let us also assume that no other edges exist in the hypergraph. In this case, the edges (is/p), (of/b capital/c germany/c) and (germany/c) all have degree  $d = 1$ , because they all participate exactly in one edge. The first two ((is/p) and (of/b capital/c germany/c)) also have deep degree  $\delta = 1$ , but the latter (germany/c) has deep degree  $\delta = 2$ , because not only does it participate directly in the edge (of/b capital/c germany/c), but it also participates at a deeper level in the outer edge (is/p berlin/c (of/b capital/c germany/c)). In other words, the higher deep degree of (germany/c) indicates that it plays an increased role as a *building block* in other edges.

### 4.4 Coreference resolution: probabilistic seed assignment

Back to figure 2, each coreference set is labeled with a probability  $p$ , representing the chance that a given seed appears in one of its edges, if we were to uniformly enumerate all edges that rely on this seed. This configures a

simple estimation of the probability of the seed by itself being used with a certain meaning, represented by the given coreference set. These probabilities are thus the ratio between the sum of the degrees of the edges in each coreference set and the total degree of all edges that include the seed, i.e. of all coreference sets.

Two simple heuristics drive this step. One is that people will tend to use an ambiguous abbreviation of a concept when the popularity of one of the interpretations is sufficiently high in relation to all the others. For example, both  $(+/b \text{ barack}/c \text{ obama}/c)$  and  $(+/b \text{ michelle}/c \text{ obama}/c)$  share the seed  $(\text{obama}/c)$ , but when referring only to  $(\text{obama}/c)$  during the period he was a US president, people tend to assume that it refers to the most frequently mentioned entity – *Barack Obama*. The other is that a given seed should only be considered as an abbreviation if it is used a sufficient amount of times as a primary concept in relations, i.e. if there is evidence that it is in fact used on its own to refer directly to some external concept, and not only as a common component of primary concepts. Put differently, seeds referring to common concepts which act often as building blocks of other concepts (i.e., higher deep degree with respect to degree) are less likely to be valid abbreviations. Such is the case for “house” (which may indifferently refer to the White House or Dr. House) and “qaida” (which is typically used as a building block for Al Qaida and never by itself).

We thus establish a criterion that consists of the fulfillment of each of these two conditions, corresponding respectively to the heuristics above. A given seed  $s$  is assigned to the coreference set  $C$  with the highest  $p$  if:

- $p$  is above a certain threshold  $\theta$
- $d_s/\delta_s$  is above a certain threshold  $\theta'$

We set the threshold to the values  $\theta = .7$  and  $\theta' = .05$ , that we verified empirically to produce good results. Naturally, these thresholds can be fine-tuned using methods more akin to hyper-parameter optimization in ML, but such optimizations are outside of the scope of this work. When the criterion is not met, the seed is left as a reference to a distinct entity. In our corpus, this happens for example with “Korea”, which remains an ambiguous reference to either “North Korea” or “South Korea”.

#### 4.5 Synonyms, disambiguation, anaphora resolution and other cases

We do not present here a general solution for coreference resolution and synonym detection. Some further cases will also be covered in the next section, notably anaphora resolution, given that this requires the discussion of predicates and relations in more detail, and along with empirical results. Other cases are left out of this work, but we would like to provide a quick insight into how they may be treated.

One obvious example is that of synonyms, which are not implied by a pure structural analysis of hyperedges – e.g. *red* and *crimson*, as well as *U.S.* and *United States*, for they share no common seed (as opposed to the cases emphasized in the previous subsection). This type of synonym detection may be achieved with the help of a general-knowledge ontology such as Wordnet or DBpedia, and/or with the help of word embeddings such as word2vec. This is a foreseeable and desirable improvement to hypergraph-based text analysis that we leave for future work.

Another case is the inverse problem of synonym detection: disambiguating atoms that correspond to the same word but to different entities, for example distinguishing “Cambridge (UK)” from “Cambridge (USA/Massachusetts)”. We do not perform this type of distinction in this work, but we present another syntactic detail that enables them from a knowledge representation perspective: the atom namespace. Quite simply, beyond the human-readable part and the type and other machine-oriented codes, a third optional slash-separated part can be added to atoms, allowing to distinguish them in cases such as the above, e.g.: *cambridge/c/1* and *cambridge/c/2*.

Finally, coreference resolution can also apply to cases where neither seed concepts are shared, nor anaphoras are present. Let’s say that one sentence refers to “Kubrick” and the next one to “the film director”. Both this type of case and the above mentioned disambiguation cases are likely to be more easily solved with the help of structured knowledge surrounding the concepts in the semantic hypergraph, eventually including general knowledge as mentioned. For example, it could be detected that a certain reference to “Cambridge” is closer to references related to the United States, or that “Kubrick” is structurally close to the concept of “film director”.

## 5 Experimental Results: Claim and Conflict Analysis

We arrive at the point where we can present some experimental results. Here we aim at demonstrating the application of the formalisms and methods discussed so far to the analysis of a large corpus of real text. More specifically, we worked with a corpus of news titles that were shared on the social news aggregator *Reddit*. We extracted all titles shared between January 1st, 2013 and August 1st, 2017 on *r/worldnews*, a community that is described as: “*A place for major news from around the world, excluding US-internal news.*” This resulted in a corpus of 404,043 news titles. We applied the methods described in sections 3 and 4 to generate a hypergraph from the titles.

We decided to focus on two specific categories of utterances that are very frequent in news sources, and of spe-



Figure 3: Two examples of relations starting with either a claim or a conflict predicate.

cial interest for the social sciences, especially the study of public spaces [34, 41]: a *claim* made by an actor about some topic and an expression of *conflict* of one actor against another, over some topic. Helpfully, the detection of such categories also allows us to illustrate simple symbolic inference over the hypergraph.

## 5.1 Knowledge Inference

We recall that semantic hypergraphs are aimed at representing everything that can be expressed in natural language, but with increased explicit structure and rigor (unambiguity). Their purpose is to organize knowledge in such a way that facilitates inference. Such is the case with the example we present here of detecting certain categories of utterances and their inner structure. In figure 3 we present two real sentences from our corpus and their respective hyperedges. Example (a) was classified as a claim and example (b) as an expression of conflict.

*Predicates and roles.* Predicates can induce specific roles that the following arguments play in a relation. Let us consider here a few very common roles: *subject*, *object*, *complement*, *relative relation* and *specification*. The first three correspond to the typical grammatical roles of subject, direct object and subject complement. A *relative relation* is a nested relation, that acts as a building block of the outer relation that contains it. In the case of example (a), it represents what is being claimed by the subject. *Specifications* were already discussed in section 2, and their purpose as hyperedges coincides with their role when participating in relations: as an additional specification to the relation (temporal, conditional, etc.). The need for argument roles in relations arises from cases where the role cannot be inferred from the type of the argument. For example, the same concept could participate in a relation as a subject or as an object. In practice

and in detail, our platform is actually able to manage and encode roles in predicates. We leave such encodings out of our examples for the sake of readability, but they can be referred to in A, including the full list of possible argument roles in relations. All this structure and argument role management finally allow us to define very simple rules that both capture more abstract categories of utterances such as claims and expressions of conflict, as well as identify their main and relevant components.

*Claims.* We consider a relation hyperedge to be a claim if the two following conditions are met:

- The lemma of the main atom of the predicate edge belongs to the set {*say*, *claim*}.
- The relation contains one argument playing the role of subject and another the role of relative relation.

As we have seen in section 2, a predicate can be a non-atomic hyperedge. The meaning of predicate atoms can be extended or modified with the help of an edge of type *auxiliary*. For example, the English verb conjugation “*was saying*” is represented as (was/a saying/p). Eventually, there is always a predicate atom that corresponds to the main verb in the predicate. When translating the corpus of news items to hyperedges, we also stored auxiliary hyperedges connecting every atom that corresponds to a word to the lemma of that word. For example:

(lemma/p saying/p say/p)

*Conflicts.* We define two similarly simple conditions for expressions of conflict, with a third optional one to assign a topic to the conflict:

- The lemma of the main atom of the predicate edge belongs to the set {*warn*, *kill*, *accuse*, *condemn*, *slam*, *arrest*, *clash*}.
- The relation contains one argument playing the role of subject and another the role of object.
- Optionally, if the relation contains a specification defined with a trigger in the set {*of/t*, *over/t*, *against/t*, *for/t*}, then the content of this specification is deemed to be the topic of the conflict.

The English language allows for vast numbers of verb constructions that indicate claims or expressions of conflict. Instead of attempting to identify all of them, we considered the 100 most common predicate lemmas in the hypergraph, and from there we identified the above sets of “claim predicates” and “conflict predicates”. Overall, we found 3730 different predicate lemmas, and their rank-frequency distribution is expectably heavy-tailed. In this case, this small fraction of the set accounts for 60.6% of the hyperedges. Naturally, coverage could be



improved by considering more predicates, but with diminishing returns.

*Subjects and actors.* The examples in figure 3 were purposely chosen to be simple, but the above rules can match more complicated cases. For example, the following sentence was correctly identified and parsed as a claim:

U.S. Secretary of State John Kerry was the intended target of rocket strikes in Afghanistan's capital Saturday, the Taliban said in a statement claiming responsibility for the attacks.

Both claim and conflict structures imply that the edge playing the role of subject in the relation is an actor. Using the methods described in section 4, we can identify the coreference set of each actor and replace all occurrences of this actor with the same edge. For each coreference set we choose the edge with the highest degree as the main identifier, following the heuristic that the most commonly used designation of an entity should be both recognizable and sufficiently compact.

As seen in figure 3(a), the inner subject (i.e., the subject of the relative relation that represents what is being claimed) can be a pronoun. These cases are very common, and almost always correspond to a case where the actor is referencing itself in the content of the claim. On one hand, we perform simple anaphora resolution: if the inner subject is a pronoun in the set {*he/c*, *it/c*, *she/c*, *they/c*}, then we replace it with the outer subject. On the other hand, we take advantage of the pronoun to infer more things about the actor. The four pronouns mentioned indicate, respectively, that the actor is a male human, a non-human entity, a female human, or a group. We take the majority case, when available, to assign one of these categories to actors. The pronoun *they* is being increasingly used as a gender-neutral third person singular case, but we haven't found such cases in our corpus.

Table 4 shows the top five actors per category, ranked by their degree in the hypergraph.

Obviously, more sophisticated rules can be devised, both for anaphora resolution and category classification. Our goal here is to illustrate that, thanks to the semantic hypergraph abstraction, it becomes possible to perform powerful inferences (i.e., both useful and at a high level of semantic abstraction) with very simple rules.

## 5.2 Topic Structure

The very definition of *topic*, for the purpose of automatic text analysis, is somewhat contingent on the method being employed.

One of the most popular topic detection methods in use nowadays is *Latent Dirichlet Allocation (LDA)* [8], which is a probabilistic topic model [7] that views topics as latent entities that explain similarities between sets

of documents. In LDA, topics are abstract constructs. Documents are seen as a random mixture of topics, and topics are characterized by their probability of generating each of the words found in the document set. LDA uses a generative process to statistically infer these probabilities. Ultimately, a topic is described by the set of words with the highest probabilities of being generated by it. Human observers can then infer some higher-level concept from these words and probabilities. For example, if the five highest probability words for a topic *X* are {*EU*, *Junkers*, *May*, *Barnier*, *Trade*}, a human observer may guess that a good label for this topic is *Brexit Negotiations*. LDA is applicable to sets of documents, for a predefined number of topics, where each document is considered to be a *bag-of-words*.

A different approach to topic detection is *TextRank* [23], which is capable of detecting topics within a single document. With TextRank, the document is first transformed into a word co-occurrence graph. Common NLP approaches are used to filter out certain classes of words from the graph (e.g., do not consider articles such as “the”). Topics are considered to be the words with the highest network centrality in this graph, according to some predefined threshold condition. Simple statistical methods over the co-occurrence graph can be used to derive *ngram* topics from the previous step. Given that the order in which words appear in the document is important, TextRank cannot be said to be a *bag-of-words* approach such as LDA. It relies a bit more on the meaning of the text, and it is more local – in the sense that it works inside a single document instead of requiring statistical analysis over a corpus of documents.

In this work, we move significantly more in the direction of text understanding and locality. Our topics are firstly inferred from the meaning of sentences. As we have shown, pattern analysis of hyperedges can be used to infer relationships such as *claim* and *conflict*, which imply both actors and topics. Given coreference detection, such topics are characterized by sets of hyperedges, but these sets are not probabilistic in the sense that LDA's are. Instead, they are a best guess of symbolic representations that map to some unique concept. Our approach relies even more on meaning than TextRank, and it allows for topic detection at an even more local scale: single sentences.

In the examples given in figure 3, the claim shown in (a) implies the rather specific topic “afraid of military us strike”, and (b) the topic “military engagement in syria”.

Another important aspect of our approach is that topics can be composed of other topics or concepts, forming a hierarchical structure. This is a natural consequence of how we model language, as explained in section 4.1. This allows us to explore topics at different levels of detail. The topic implied by a claim or conflict can be very specific and possibly unique in the dataset, but the more general subtopics or concepts that it contains can be used to

Type	Rank	Actor	Edges (coreference set)	Degree
<i>non-human</i>	1	U.S.	us/c, (the/m us/c)	9314
	2	China	china/c	8366
	3	Russia	russia/c	8197
	4	Israel	israel/c	4767
	5	Iran	iran/c, (the/m iran/c)	3863
<i>male</i>	7	Putin	putin/c, (+/b president/c vladimir/c v/c putin/c), (+/b president/c vladimir/c putin/c), (+/b vladimir/c putin/c), (+/b president/c putin/c), (russian/m (+/b president/c putin/c)), (the/m (russian/m (+/b president/c putin/c)))	3364
	10	Obama	(+/b president/c obama/c), (+/b us/c president/c barack/c obama/c), obama/c, (+/b us/c president/c barack/c obama/c), (+/b us/c president/c obama/c), (+/b barack/c obama/c), (+/b president/c barack/c obama/c)	2860
	20	Trump	(+/b donald/c trump/c), (+/b us/c president/c donald/c trump/c), (+/b president/c trump/c), (+/b us/c president/c elect/c donald/c trump/c), (+/b president/c donald/c trump/c), trump/c	1709
	25	Netanyahu	(+/b prime/c minister/c netanyahu/c), netanyahu/c, (+/b benjamin/c netanyahu/c), (+/b prime/c minister/c benjamin/c netanyahu/c)	1208
	26	David Cameron	(+/b prime/c minister/c david/c cameron/c), (+/b uk/c prime/c minister/c david/c cameron/c), cameron/c, (+/b david/c cameron/c)	1197
	36	Angela Merkel	merkel/c, (+/b chancellor/c merkel/c), (+/b angela/c merkel/c), (+/b chancellor/c angela/c merkel/c)	975
<i>female</i>	73	Theresa May	(+/b theresa/c may/c), (+/b prime/c minister/c theresa/c may/c)	429
	278	Nicola Sturgeon	sturgeon/c, (+/b nicola/c sturgeon/c)	112
	318	Suu Kyi	(s/b myanmar/c (+/b suu/c kyi/c)), (+/b aung/c san/c suu/c kyi/c), (s/b myanmar/c (+/b aung/c san/c suu/c kyi/c)), (+/b suu/c kyi/c)	99
	613	Malala Yousafzai	malala/c	51
<i>group</i>	49	The Palestinians	palestinians/c, (the/m palestinians/c)	658
	57	U.S. officials	(+/b us/c officials/c)	557
	110	The Kurds	kurds/c, (the/m kurds/c)	288
	138	The Russians	russians/c, (the/m russians/c)	243
	154	The Saudis	saudis/c, (the/m saudis/c)	213

Table 4: Five actors with highest hypergraphic degree in each category: non-human, male, female, group (in decreasing order of highest degree).

find commonalities across the hypergraph. Considering the hyperedge from one of the topic examples above, from (military/m (in/b engagement/c syria/c)) it is possible to extract concepts from inner edges that correspond to more general concepts, for example (syria/c), (engagement/c) and (in/b engagement syria/c). With the help of the implicit taxonomy, which indicates that (in/b engagement syria/c) is a type of engagement/c, a simple rule could also infer that (military/m (in/b engagement/c syria/c)) is a type of (military/m engagement/c syria/c).

In the various tables of results that we will subsequently present, actors and topics are represented by labels in natural language. During the transformation of text to hyperedge, every hyperedge that is generated is associated with the chunks of text from which it comes. These chunks are then used as textual labels for the hyperedges.

### 5.3 Inter-actor criticism

Focusing on France, Germany and Israel as target actors  $a$ , we gather the results for the detection of conflict patterns in the tables 5, 6 and 7. Each of these actors is involved in active or passive criticism of other actors, i.e.

either critical ( $\rightarrow$ ) or criticized by ( $\leftarrow$ ) other actors. The critique is related to a topic, and may go in both directions, i.e. Germany criticizes Greece for debt commitments (first row of table 6).

The topics presented here correspond to the detailed topics discussed in the previous section. This structured enumeration provides a way to scan the direction, target and frequency of claims by actors on other actors in a given text corpus.

actor	topic
$\Rightarrow$ assad	$\rightarrow$ scuppering syria peace talks
$\Rightarrow$ damascus	$\rightarrow$ war crimes in aleppo
$\Rightarrow$ russia	$\rightarrow$ continuing to use chemical weapons
$\Rightarrow$ the united states	$\rightarrow$ meddling in election
$\Rightarrow$ us	$\rightarrow$ weakening of europe
$\Rightarrow$ germany	$\rightarrow$ espionage
$\Leftarrow$ council of europe	$\leftarrow$ being europe's biggest problem child
$\Leftarrow$ kagame	$\leftarrow$ wage dumping in the meat sector
$\Leftarrow$ london	$\leftarrow$ allowing parents to hit and spank their children
$\Leftarrow$ un	$\leftarrow$ rwanda genocide
	$\leftarrow$ plot to wreck britain
	$\leftarrow$ racist attacks on black minister

Table 5: List of actors criticizing or being criticized by ego (here, France), and the topics over which the critique applies. Single arrows show the critique direction (left to right: ego criticizes that actor) for each underlying hyperedge, double arrows indicate the overall critique direction (which can thus go both ways).

actor		topic	
⇒	greece	→	debt commitments
⇒	iraqi kurds	→	one sided referendum plans
⇒	israel	→	latest settlement expansion in east jerusalem
⇒	maduro	→	holding venezuelans hostage
⇒	uk	→	leaving eu
⇒	ukraine	→	graft
⇔	russia	←	halting arms deal
		→	cyber attack on ukraine peace monitors
		→	military engagement in syria
⇔	us	←	causing instability
		→	ceding lead role to china
		→	stasi methods
⇔	france	←	wage dumping in the meat sector
		→	being europe's biggest problem child
⇔	turkey	←	cultural racism over eu accession remarks
		←	engaging in diplomatic rudeness
		←	genocide speech
		←	harbouring terrorists
		←	providing succor to its enemies
		←	providing succor to its enemies
		←	working against erdogan
		→	further distancing itself from europe by reinstating the death penalty after a disputed referendum
⇐	erdogan	→	nazi jibes amid referendum row
		→	supporting terrorism
		←	blocked political rallies
⇐	italy	←	nazi practices
⇐	marine le pen	←	undermining its economic efforts
⇐	moscow	←	opening doors to refugees for cheap labour
⇐	orban	←	hushing up russian girl's rape
⇐	snowden	←	rude tone over refugees
⇐	un	←	aiding nsa in spying efforts
⇐	un committee	←	institutional racism and racist stereotyping against people of african descent
		←	violating an anti-racism convention by not prosecuting a politician's comments about turks and arabs

Table 6: List of actors criticizing or being criticized by ego (here, Germany), and the topics over which the critique applies. Single arrows show the critique direction (left to right: ego criticizes that actor) for each underlying hyperedge, double arrows indicate the overall critique direction (which can thus go both ways).

## 5.4 Dyadic claims

Here we focus on claims that actors make about other actors (or themselves). In other words, we refer to claims where the subject of the claim is itself an actor. Furthermore, we consider only claims for which the claim relation contains an argument playing the rule of complement, meaning that the subject of the claim is being linked with some concept, for example expressing membership in a class (e.g.: “Pablo is a cat.”) or the possession of some property (e.g.: “North Korea is afraid”).

The predicate of the relative relation that expressed the claim is inspected to further determine the tense of the attribution (present, past, future, hypothetical), and to identify negations. Once again, this is achieved by simple rules over the hypergraphic representation:

- The presence of an auxiliary atom with sub-type “negation” implies that the attribution is negated, as is in fact the case with the first example of figure 3.
- The presence of one of the auxiliary atoms will/a, wo/a implies the future tense.
- The presence of one of the auxiliary atoms would/a, could/a, may/a, can/a implies an hypotehetical case.
- Otherwise, the verb tense directly encoded into the main predicate atom determines present of past tense.

actor		topic	
⇒	arabs	→	bashing it at nuclear un meeting
⇒	belgium	→	lax airport security
⇒	britain	→	secretly playing lead role in un resolution on settlements
⇒	european council	→	circumcision ruling
⇒	facebook	→	contributing to west bank violence
⇒	google	→	dashing peace hopes
⇒	hamas	→	abusing permits issued to gaza residents
⇒	lebanon	→	keeping crossing gaza
⇒	sweden	→	responding to alleged attack
⇔	eu	←	palestine
		→	latest settlement expansion in palestine
		←	bad deal on iran
		←	escalating violence
		←	harvesting organs of dead attackers
⇔	palestinians	←	violating gaza truce
		→	incitement over talks
		→	leaking news from peace talks
		←	destruction of jordan valley homes
⇔	un	←	emek shilo settlement
		←	violating health of syrians
		→	spreading propaganda
		←	assassins hiring to out take nuclear scientists
⇔	iran	←	being behind beirut bombings
		←	damaging nuclear talks
		←	syria crisis
		→	nuclear weapons tests
		←	committing genocide
⇔	abbas	←	executing 13-year old
		←	igniting religious war
		←	jerusalem holy site
		←	religious war
		→	using wild boars against palestinians
		→	sabotaging peace efforts as palestinian president moves to form government with hamas
		←	aggression
⇔	syria	←	attacking near damascus for the second time in a week
		←	bombardment
		←	bombing damascus airport
		←	crushing response to strikes on damascus airbase
		←	surprise retaliation
		→	golan heights clashes
		←	advancing settlements in east jerusalem
		←	annexing west bank
⇔	us	←	demolishing palestinian town
		←	demolishing palestinian village in west bank
		←	excessive force using against palestinians
		←	immediate crisis
		←	systematically seizing palestinian land
		←	targeting family of murdered palestine teen
		→	targeting kin of murdered palestinian teen
		→	backing palestinian unity government
⇐	amnesty	←	gaza war crimes
⇐	amnesty international	←	use of force in land day demonstrations
⇐	arab league	←	continuing to harm al aqsa
⇐	arab league chief	←	stealing palestinian books
⇐	assad	←	destablising syria
⇐	assad regime	←	supporting terrorists in syria
		←	being al qaeda's air force as conflict edges closer to shared border
⇐	bds	←	repression
⇐	brazilian president	←	carrying out a massacre in gaza
⇐	british	←	undermining iran deal
⇐	correa	←	committing genocide and cancels visit to israel
⇐	david cameron	←	gaza
⇐	germany	←	latest settlement expansion in east jerusalem
⇐	group	←	cave in
⇐	hezbollah	←	assassinating commander
⇐	hezbollah chief	←	attack
⇐	hrw	←	abusive arrests of palestinian kids
		←	abusive arrests of palestinian children as young as 11
⇐	human rights watch	←	and threats of using them to force to sign confessions
		←	war crimes in gaza
⇐	iran's khamenei	←	al aqsa
⇐	jordan	←	state terrorism in jerusalem
⇐	jordan's king	←	western wall prayer platform
⇐	lgbt activists	←	jerusalem provocation
⇐	nasrallah	←	pinkwashing palestinian occupation
⇐	nasrallah	←	any stupid moves in lebanon
⇐	palestine	←	talks failure
⇐	palestinian authority	←	jerusalem shooting terrorists
⇐	palestinian leader	←	killing of palestinian attackers
⇐	palestinian leader abb	←	religious war
⇐	palestinian president mahmud abbas	←	war of genocide
⇐	ru <sup>ssia</sup>	←	committing genocide in gaza
⇐	the united nations	←	jets detected near syria-lebanon border
		←	illegally demolishing houses of palestinians in east jerusalem
⇐	turkey	←	helping overthrow cairo's islamist president
⇐	un agency	←	new settlements in east jerusalem
⇐	un chief	←	imposing apartheid regime on palestinians
⇐	unesco	←	settlement plans in wake of quartet report
		←	aggression on temple mount
⇐	united church of chris	←	restricting worship at al aqsa
⇐	un panel	←	treatment of palestinian children
⇐	un report	←	detaining migrant children
⇐	un rights chief	←	child torture
⇐	un rights expert	←	war crimes committing in gaza
⇐	un's ban ki moon	←	excessive force against palestinians
⇐	us envoy	←	breeding palestinian terror attacks
		←	turning a blind eye to settlers' violence against palestinians saying the jewish state employs two sets of rules for israelis and palestinians
⇐	us intel sources	←	sharing secrets with trump administration

Table 7: List of actors criticizing or being criticized by ego (here, Israel), and the topics over which the critique applies. Single arrows show the critique direction (left to right: ego criticizes that actor) for each underlying hyperedge, double arrows indicate the overall critique direction (which can thus go both ways).

source		target		property
north korea	says	north korea	is	able to nuke us mainland a great place for human rights close to developing a new satellite just the place for you open to talks with south korea open ready for war with us ready to deploy ready to strike us aircraft carrier the victim of intensive cyberattacks <i>afraid of military us strike</i> <i>frightened by military us threat</i> <i>opposed to resuming dialogue with us</i> <i>responsible for righteous sony hacking</i>
putin	says	russia	was	ready to put nuclear forces on alert over crimea moral compass of the world willing to hand over to us house of representatives and senate interested in using national currencies with other brics
russia			is	<i>not a threat to anyone</i> open to coordinating with us in syria patience with us planning joint isis operations with france ready for dialogue with petro poroshenko next ukraine's president ready to deal with new ukraine president ready to provide the free syrian army with air support in fight against islamic state ready to retaliate for us election sanctions ready to cooperate with philippines president rodrigo duterte after us snub
obama			is	prepared to act on syria if diplomacy fails ready to aid france after attack
putin	says	us	will be was	<i>not tangled in venezuela's politics</i> the mastermind of the ukrainian coup world's only superpower
us			is	<i>not banana republic</i> ready to engage north korea in authentic credible negotiations after iran nuclear deal disappointed over china's failure to hand over fugitive intelligence analyst edward snowden open nuclear
			may be	<i>not surprised</i>

Table 8: List of claims, or attributions by subject actors (sources) about other actors (targets). Automatically identified negative claims are emphasized.

In table 8, we present such attributions between the actors: North Korea, Russia, Putin, Obama and U.S.

## 5.5 Topic-based conflict network

So far we have presented actor-centric results. Here we will consider all conflicts that contain “Syria” as topic or subtopic (according to the definitions of section 5.2). From this set of hyperedges we extracted a directed network connecting actors engaging in expressions of conflict over Syria. A visualization of this network is presented in figure 4.

We devised a very simple algorithm to identify two factions in this conflict graph. Firstly, we attribute a score  $s_{ij}$  to every edge  $(e_{ij})$ :

$$s_{ij} = \min(d_i, d_j)$$

where  $d_i$  is the degree (in- and out-) of node  $i$ . Then we iterate through the edges in descending order of  $s$ . This heuristic assumes that edges connecting more active nodes are more likely to represent the fundamental dividing lines of the overall conflict. The first edge assigns one node to faction A and another to faction B. From then on, a node is assigned to a faction if it does not have a

conflict with any current member of this faction, and has a conflict with a current member of the opposite faction. In the case that the node cannot be assigned to any faction, it remains unassigned.

This resulted in faction A containing the actors: {russia, iran, assad, moscow, putin, china, damascus, erdogan, tunisia}, faction B the actors: {us, west, israel, turkey, the united states, kerry, france, un, netanyahu, uk, germany} and the following actors remaining unassigned: {palestinians, obama}. Faction A is shown in blue in figure 4, and faction B in red. This categorization and network visualization suggest that the main axis of the conflict around Syria is a Russia / U.S conflict. Factions A and B contain state actors and political leaders that are typically aligned with, respectively, Russia and the U.S.

Naturally, more sophisticated faction and alliance detection methods can be employed. Here we are mostly interested in showing the effectiveness of our approach in summarizing complex situations from large natural language corpora, and to provide some empirical validation that these results are sensible.



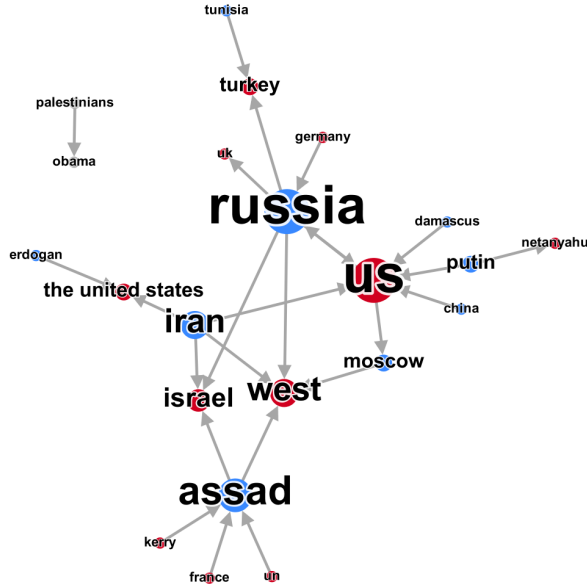


Figure 4: Network of conflicts between actors over the topic “Syria”. Arrows point from the originator of the conflict to its target. Size of nodes is proportional to their degree. Two factions were identified by a simple algorithm. One faction is represented as red, the other blue. Gray nodes do not belong to any faction.

## 5.6 Validation

In this article, it is not our intention to claim that any of our methods outperform others in some specific task. There is undoubtedly a lot of room for improvement in the various approaches that we presented here. Instead, we want to show that the semantic hypergraph formalism is useful in unifying different automated text analysis tasks in a synergistic way, that it allows for simple rules to have highly expressive power, and that it is already possible to perform a translation from natural language to a semantic hypergraph representation that is sufficiently correct for practical application in research, namely in the domain of the computational social sciences.

task	error type	error count	error rate
claim inference	not a claim	3/100	3%
	wrong actor	1/100	1%
	wrong topic	1/100	1%
	bad anaphora resolution	0/15	0%
	minor defects in topic	16/100	16%
conflict inference	not a conflict	1/100	1%
	wrong origin actor	2/100	2%
	wrong target actor	3/100	3%
	wrong topic	1/100	1%
	minor defects in topic	10/100	10%

Table 9: Evaluation of several types of error in claim and conflict inference. Error counts and rates are presented, based on the manual inspection of 100 randomly selected claims and 100 randomly selected conflicts.

In table 9 we present an evaluation of accuracy based on the manual inspection of 100 claims and 100 conflicts that were randomly selected from the hypergraph. Defects are deemed to be minor if they do not interfere with the overall meaning of the hyperedge (e.g., by leading to one of the other, more serious errors listed in the table). To illustrate with a minor defect from our dataset:

```
(says/p ((has/a (never/a left/p)) (+/b
cold/c war/c) (+/b putin/c 's/c mind/c))
harper/c (in/t germany/c))
```

In this case, the concept “Putin’s mind” would be better represented by the hyperedge (’s/b putin/c mind/c).

In conjunction with the qualitative appraisal of the results presented in this article, these error rates offer some confidence that the method is sufficiently accurate to be useful.

## 6 Discussion

In this article we have explored what we believe to be a fertile idea: that more powerful methods of analysis of text corpora in natural language – powerful in the sense of richer in meaning, or closer to natural language understanding – can be organized around the semantic hypergraph formalism. We have shown how conventional NLP methods can be leveraged to convert text to this hypergraph representation. Then, we have shown an example of application of this representation in a real text analysis task: generating overviews of claims and conflicts across several years of international news headlines, organized by the perspective of individual actors, intersections of actors and topics of conflict.

We bet here on an hybrid approach to computational intelligence, combining modern machine learning methods – statistical in nature – with symbolic systems. The idea is to take advantage of the strengths of each branch while minimizing their weaknesses. With machine learning it is possible to train models that make the complexity more tractable for symbolic methods, allowing researchers to take advantage of the understandability of the latter.

We believe that the results that we have shown in section 5 illustrate the potential of this approach to unify various text analysis tasks in a principled way, namely named-entity recognition (NER), sentiment analysis and topic detection, as well as extracting more meaning than what is typical of contemporary approaches.

There is undoubtedly much room for improvement. For example, the usefulness of inference can be expanded, both you more sophisticated inference methods and with the help of general-purpose knowledge bases combined with more powerful disambiguation and synonym detection.

All the methods and formalisms described in this article have been implemented in the *GraphBrain*<sup>3</sup> free software / open source package. We released this package as a research tool, in the hope that the work presented here can be of use to other researchers, and expanded upon.

**Acknowledgements.** The authors wish to thank Chih-Chun Chen for interesting preliminary discussions. This work has been partially supported by the “Algodiv” grant (ANR-15-CE38-0001) funded by the ANR (French National Agency of Research) and by the “Socsemics” Consolidator grant funded by the European Research Council (ERC) under the European Union’s Horizon 2020 research and innovation programme (grant agreement No. 772743).

## References

- [1] B. Adida, M. Birbeck, S. McCarron, and S. Pembrton. Rdfa in xhtml: Syntax and processing. *Recommendation, W3C*, 7, 2008.
- [2] E. Agirre and A. Soroa. Personalizing pagerank for word sense disambiguation. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 33–41. Association for Computational Linguistics, 2009.
- [3] C. Alberti, D. Andor, I. Bogatyy, M. Collins, D. Gillick, L. Kong, T. Koo, J. Ma, M. Omernick, S. Petrov, et al. Syntaxnet models for the conll 2017 shared task. *arXiv preprint arXiv:1703.04929*, 2017.
- [4] J. F. Allen and A. M. Frisch. What’s in a semantic network? In *Proceedings of the 20th annual meeting on Association for Computational Linguistics*, pages 19–27. Association for Computational Linguistics, 1982.
- [5] D. Andor, C. Alberti, D. Weiss, A. Severyn, A. Presta, K. Ganchev, S. Petrov, and M. Collins. Globally normalized transition-based neural networks. *arXiv preprint arXiv:1603.06042*, 2016.
- [6] T. Berners-Lee and J. Hendler. Publishing on the semantic web. *Nature*, 410(6832):1023, 2001.
- [7] D. M. Blei. Probabilistic topic models. *Communications of the ACM*, 55(4):77–84, 2012.
- [8] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.
- [9] H. Boley. Directed recursive labelnode hypergraphs: A new representation-language. *Artificial Intelligence*, 9(1):49–85, 1977.
- [10] A. Bretto. Hypergraph theory. *An introduction. Mathematical Engineering. Cham: Springer*, 2013.
- [11] C. Cattuto, C. Schmitz, A. Baldassarri, V. D. Servedio, V. Loreto, A. Hotho, M. Grahl, and G. Stumme. Network properties of folksonomies. *Ai Communications*, 20(4):245–262, 2007.
- [12] D. Chavalarias, J. D. Wallach, A. H. T. Li, and J. P. Ioannidis. Evolution of reporting p values in the biomedical literature, 1990–2015. *Jama*, 315(11):1141–1148, 2016.
- [13] J. D. Choi, J. Tetreault, and A. Stent. It depends: Dependency parser comparison using a web-based evaluation tool. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 387–396, 2015.
- [14] M.-C. De Marneffe, T. Dozat, N. Silveira, K. Haverinen, F. Ginter, J. Nivre, and C. D. Manning. Universal stanford dependencies: A cross-linguistic typology. In *LREC*, volume 14, pages 4585–4592, 2014.
- [15] C. Eslahchi and A. Rahimi. Some properties of ordered hypergraphs. *MAT. BEC.*, 59:9–13, 2007.
- [16] B. Goertzel. Patterns, hypergraphs and embodied general intelligence. In *Neural Networks, 2006. IJCNN’06. International Joint Conference on*, pages 451–458. IEEE, 2006.
- [17] M. Honnibal, M. Johnson, et al. An improved non-monotonic transition system for dependency parsing. In *EMNLP*, pages 1373–1378, 2015.
- [18] B. Iordanov. HyperGraphDB: A generalized graph database. In H. T. Shen, J. Pei, M. T. Özsu, L. Zou, J. Lu, T.-W. Ling, G. Yu, Y. Zhuang, and J. Shao, editors, *Web-Age Information Management*, pages 25–36, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.
- [19] M. Lippi and P. Torroni. Argumentation mining: State of the art and emerging trends. *ACM Transactions on Internet Technology (TOIT)*, 16(2):10, 2016.
- [20] W. Lowe. Understanding wordscores. *Political Analysis*, 16(4):356–371, 2008.
- [21] J. McCarthy. Recursive functions of symbolic expressions and their computation by machine, part i. *Communications of the ACM*, 3(4):184–195, 1960.

---

<sup>3</sup><http://graphbrain.net>

- [22] R. Mihalcea. Unsupervised large-vocabulary word sense disambiguation with graph-based algorithms for sequence data labeling. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 411–418, 2005.
- [23] R. Mihalcea and P. Tarau. Textrank: Bringing order into text. In *EMNLP*, volume 4, pages 404–411, 2004.
- [24] B. L. Monroe, M. P. Colaresi, and K. M. Quinn. Fightin’ words: Lexical feature selection and evaluation for identifying the content of political conflict. *Political Analysis*, 16(4):372–403, 2008.
- [25] K. Murakami, E. Nichols, J. Mizuno, Y. Watanabe, S. Masuda, H. Goto, M. Ohki, C. Sao, S. Matsuyoshi, K. Inui, et al. Statement map: reducing web information credibility noise through opinion classification. In *Proceedings of the fourth workshop on Analytics for noisy unstructured text data*, pages 59–66. ACM, 2010.
- [26] D. Nadeau and S. Sekine. A survey of named entity recognition and classification. *Linguisticae Investigationes*, 30(1):3–26, 2007.
- [27] R. Navigli and M. Lapata. Graph connectivity measures for unsupervised word sense disambiguation. In *IJCAI*, pages 1683–1688, 2007.
- [28] B. Pang, L. Lee, et al. Opinion mining and sentiment analysis. *Foundations and Trends® in Information Retrieval*, 2(1–2):1–135, 2008.
- [29] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*, 2018.
- [30] J. Ramos et al. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning*, volume 242, pages 133–142, 2003.
- [31] S. Reddy, O. Täckström, M. Collins, T. Kwiatkowski, D. Das, M. Steedman, and M. Lapata. Transforming dependency structures to logical forms for semantic parsing. *Transactions of the Association for Computational Linguistics*, 4:127–140, 2016.
- [32] A. Ritter, S. Clark, O. Etzioni, et al. Named entity recognition in tweets: an experimental study. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1524–1534. Association for Computational Linguistics, 2011.
- [33] C. Roth. Socio-semantic frameworks. *Advances in Complex Systems*, 16(04n05):1350013, 2013.
- [34] P. Ruiz, C. Plancq, and T. Poibeau. More than word cooccurrence: Exploring support and opposition in international climate negotiations with semantic parsing. In *LREC: The 10th Language Resources and Evaluation Conference*, page 1902, 2016.
- [35] N. Shadbolt, T. Berners-Lee, and W. Hall. The semantic web revisited. *IEEE intelligent systems*, 21(3):96–101, 2006.
- [36] A. Singhal. Modern information retrieval: A brief overview. *IEEE Data Eng. Bull.*, 24(4):35–43, 2001.
- [37] W. M. Soon, H. T. Ng, and D. C. Y. Lim. A machine learning approach to coreference resolution of noun phrases. *Computational linguistics*, 27(4):521–544, 2001.
- [38] J. F. Sowa. *Principles of semantic networks: Explorations in the representation of knowledge*. Morgan Kaufmann, 2014.
- [39] S. Staab and R. Studer. *Handbook on ontologies*. Springer Science & Business Media, 2010.
- [40] W. Van Atteveldt, J. Kleinnijenhuis, and N. Ruigrok. Parsing, semantic networks, and political authority using syntactic analysis to extract semantic relations from dutch newspaper articles. *Political Analysis*, 16(4):428–446, 2008.
- [41] W. van Atteveldt, T. Sheaffer, S. R. Shenhav, and Y. Fogel-Dror. Clause analysis: using syntactic information to automatically extract source, subject, and predicate from texts with an application to the 2008–2009 gaza war. *Political Analysis*, pages 1–16, 2017.
- [42] H. Wang, D. Can, A. Kazemzadeh, F. Bar, and S. Narayanan. A system for real-time twitter sentiment analysis of 2012 US presidential election cycle. In *Proceedings of the ACL 2012 System Demonstrations*, pages 115–120. Association for Computational Linguistics, 2012.

## A Hyperedge encodings: subtypes, roles and other details

For the sake of clarity, we have left some encoding details out of the main text. The 11 hyperedge types presented in table 2 represent the complete set of building blocks that allow semantic hypergraphs to express arbitrary constructs in natural language. However, it is possible to further extend these basic types with information that helps to perform automatic inference tasks.

In table A.1 we present all the subtypes generated by the parser we used in this work. Unlike the main types,

supertype	type	subtype	example	
concepts	concept	<i>c</i>		
		common	<i>cc</i>	apple/cc
		proper	<i>cp</i>	mary/cp
		number	<i>cn</i>	27/cn
		pronoun	<i>ci</i>	she/ci
		interrogative	<i>cw</i>	who/cw
connectors	predicate	<i>p</i>		
		declarative	<i>pd</i>	is/pd
		interrogative	<i>p?</i>	is/p?
		imperative	<i>p!</i>	go/p!
		conceptual	<i>pc</i>	go/pc
		meta	<i>pm</i>	and/pm
	builder	<i>b</i>		
		possessive	<i>bp</i>	's/bp
		relational	<i>br</i>	in/br
	auxiliary		<i>b+</i>	and/b+
				been/a
	modifier		<i>an</i>	not/an
	meta modifier	<i>a</i>		
		negation		
	subpredicate	<i>m</i>		
		adjective	<i>ma</i>	green/ma
		possessive	<i>mp</i>	my/mp
		determinant	<i>md</i>	the/md
		number	<i>mn</i>	100/mn
				genetically/w
				by/x
	trigger	<i>w</i>		
		<i>x</i>		
		<i>t</i>		
		conditional	<i>t?</i>	if/tc
		temporal	<i>tt</i>	when/tt
		local	<i>tl</i>	where/tl
		modal	<i>tm</i>	modal/tm
		causal	<i>t&gt;</i>	because/t>
		comparative	<i>t=</i>	like/t=
		concessive	<i>tc</i>	although/tc
relations	relation	<i>r</i>	<i>same as predicates</i> (is/p mary/c nice/c)	
arguments	dependent	<i>d</i>	(by/x scientists/c)	
	specifier	<i>s</i>	<i>same as triggers</i> (in/t 1976/c)	

Table A.1: Full taxonomy of entity types with subtypes.

this is not meant to be a closed set. As seen in the table, subtypes are specified simply by adding a character in front of the main type character. For example, this makes this possible to distinguish a common concept (e.g.: sky/cc) from a concept related to a proper noun (e.g.: europe/cp).

We further introduce a dot separator to enrich an atom with type-specific additional information. For example, concepts can be annotated as being singular (e.g.: apple/cn.s) or plural (e.g.: apples/cn.p).

In this work we make use and allude in the main text to another type-specific annotation: *argument roles* for predicates and builders. These are sequences of characters that indicate the role of their respective argument following the connector. The case of builders is very simple, as there are only two possible roles: “main” (denoted by *m*) and “auxiliary” (denoted by *a*). To give an example:

(+/b.am tennis/c ball/c)

It can be inferred from the argument roles annotation



that ball/c is the main concept in the construct, meaning that (+/b.am tennis/c ball/c) is a type of ball/c.

type	code
subject	s
passive subject	p
agent	a
subject complement	c
direct object	o
indirect object	i
specifier	x
relative (inner) relation	r
parataxis	t
remnant	r
reparandum	e
dislocated	d
vocative	v

Table A.2: Predicate argument roles.

The same applies to predicates, but here there are more possible roles. They are shown in table A.2 and indicate common roles that arguments can play in relations. For example, the argument roles string “sio” would indicate that the three arguments following the predicate in a relation, respectively play the roles of subject, indirect object and direct object. To illustrate:

```
(gave/p.d.sio john/cp.s mary/cp.s (a/m
    flower/cn.s))
```

Finally, namespaces are a way to distinguish atoms that correspond to the same word in a given language, but point to different concepts, for example: cambridge/cp.s/1 could point to Cambridge, U.K. and cambridge/cp.s/2 to Cambridge, Mass. The namespace can also be used to encode the language of origin of an atom, especially useful when dealing with multilingual corpora. In this work we do not perform disambiguation that requires this type of distinction, but this representation possibility allows for the disambiguation of homographs in future works.

## B Mapping Universal Stanford Dependencies to hyperedges

We used the *Universal Stanford Dependencies* to guide the development of our knowledge model. In table B.1, we present one example of hyperedge for each grammatical relation in the Universal Dependencies. This is meant as empirical evidence for the completeness of our model, in terms of its ability to map to natural language constructs in most human languages.

Gramatical Relation	Hyperedge Example
<b>nsubj</b> : nominal subject	(is/pd.sc (the/m <u>dog/c</u> ) cute/c)
<b>nsubjpass</b> : passive nominal subject	((was/a played/pd.pa) (the/m <u>piano/c</u> ) (by/x mary/c))
<b>dobj</b> : direct object (accusative)	(gave/pd.sio mary/c john/c (a/m <u>gift/c</u> ))
<b>iobj</b> : indirect object (dative)	(gave/pd.sio mary/c <u>john/c</u> (a/m gift/c))
<b>csubj</b> : clausal subject	(makes/pd.so ( <u>said/pc.os</u> what/c she/c) sense/c)
<b>csubjpass</b> : clausal passive subject	((was/a suspected/pd.pa) (that/x ( <u>lied/pc.s</u> she/c)) (by/x everyone/c))
<b>ccomp</b> : clausal complement	(says/pd.so he/c ( <u>like/pd.so</u> you/c (to/m swim/c)))
<b>xcomp</b> : open clausal complement	(says/pd.so he/c (like/pd.so you/c (to/m <u>swim/c</u> )))
<b>nmod</b> : nominal modifier	( <u>some/m</u> (of/m (the/m toys/c)))
<b>advcl</b> : adverbial clause modifier	(talked/pd.sxx he/c (to/t him/c) (to/t ( <u>secure/pc.o</u> (the/m account/c))))
<b>advmod</b> : adverb modifier	(( <u>genetically/w</u> modified/m) food/c)
<b>neg</b> : negation modifier	(( <u>not/an</u> is/pd.sc) bill/c (a/m scientist/c))
<b>vocative</b> : vocative	(know/pd.sv i/c <u>john/c</u> )
<b>discourse</b> : discourse	N/A
<b>expl</b> : expletive	(( <u>there/a</u> is/pd) (a/m (in/b ghost/c (the/m room/c))))
<b>aux</b> : auxiliary	(( <u>has/a</u> (been/a killed/pd.p)) kennedy/c)
<b>auxpass</b> : passive auxiliary	((has/a ( <u>been/a</u> killed/pd.p)) kennedy/c)
<b>cop</b> : copula	(is/pd.sc bill/c <u>big/c</u> )
<b>mark</b> : marker	(says/pd.so he/c ( <u>that/x</u> (like/pd.so you/c (to/m swim/c))))
<b>punct</b> : punctuation	N/A
<b>conj</b> : conjunction	(is/pd.so bill/c (and/b big/c <u>honest/c</u> ))
<b>cc</b> : coordination	(is/pd.so bill/c ( <u>and/b</u> big/c honest/c))
<b>nummod</b> : numeric modifier	(ate/pd.so sam/c ( <u>3/m</u> sheep/c))
<b>relcl</b> : relative clause modifier	(saw/pd.so i/c (the/m ( <u>love/pc.so</u> you/c man/c)))
<b>det</b> : determiner	(is/pd.sc ( <u>the/m</u> man/c) here/c)
<b>compound</b> : compound	(has/pd.so john/c (the/m (+/b phone/c <u>book/c</u> )))
<b>name</b> : multi-word proper nouns	(+/b <u>marie/c curie/c</u> )
<b>mwe</b> : multi-word expression	(cried/pd.sx he/c ( <u>because/t.&gt;</u> (of/m you/c)))
<b>foreign</b> : foreign words	<i>misc.</i>
<b>goeswith</b> : goes with	(come/pd.sox they/c here/c (without/t.m permission/c))
<b>case</b> : case marking	(the/m ( <u>'s/b</u> school/c grounds/c))
<b>list</b> : list	('s/b mary (list/c (:/b <u>phone/c</u> 555-981/c) (:/a <u>age/c</u> 33/c)))
<b>dislocated</b> : dislocated elements	(is/pd.scd this/c (our/m office/c) (and/b me/c sam/c))
<b>parataxis</b> : parataxis	(left/pd.tsi ( <u>said/p.d.s</u> john/c) (the/m guy/c) (in/b early/c (the/m morning/c)))
<b>remnant</b> : remnant in ellipsis	(won/pd.sor john/c bronze/c (+/x <u>mary/c silver/c</u> ))
<b>reparandum</b> : overridden disfluency	(go/pl.eo (to/x (the/m <u>right/c</u> )) (to/x (the/m left/c)))
<b>root</b> : sentence head	( <u>is/pd.sc</u> (the/m dog/c) cute/c)
<b>dep</b> : unspecified dependency	<i>misc.</i>

Table B.1: Examples of hyperedges for each grammatical relation in the Universal Dependencies.