Knowlywood: Mining Activity Knowledge From Hollywood Narratives

Niket Tandon
Max Planck Institute for Informatics
ntandon@mpi-inf.mpg.de

Gerard de Melo
Tsinghua University, China
demelo@tsinghua.edu.cn

Abir De
IIT Kharagpur, India
abir.de@cse.iitkgp.ernet.in

Gerhard Weikum
Max Planck Institute for Informatics
weikum@mpi-inf.mpg.de

ABSTRACT

Despite the success of large knowledge bases, one kind of knowledge that has not received attention so far is that of human activities. An example of such an activity is proposing to someone (to get married). For the computer, knowing that this involves two adults, often but not necessarily a woman and a man, that it often takes place in some romantic location, that it typically involves flowers or jewelry, and that it is usually followed by kissing, is a valuable asset for tasks like natural language dialog, scene understanding, or video search.

This corresponds to the challenging task of acquiring semantic frames that capture human activities, their participating agents, and their typical spatio-temporal contexts. This paper presents a novel approach that taps into movie scripts and other narrative texts. We develop a pipeline for semantic parsing and knowledge distillation, to systematically compile semantically refined activity frames.

The resulting knowledge base contains hundreds of thousands of activity frames, mined from about two million scenes of movies, TV series, and novels. A manual assessment study, with extensive sampling and statistical significance tests, shows that the frames and their attribute values have an accuracy of at least 80 percent. We also demonstrate the usefulness of activity knowledge by the extrinsic use case of movie scene search.

Categories and Subject Descriptors

I.2.7 [**Natural Language Processing**]: Text Analysis

Keywords

Activity Knowledge; Commonsense Knowledge Acquisition

1. INTRODUCTION

Motivation and Problem. Knowledge graphs like DBpedia, Freebase, or Yago [2, 5, 40] have become major assets for enriching the Web towards more semantic search and recommendations. They are heavily used at large companies such as Baidu, Facebook, Google, Microsoft, and others. The emphasis in these settings is on individual entities like people, organizations, and products or creative works, with focus on *factual knowledge* about such entities (e.g., songs and awards of an artist, CEOs and products of companies, cities and restaurants visited by friends, etc.).

Recently, however, with ground-breaking new products like Amazon Echo as well as assistants like Google Now, Microsoft's Cortana, and Apple's Siri, there is a strong need for commonsense knowledge enabling smart interpretation of queries relating to *everyday human activities*.

Unfortunately, fact-oriented knowledge graphs do not provide detailed knowledge of human activities. The same holds for more commonsense-oriented knowledge bases, from the seminal projects Cyc [22] and WordNet [15] to more recent endeavors such as ConceptNet [17], WebChild [41], and NEIL [9]. While these contain millions of assertions between general concepts, referring to predicates like *isPartOf* (e.g., *engine isPartOf car*), *usedFor* (e.g., *car usedFor transportation*, *hasTaste* (e.g., *chocolate hasTaste sweet*), *hasShape* (e.g., *apple hasShape round*), *occursInScene* (e.g., *car occursInScene streetTraffic*), and more, they do not deliver fine-grained information about large numbers of specific activities.

In this paper, we fill this void by automatically compiling large amounts of knowledge about human activities from narrative text. For example, `climbing a mountain` should be a known activity, along with attributes like participating agents — a human, especially a climber, typical location, and time of day. This knowledge should be organized in a frame-style representation, as illustrated in Figure 1. Further, the activities must be semantically grouped (here with hiking up a hill), and these semantic groups must be hierarchically arranged. These activity groups should also be temporally linked to typical previous and next activities. Having this sort of data can greatly improve computer behavior in tasks like natural language dialog, scene understanding, or video search.

While parts of our approach could be applied to other genres, we focus on narrative text because it possesses some attractive yet under-utilized properties. Rather than being limited to newsworthy events, narrative text may include descriptions of common, rather mundane everyday activities. These are often described in a very detailed way and in chronological order with marked boundaries. For instance, we may find that one often unlocks a door before entering a building. Finally, we wish to connect our knowledge to visual content in movies, which enables several new applications, including the ones we consider later in Section 7.

Recent work in computer vision [34] has manually compiled a small collection of activity scripts, based on short videos about cooking. This contains about 65 different activities such as *melting butter* or *cooking pasta*, with attributes *tool=pan* or *tool=sieve*. Our goal is to broaden and automate the construction of these kinds of semantic frames, in order to populate a comprehensive *activity knowledge base*, in which, all concepts are sense-disambiguated

CIKM'15, October 19–23, 2015, Melbourne, Australia.

DOI: http://dx.doi.org/10.1145/2806416.2806583.

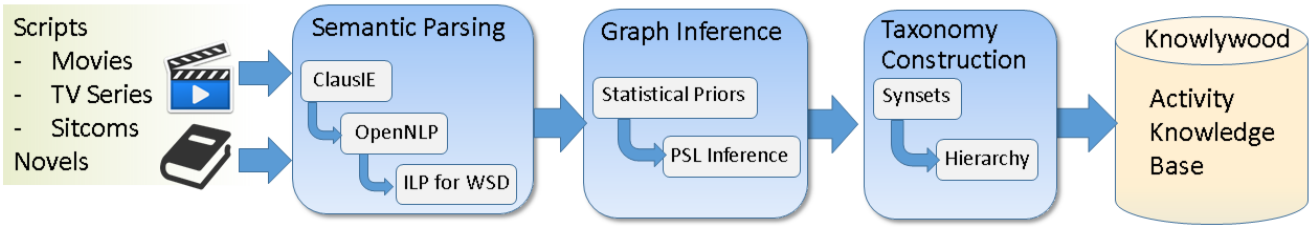

Figure 2: Knowlywood System Overview

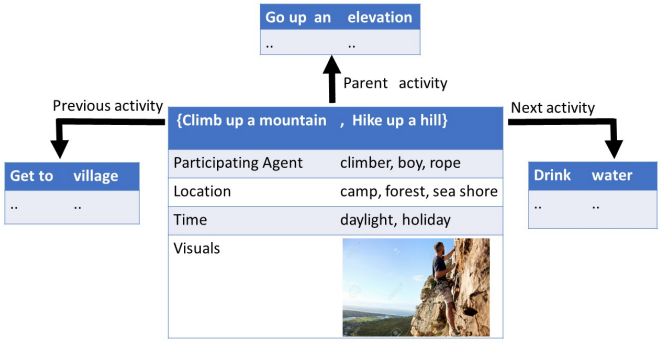

Figure 1: Activity Frame Example

and thus canonicalized with regard to high-quality linguistic resources like WordNet or VerbNet [36, 18].

Approach and Contribution. We have developed an advanced pipeline for semantic parsing and knowledge distillation, which allows us to systematically compile semantically refined activity frames from scripts and novels. We have processed nearly 2 million scenes from 560 movies, 460 TV series, and 100 books, and constructed a high-quality activity knowledge base with almost one million frames. Specifically, we represent activities as JSON objects which gives us typed attributes (also known as slots in knowledge representation terminology) and set-valued entries for attributes (also known as values or fillers). JSON is a popular format for data export/import. Our frames can also be easily cast into RDF triples. Overall, our contributions are:

- The first system, called Knowlywood, that automatically acquires detailed knowledge about activities, by tapping into movie/TV scripts and narrative texts and combining semantic parsing techniques for candidate generation with probabilistic inference and graph algorithms for candidate cleaning.
- New techniques for sense disambiguation of multi-word phrases (mapping them to WordNet) and taxonomy induction for activities, as building blocks of our construction pipeline.
- A large knowledge collection with nearly one million activities in the form of semantic frames, and with linkage to visual contents where activities occur. This Knowlywood collection will be made publicly accessible.[1] Its high quality has been confirmed by manual assessment with extensive sampling.

Our activity frames are valuable for use-cases such as video search, provide background knowledge for human-computer dialog, and can aid tasks like video scene understanding and the generation of textual descriptions for visual contents. Note also that the developed methodology is general and can be applied to other input sources if available, for example, personal diaries or travel logs.

[1]http://tinyurl.com/knowlywood

2. SYSTEM OVERVIEW

Figure 2 illustrates the Knowlywood pipeline of methods and tools. For automatically building the Knowlywood KB, we take the following main steps:

- **Semantic Parsing:** We first apply information extraction techniques on our input sources and then feed the output into a novel technique for semantic parsing, based on identifying clauses, mapping words and phrases to WordNet and VerbNet, and using integer linear programming (ILP) for the final disambiguation and construction of candidate activity frames.
- **Graph Inference:** We use the output data of the first stage to construct a preliminary activity knowledge graph, with noise and false positives. We then use Probabilistic Soft Logic (PSL) for efficient inference to construct a cleaner graph as consistent, high-quality output.
- **Taxonomy Construction:** We merge activities into equivalence classes, so-called *synsets* in the terminology of lexical resources (e.g., WordNet). An example is merging activity *propose to girlfriend* with *propose to fiancee*. Finally, we construct a subsumption hierarchy of activity synsets, which connects activities by the *hasType* relation. An example is: *propose to girlfriend* hasType *propose to someone*.

We additionally attach video frames to activities. To align scenes in movie scripts with their respective video frames, we exploit timestamp information in subtitle data.

Computational Model. The input to our methods is primarily scripts about movies or episodes of TV series. Figure 3 shows an example from the movie "Sex and the City" (obtained from the website `imsdb.com`). Although this is in a free format, there is some structure that we can exploit. Specifically, there are cues for detecting scene boundaries, we can identify speakers, and we can extract short descriptions about the setting of a scene that typically precede the actual dialog. Also, there are short narrative texts in between dialogs. Our methods are primarily geared for narrative snippets such as "Big proposes to Carrie" or "Big and Carrie kiss". Section 3 discusses how to further process these snippets and extract semantically cleaner information.

Obviously, individual scripts may be too noisy for automated methods to extract any meaningful information. Our method leverages that certain cues for activities appear in several scenes of different movies. Further, our scope is beyond movie scripts, including sitcoms, TV series, and novels, providing us with a broad spectrum of activities and higher redundancy.

We treat *verbal phrases* in narrative snippets as surface expressions for activity candidates. Using NLP techniques , this gives us

cues such as "propose to a woman" and "kiss someone". Generally, we extract *verb-object* pairs, where the verb can have a preposition (e.g., "propose to") and the object is a noun phrase, potentially a multi-word phrase. Initially, these are still ambiguous words that may have many different meanings. Our methods map both verb and object to unambiguous senses, so-called *synsets* in the WordNet lexical thesaurus [15]. This is crucial for semantic interpretation, and also key to being able to combine cues from different scenes and to organize activities in a clean taxonomy. In the example, we would obtain *propose#5 woman#1* where #5 and #1 are the WordNet sense numbers of the ambiguous words "propose" and "woman". The result of this sense disambiguation forms the core of an activity frame.

Definition 1. *An* activity *is a pair* (v, o) *where* v *is a WordNet synset for a verb or verb phrase and* o *is a WordNet synset for a noun or noun phrase.*

Activities are then enriched by *attributes* (or frame slots) about location, time, and involved participants. The latter includes both the humans in an activity (e.g., man, woman, judge) and objects or props that play role in the activity (e.g., diamond or ring). We obtain cues for the location and time attribute values using NLP techniques as well as from the scene description (before the dialog starts), e.g., "apartment", "courthouse", "spring", "day". For the participants attributes, we extract cues from both the characters in a scene and noun phrases in narrative snippets or dialogs. All these will also be sense-disambiguated in the output for the KB.

Definition 2. *An* activity frame *is an activity enhanced with attribute values for location, time, and participants.*

- *For location, the allowed values are WordNet senses that are hyponyms (specialization) of the WordNet sense* location#1.
- *For time, the allowed values are hyponyms of the sense* time period#1 *or* event#1.
- *For participants, the allowed values are hyponyms of* living thing#1 *or* physical object#1.

Each attribute can have zero, one or multiple values.

Finally, as we may extract activity candidates from each scene, we can relate activities from successive scenes (if there is a typical pattern found in several movies). To this end, we introduce frame attributes *prev* for a previous activity and *next* for a following activity. This way we link different activity frames to form entire chains. In the example, *propose#5 woman#1* would be *next*-linked to *kiss#1 someone#1*.

Definition 3. *An* activity chain *is a sequence of temporally related activities connected by* prev *and* next *links.* A.next = B *and B.prev = A denote that activity A is often followed by activity B.*

3. SEMANTIC PARSING

We have devised a customized pipeline for semantic parsing that starts with the input scripts and extracts and disambiguates constituents, all the way to constructing a frame structure for candidate activities.

Consider the input sentence *He shot a video in the moving bus*. The output frame for this input is shown in the last column of Table 1. The activity name is given by the verb followed by an object (i.e., shoot#4;video#1). Note that the words in this frame are mapped to disambiguated *WordNet senses* [15], denoted by the numbers after the # symbols. If a phrase is absent in WordNet, e.g. *moving bus*, then we merely map its head word (*bus*). The other columns in

235 INT. PENTHOUSE APARTMENT – LATER – SPRING
Carrie and Big are on the carpeted floor.
Big proposes to Carrie.

BIG (CONT'D)
Carrie Bradshaw
love of my life -
will you marry me?

She nods. Speechless. Overcome. He smiles.

BIG (CONT'D)
See, this is why
there's a diamond.

236 INT. COURTHOUSE/ROOM – DAY – SPRING

Carrie stands with Big in front of a JUDGE.

JUDGE
By the power vested in me, by
the state of New York, I now
pronounce you husband and wife.
You may now kiss the bride

Big and Carrie kiss.

Figure 3: Excerpt from Movie Script

Table 1: Semantic parsing example

Phrase	WordNet Mapping	VerbNet Mapping	Output Frame
the man	man#1	Agent . animate	Agent: man#1
began to shoot	shoot#4	shoot#vn#3	Action: shoot#4
a video	video#1	Patient . solid	Patient:video#1
in	in	PP . in	
the moving bus	bus#1	NP . Location . solid	Location: moving bus#1

Table 1 show the input phrases (after chunking) and their mappings to WordNet senses and entries in VerbNet [18], as discussed below.

Sentence Analysis. We first use ClausIE [11] to decompose sentences into shorter clauses, whenever possible. These are then further decomposed by applying the OpenNLP (*opennlp.sourceforge.net*) maximum entropy model for chunking the text of each individual clause. In the example in Table 1, this results in the sentence being split as shown in the first column.

Sense and Argument Analysis. Understanding the verb is the most critical task for semantic interpretation. We address this by mapping the verb or verb phrase to its proper sense in WordNet [15], which in turn is linked with VerbNet [18], a manually curated linguistic resource for English verbs. For each verb class, VerbNet lists relevant thematic roles, semantic restrictions on the arguments, and syntactic frames. For example, for the main predicate verb *shoot* in our example sentence, VerbNet lists multiple candidate senses, and for the first of these, `shoot#vn#1`, it provides, among others, the following syntactic frame:

Agent.animate V Patient.animate PP Instrument.solid

This would match *He shot the man with a gun*. Here, several roles are accompanied by a semantic constraint, known as a *selectional restriction*. A selectional restriction such as `animate` for the patient requires that this patient be a living being when used in the given syntactic frame. This can guide the choice of the proper WordNet mappings for the objects and for other words. For instance, *the man* in our example sentence could be disambiguated as `man#1`, which in turn is in a *hasInheritedHypernym* relationship with `living_thing#1`, which leads us to the `animate` label from VerbNet, and helps us find the right VerbNet sense for the verb.

These dependencies are captured as constraints in our joint disambiguation method, based on Integer Linear Programming (ILP) — discussed below. The ILP method uses prior weights obtained from simpler heuristics for word sense disambiguation (WSD) — discussed next.

WSD Priors. For an initial disambiguation of individual words and phrases, we use the state-of-the-art WSD system It-Makes-Sense (IMS) [43], which relies on supervised learning. We obtain the following scores for mapping a word i to sense j:

$$\tau_{ij} = \begin{cases} \text{score of IMS} & \text{for } j \in S_{\text{W}} \\ \sum_{j'} \tau_{ij'} & \text{for } j \in S_{\text{V}} \text{ linked to } j' \in S_{\text{W}} \end{cases}$$

Here, S_{W} denotes the set of candidate WordNet senses for the verbs and S_{V} denotes the set of candidate VerbNet senses. Note that VerbNet is much smaller and thus coarser-grained than WordNet, hence the summation over all WordNet senses linked with the same VerbNet verb.

An additional feature used in the ILP later are the most-frequent-sense ranks that WordNet provides, based on manual annotation of a large news corpus:

$$\theta_{ij} = \begin{cases} \text{1/rank(j for i)} & \text{for } j \in S_{\text{W}} \\ \sum_{j'} \theta_{ij'} & \text{for } j \in S_{\text{V}} \text{ linked to } j' \in S_{\text{W}} \end{cases}$$

Finally, we compute syntactic and semantic priors based on how well the input verb matches a VerbNet entry:

syn_{ij} frame match score for word i and VerbNet sense j

sem_{ij} selectional restriction score of the roles in a VerbNet frame j for word i

ILP Model. For the *joint* disambiguation of all words in the input sentence, we have devised an ILP with *binary decision variables* x_{ij} set to 1 if word i is mapped to sense j (in WordNet and/or VerbNet). V denotes the set of all input words or phrases i that are verb chunks. Our ILP is defined as follows:

maximize

$$\sum_{i,j} x_{ij}(\alpha\tau_{ij} + \beta_1\theta_{ij} + \beta_2\text{syn}_{ij} + \beta_3\text{sem}_{ij})$$

subject to

$$\begin{array}{lcll} \sum_{j \in S_{\text{V}}} x_{ij} & \leq & 1 & \forall i \in V \\ x_{ij} & \leq & x_{ij'} & \forall i \in V, j \in S_{\text{W}}, \\ & & & j \textit{ mapped to } j' \in S_{\text{V}} \\ x_{i_0 j_0} & \leq & x_{ij} & \forall i_0 \in V, j \in S_{\text{V}}, \\ & & & x_{ij} \in \textit{role-restr}(x_{i_0 j_0}) \\ \sum_{j} x_{ij} & \leq & 1 & \forall i \notin V \\ x_{ij} & \in & \{0,1\} & \end{array}$$

The objective function combines the various prior scores, with coefficients tuned on withheld training sentences that are manually labeled. The first constraint ensures that at most one VerbNet sense is chosen for each verb. The second one ensures consistency between choices of WordNet senses and corresponding VerbNet ones. The third constraint covers the selectional restrictions described earlier. The fourth constraint ensures that at most one sense is chosen for each non-verb word. We instantiate a separate ILP for every sentence, and thus the ILP size and complexity remain tractable.

4. GRAPH INFERENCE

Based on the output frames of the semantic parsing phase, we derive connections between different activity frames: parent types (hypernyms), semantic similarity edges, and temporal order (previous/next). We cast this as a graph inference problem, denoting the three types of connections as T, S, and P (previous) edges. We tackle this task using the Probabilistic Soft Logic (PSL) framework [6] for relational learning and inference.

For each frame, the activity name is either a single word that is directly mapped to a WordNet/VerbNet sense, or it is a multi-word phrase. In the latter case, we only map the head word of the phrase to WordNet or VerbNet. For an activity a, we denote the mapped part as $h(a)$ and the remaining part as $\bar{h}(a)$ ($\bar{h}(a) = \emptyset$ for single-word activity names).

Edge Priors. We define an activity as a (verb-sense,noun-sense) pair. This allows us to leverage WordNet's taxonomic hierarchy to estimate parent types and similarities between activities.

Our model starts off with prior probabilities for each of the three kinds of edges. The prior for T (parent type) edges between two pairs (v_1, n_1), (v_2, n_2) is calculated as a multiplicative score $\text{t}(v_1, v_2) \cdot \text{t}(n_1, n_2)$. For the noun senses, we use the WordNet hypernymy: The score is 1 if parent and child are connected by hypernymy, and 0 otherwise. For the verb senses, we check both WordNet hypernymy and VerbNet verb hierarchy.

Finally, we derive edges from the subsumption of activity participants, retrieved from WordNet, e.g. between *drinking tea* and *drinking beverage*.

We create S (similarTo) edges based on the similarity between (v_1, n_1), (v_2, n_2) using the multiplicative score: $\text{sim}(v_1, v_2) \cdot \text{sim}(n_1, n_2)$. The taxonomic relatedness score between two noun senses n_1, n_2 is computed using a WordNet path similarity measure [30]. Scores between two verb senses v_1, v_2 are computed using WordNet verb groups and VerbNet class membership [36].

P (previous) edges: Scripts come with scene boundaries. We assume that the activity sequences that occur in a scene are temporally alignable. While an exact sequence of activity does not bring much redundancy, a gap-enabled sequence of activities can have rich statistics. Secondly, generalizing activities to potential parent nodes brings more redundancy, and hence richer statistics. We use a generalized sequence mining algorithm, GSP [38], perfectly suitable to our scenario. We define two parameters: minimum support=3 and maximum gap = 4. We use GSP to efficiently determine P edges. We provide priors to P edges according to the following. An activity a_1 precedes a_2 with probability proportional to the support $\frac{\text{freq}(a_1 \text{ prev } a_2)}{\text{freq}(a_1)\,\text{freq}(a_2)}$.

Inference. Based on the initial prior scores for T, S, P, our task is to compute a cleaner graph of T, S, and P edges with scores reflecting their joint dependencies. These dependencies are captured in our PSL model with the following soft first-order logic rules. Since these are soft rules, they do not need to hold universally. The model automatically determines to what extent they should contribute to the final solution.

1. Parents often inherit prev. (P) edges from their children: $P(a, b) \wedge T(a, a') \wedge T(b, b') \Rightarrow P(a', b')$.
2. Similar activities are likely to share parent types $S(a, b) \wedge T(b, b_0) \Rightarrow T(a, b_0)$.
3. Likely mutual exclusion between edge types: $T(a, b) \wedge S(a, b) \Rightarrow \neg P(a, b)$.
4. Siblings are likely to be similar: $T(a, c) \wedge T(b, c) \Rightarrow S(a, b)$.

5. Similarity is often transitive:
   $S(a,b) \wedge S(b,c) \Rightarrow S(a,c)$.

6. Similarity is normally symmetric:
   $S(a,b) \Rightarrow S(b,a)$.

The inference weights w_i are tuned based on withheld data, using the PSL system's weight learning module.

5. TAXONOMY CONSTRUCTION

Activity Merging. The previous steps of our pipeline yield fairly clean activity frames, but may produce overly specific activities such as "embrace spouse", "hug wife", "hug partner", "caress someone", etc. These are sufficiently similar to be grouped together into a single frame (with slightly generalized semantics). Thus, the relation S from the previous step provides a pruned starting point for activity merging.

Definition 4. *An* activity synset *is a group of activities with highly related semantics. For a synset* $\{(v_1,o_1),(v_2,o_2),\dots\}$ *of verb-sense/object-sense pairs, we require that* $a_i = (v_i,o_i)$ *and* $a_j = (v_j,o_j)$ *have a semantic distance in WordNet below a certain threshold.*

Specifically, we consider *WordNet path similarity* [30] as a measure of semantic distance. To this end, we construct a graph between activity frames based on the synset (i.e., equivalence) and hypernym/hyponym relations in WordNet. The edges in this graph could be weighted by relatedness strength, such as gloss overlap [3], but we simply used uniform weights,i.e. simple path lengths $dist(v_i,v_j)$. For two activities a_i, a_j, we compute

$$\frac{1}{2}\left(\frac{1}{1+\text{dist}(v_i,v_j)} + \frac{1}{1+\text{dist}(o_i,o_j)}\right).$$

In addition, we consider the participants sets P_i, P_j in the frames of a_i,a_j, respectively. Recall that each P_i is a set of WordNet noun-phrase senses. We compute the WordNet path similarity for each element in $P_i \times P_j$, and aggregate them into an overall measure by taking the maximum (or alternatively the average). The final distance between a_i and a_j is the average of the verb-sense/object-sense distance and the participants distance.

The threshold for merging two activities into a synset is determined by manually grouping a small sample of activities and computing the threshold that achieves the synsets in the sample. We transitively merge activities whenever their distance is below that tuned threshold. We perform a transitive closure on this pruned neighborhood to allow grouping of activities.

Hierarchy Induction. The above techniques provide us with a suitably grained but still flat collection of activity synsets. However, some of these may semantically subsume others. For example, *divorce husband* is subsumed by *break up with a partner*. Again, the relation T from the previous step provides a pruned starting point for hierarchy induction.

Definition 5. *An* activity taxonomy *is a DAG (directed acyclic graph) of activity synsets such that* $a_i \sqsubset a_j$ *is an edge in the DAG if* a_i *is semantically subsumbed by* a_j*. That is, the verb or object of* a_j *is more general than that of* a_i*.*

To construct the hierarchy, we again use WordNet path similarity but consider only *hypernym* relations now (i.e., disregard hyponyms). For this asymmetric measure, we again tune a threshold by manually assessing a small sample. The resulting taxonomy graph initially contains all subsumption pairs with semantic distance below the threshold. As this may create cycles, we finally break cycles by greedily removing low-weight edges. In building the Knowlywood KB, we had to eliminate only few cycles.

6. RESULTS

To evaluate our approach, we conducted a series of experiments to thoroughly examine our pipeline for semantic parsing and knowledge distillation, as well as the resultant Knowlywood activity frames.

Data Processing. Knowlywood is constructed by processing 1.89 million scenes from several sources:

- 560 movie scripts, scripts of 290 TV series, and scripts of 179 sitcoms. We crawled this data from Web sites like `wikia.com` and `dailyscript.com`.
- The Novels dataset comprises 103 novels from Project Gutenberg [14].
- Crowdsourcing: We use the data from [34], which consists of textual descriptions of videos portraying humans engaging in cooking related activities.

6.1 System Components

Semantic Parsing. In order to gain deep insights about our system, we had human judges annotate at least 250 random samples of the outputs of the different stages in our semantic parsing method, i.e., sentence extraction by pre-processing datasets, clause level splitting, the basic NLP pipeline (tagging, chunking, etc.), and finally disambiguation and VerbNet-based role assignment. Table 2 presents the resulting precision scores with statistical significance given as Wilson score intervals for $\alpha = 95\%$ [7].

We observe that most of the errors stem from the NLP pipeline, especially chunking. This could be addressed by using more advanced NLP tools, which, however, tend to be slower. Processing the sitcom and TV series data is the most challenging and error-prone due to the nature of these texts: the sentences are long and often filled with slang (e.g., "hold'em"). Some errors are also introduced at the early stage of pre-processing movie scripts, where we rely on regular expressions to parse the semi-structured text files (e.g., the introductory text for each scene which introduces the location, time, etc.).

Graph Inference. Next, we evaluate the PSL-based graph inference. Our findings indicate that it was instrumental in cleaning the candidate relations between activities and also in acquiring new edges between them. Table 3 shows the precision and size of the graph before and after the inference step. For example, from the P edge between `acquire#1; cutting knife#1` and `use#1; cutting knife#1`, a new P edge is derived from `acquire#1; knife#1` to `use#1; knife#1`. The transitive closure on the S relationships adds new edges. Thus, our graph inference increases Knowlywood's coverage and accuracy by inferring missing edges and removing inconsistent ones.

Table 3: Effect of PSL inference

	Before inference		After inference	
	Precision	#Edges	Precision	#Edges
T	0.77±0.04	1,906,520	0.87±0.03	4,511,203
S	0.84±0.02	1,022,700	0.85±0.04	3,421,210
P	0.78±0.04	116,186	0.84±0.09	205,678

Table 2: Evaluation of the semantic parsing

	sentence	clause	NLP	senses	participant roles	overall
Movie scripts	0.79 ±0.11	0.84 ±0.07	0.87 ±0.06	0.96 ±0.04	0.96 ±0.03	0.91 ±0.03
TV series	0.90 ±0.06	0.96 ±0.04	0.40 ±0.10	0.65 ±0.10	0.79 ±0.08	0.74 ±0.04
Sitcoms	0.91 ±0.07	0.93 ±0.06	0.38 ±0.12	0.67 ±0.12	0.72 ±0.11	0.73 ±0.05
Novels	0.94 ±0.05	0.91 ±0.07	0.74 ±0.12	0.85 ±0.09	0.93 ±0.06	0.90 ±0.04
Crowdsourcing	0.96 ±0.04	0.96 ±0.04	0.86 ±0.09	0.75 ±0.11	0.91 ±0.07	0.91 ±0.03

Synset and Hierarchy Construction. We performed a static analysis of the hierarchy as well as an empirical evaluation. There were 543 cycles in the graph. These were of a very small length (average length 3). After breaking the cycles, the DAG consists of 505,788 synset nodes without any cycles. The maximum depth of the graph is 5.

Over a random sample of 119 activity synsets, human judges were asked if the edge between random synset members was indeed a synonymy relation, i.e. semantically equivalent activities. To evaluate the hierarchy, in a similar way, human judges were asked if the edge between two activity synsets was one of hypernymy, i.e. subsuming activity synsets.

The synset grouping achieved a very high accuracy of 0.976±0.02 (Wilson score intervals for $\alpha = 95\%$ [7]). One of the reasons for this high accuracy was the tight threshold for taxonomic similarities. We had empirically chosen a high threshold of 0.40 for the synset similarity.

The hierarchy grouping achieved a high accuracy of 0.911±0.04 (Wilson score intervals for $\alpha = 95\%$). An example error case was *walk with a fly* having a hypernymy link to *travel with a beast*. This is because *animal* and *beast* are synonymous. Such mildly incorrect cases led to a slightly lower precision.

6.2 Knowlywood KB Evaluation

In total, the Knowlywood pipeline produced 964,758 unique activity instances, grouped into 505,788 activity synsets. In addition to the edges mentioned above, we also obtain 581,438 `location`, 71,346 `time`, and 5,196,156 `participant` attribute entries over all activities.

Quality. To evaluate the quality of these activity frames, we compiled a random sample of 119 activities from the KB, each as a full frame with values for all attributes (participants, location, time, previous and next activity, etc.). We relied on expert human annotators to judge each attribute for each of these activities. An entry was marked as correct if it made sense to the annotator as typical knowledge for the activity. The judgement were aggregated separately for each attribute, and we computed the precision as $\frac{c}{c+i}$, where c and i are the counts of correct and incorrect attribute values, respectively. For statistical significance, we again computed Wilson score intervals for $\alpha = 95\%$. The per-attribute results are reported in Table 4. The inter-annotator agreement for three judges in terms of Fleiss' κ is 0.77.

We can observe from these assessments that Knowlywood achieves good precision on most of the attributes. In some datasets like the Crowdsourcing collection, no information on time or location is available. This accounts for the low scores.

Examples. Fig. 4 presents anecdotal examples of Knowlywood's activity frames, with specific sense numbers from WordNet.

Comparison with ConceptNet. There is no direct competitor that provides frames of semantically refined activities. We thus compared Knowlywood with ConceptNet 5 (CN), the closest available

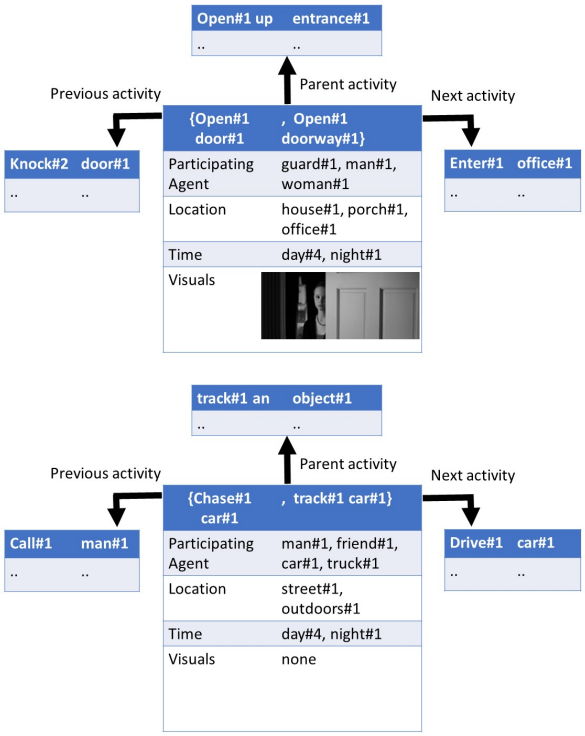

Figure 4: Anecdotal Examples

resource, assuming that any concept name matching the pattern *verb [article] object* is an activity. We mapped CN's relations to our notion of activity attributes as follows:

- IsA, InheritsFrom → type,
- Causes, ReceivesAction, RelatedTo, CapableOf, UsedFor → agent,
- HasPrerequisite, HasFirstSubevent, HasSubevent, HasLastSubevent, MotivatedByGoal → prev/next,
- SimilarTo, Synonym → similarTo,
- AtLocation, LocationOfAction, LocatedNear → location.

The activities derived this way from CN were manually assessed by the same pool of annotators that assessed the Knowlywood frames. We randomly sample 100 activities from CN and take all their relations but adding further relationships if we encountered too few of any one relationship type. The last row of Table 4 shows the results — both coverage and precision. We see that CN works well for eliciting previous/next activities. Here its quality exceeds that of Knowlywood. CN's crowdsourcing-based knowledge acquisition leads to fine-grained temporal knowledge that is rather difficult to mine from narrative texts (e.g., that riding a horse is preceded by keeping your heel down, and followed by your bottom getting sore).

Table 4: Knowlywood coverage and precision

Source	#Input Scripts	#Scenes	#Unique Activities	Parent	Participant	Prev	Next	Loc.	Time	Avg.
Movie scripts	560	148,296	244,789	0.87	0.86	0.78	0.85	0.79	0.79	0.84
TV series	290	886,724	565,394	0.89	0.85	0.81	0.84	0.82	0.84	0.86
Sitcoms	179	286,266	200,550	0.88	0.85	0.81	0.87	0.81	0.83	0.87
Novels	103	383,795	137,365	0.84	0.84	0.78	0.88	0.85	0.72	0.84
Crowdsrc.	25	3,701	9,575	0.82	0.91	0.91	0.87	0.74	0.40	0.86
Knowlywood	1,157	1,708,782	964,758	0.87	0.86	0.84	0.85	0.78	0.84	**0.85±0.01**
ConceptNet 5	-	-	4,757	0.15	0.81	0.92	0.91	0.33	N/A	**0.46±0.02**

However, these high precision values also result from the specific nature of CN's knowledge representation. Since CN's concepts are essentially strings (not word senses), we instructed our annotators to evaluate an attribute value as correct even if it holds true for just one possible interpretation of the concept names, ignoring ambiguity. The data also contains duplicates (e.g. "you open your wallet", "open your wallet", "open wallet", ...) that were all judged as correct as predecessors of "taking out money". CN's less formalized nature is particularly apparent from the fact that the parent type attribute obtains a precision of only 15%. Generally, except for the temporal ordering of activities, the precision of CN is substantially below that of Knowlywood.

Most importantly, Knowlywood's coverage of activities dwarfs that of CN. CN merely provides 4,757 activities, most of which are also included in Knowlywood, while the latter additionally contains nearly a million activity frames.

Comparison with ReVerb. We also compare Knowlywood with ReVerb [12], the most widely used system for broad-coverage open information extraction. Open information extraction aims at mining all possible subject-predicate-object triples from text. We mine activity knowledge from these triples such that the subject is an agent, and the predicate and object together form an activity, e.g. *drink + coffee*.

For role assignment, we mine `MovieClips.com` to obtain mappings from words to labels. MovieClips contains high-quality human-annotated and categorized tags for nearly 30,000 movie scenes (e.g. "action:singing", "prop:violin", "setting:theater"). These tags have a direct correspondence to our attributes (see Table 6). The tag co-occurrence statistics can be used to create a Bayesian classifier as $P(\text{class}|\text{word}) = \frac{P(\text{class},\text{word})}{\sum_{w_i} P(\text{class},w_i)}$, relying on the joint probabilities for classes and words from `MovieClips.com`. One may also consider using semantic role labeling systems as an alternative. However, they cannot solve our semantic parsing task because they require large amounts of domain-specific labeled training data. Moreover, they suffer from poor scalability.

We consider two different datasets as input to ReVerb. First, all the input Script data that we used for our system (setup called ReVerbMCS). Second, all of ClueWeb09 dataset (setup called ReVerbClue). ReVerb extractions over ClueWeb09 are already available in the form of a publicly available dataset [12], consisting of 15 million unique SVO (Subject Verb Object) triples. The ReVerbClue data does not contain enough context to use the MovieClips-based role classifier because it consists of only SVO triples.

Since both ReVerbMCS and ReVerbClue extractions are strings (not word senses), we leniently evaluated an attribute value as correct if it holds true for any possible sense of the concept. This is thus a much easier task than Knowlywood's, for which we required the correct sense disambiguation.

Table 5: ReVerb baselines (counts and precision scores)

	Activities	Participant	Location	Time
ReVerbMCS	0.37M	0.37M, 0.77	0.17M, 0.83	0.05M, 0.66
ReVerbClue	0.86M	1.47M, 0.41	0.055M , -	0.008M, -

Table 6: Mappings between MovieClips.com and Knowlywood

MovieClips tag	Knowlywood attributes	Example
action	activity.v	cut
prop	activity.o	knife
setting	location	bar
occasion	time	thanksgiving
charactertype	participant	policeman

In Table 5, we list the number of activities as well as numbers and precision of several roles. The precision values are obtained by evaluating the frames corresponding to the activities overlapping with the Knowlywood test set of 119 activities resulting in more than 400 attribute triples. Knowlywood outperforms both the ReVerb based baselines (compare to Table 4), in terms of both precision and counts. The role labels score in ReVerbMCS reflect the rich statistics (though limited in size) obtained from the manually curated MovieClips. We also see that extractions from ClueWeb09 data, which is an order of magnitude larger than our scripts data, did not entail better quality.

Multimodal Content. By automatically aligning the movie scripts with subtitled videos, we were also able to associate 27,473 video frames with Knowlywood's activities. We believe that this will be an important asset for computer vision, because existing systems for activity detection in videos suffer from a lack of training data and background knowledge, and hence have been quite limited in their coverage.

7. USE CASES

7.1 Movie Scene Tagging

In order to evaluate the usefulness of the Knowlywood KB extrinsically, we introduce the task of predicting the activity portrayed in a movie clip, without task-specific training data, given only the location and participants in the corresponding scene.

As ground truth, we consider `Movieclips.com`, which contains high quality, manually curated categorized tags for nearly

30,000 movie clips/ scenes. Examples of these include: "location/setting: cemetery", "participating object/prop: rose", "action: obituary speech". By analyzing the co-occurrence statistics over the tags of these clips, we obtain a scored list of activities for a given [participant(s), location(s), time(s)]. We randomly select 1,000 clips from this gold data.

The evaluation task is to assess Knowlywood's (or any baseline activity KB's) top-k activity recommendations given only [participant(s), location(s), time(s)]. This task is more complex than a simple tag recommendation which would ignore any tag categories. As KBs, we use the Knowlywood KB, and the various baselines: ConceptNet, ReverbMCS, and ReverbClue.

Table 7: Movie Scene Tagging evaluation

	MRR	Hit rate
ReVerbClue	0.070	0.180
ConceptNet	0.143	0.345
ReVerbMCS	0.254	0.415
Knowlywood	0.327	0.610

The evaluation is based on a comparison of the predicted top-k activity list with the ranked gold list of activities. We report the standard IR-metric [21] Mean Reciprocal Rank (MRR) that rewards early hits in the predictions. We also report Hit-Rate metric which is one whenever the top-10 results contain atleast one good tag.

We then evaluated the various KBs on the movie scene tagging task. This is an automated evaluation, as the ground truth gold data is already available. For both the KBs and the gold-set, we uniformly set k=10, i.e. we compare the top-10 predictions against the top-10 ground truth rankings. The results in Table 7 demonstrate that although Knowlywood has not been trained or mined from `Movieclips.com` tags at all, the system is able to outperform the baselines by a large margin both on MRR and Hit rate. ReverbMCS outperforms other baselines because the role label classifer in ReverbMCS uses `Movieclips.com` statistics already. Knowlywood also yields a much better coverage in terms of the hit rate.

7.2 Scene Search

For a second extrinsic evaluation, we use Knowlywood to build a search platform over the corpus, which again comprises movie scripts, the Crowdsourcing dataset, TV series, sitcoms, and novels. This search system takes a text query q as input, which is expected to correspond to some activity. Examples of such queries are *animal attacks man*, *kissing during a romantic dinner*. As output, we expect a ranked list of scenes over the indexed corpus.

Approach. We use the textual (not visual) content of the scenes to obtain the score of a scene s for a given query.

Given an activity $a \in K$, where K denotes the Knowlywood knowledge base, let a_p be the set of

participants according to K and $A_\text{p} = \bigcup_{a \in K} a_\text{p}$ be the set of all participants associated

with activities in K. We derived a query-likelihood statistical language model as follows.

The probability that the scene s generates a query q is given by

$$P(q|s_t) = \sum_{a \in K} \sum_{p \in A_\text{p}} P(q|a) \cdot P(a|p) \cdot P(p|s_t)$$

- s_t is the textual representation of the scene,
- $P(p|s_t)$ is the probability that the scene generates participant p of an activity (e.g., girl, ring, etc.), estimated from noun-phrase occurrences in t with corpus smoothing,

Table 8: Query Frames.

Frame	Semantic restriction
S	WN physical entity
V	WN verb (compulsory)
O_1	WN physical entity
O_2	WN physical entity
L	WN location or WN physical entity
T	WN time-period

Table 9: Performance of the two search methods.

Algorithm	NDCG	MAP	Precision@5	MRR
Knowlywood	0.8972	0.9512	0.8809	0.9840
Text retrieval	0.0772	0.0696	0.0404	0.0730

- $P(a|p)$ is the probability that participant p generates activity a, again with smoothing, and
- $P(q|a)$ is the query likelihood of activity a, estimated by the occurrences of the verb-object words of a in the query, once more with smoothing.

Experimental Setup. As there is no similar activity search system or evaluation dataset, we construct a benchmark dataset by gathering 100 queries of a predefined frame (`S V O1 O2 Location Time`), such as, *man kissed the girl on the cheek at the movie theater in the evening*. For this, we relied on a user interface as in Table 8, asking two people (one outsider and one of the authors) to enter arbitrary queries of their choice, as long as it fit the template. Further examples of these gathered queries include *frying onion* and *killing a bird*.

For this set of 100 queries, we generate search results using our generative model over the Movie script, Crowdsourcing, Sitcom, TV series, and Novels datasets.

For comparison, we also obtain the search results using a text-retrieval baseline, in particular, a statistical language model with Dirichlet smoothing, as implemented in the well-known INDRI system [39].

Two annotators evaluated the top-10 results for each of these queries both for the baseline and the Knowlywood search system. Each result was scored between 1 (irrelevant) to 5 (perfectly relevant). The final rating for each result is given by the average of the ratings by the two annotators.

The annotation ratings were then used to compute four widely-used IR evaluation metrics, namely NDCG, Precision@k, MAP, and MRR [21].

Scene Search Results. Table 9 gives a comparative analysis of the NDCG, MAP, Precision@5, and MRR scores for both search methods. Since MAP, Precision@5, and MRR involve binary notions of relevance, we assume that those scenes that are rated with a score of at least 3 are the only relevant scenes.

We observe that for all four metrics, the Knowlywood search method performs best. We observed that the text retrieval engine often returns scenes with script text that closely matches the words in the query, while Knowlywood achieves a higher level of abstraction. For example, given the query *man climbs mountain*, the text engine favors scenes with many occurrences of the keywords *mountain* and *climb*, but not used in the specific sense of climbing mountains. The Knowlywood search method, on the other hand, uncovers those scenes that portray the activity, even if they do not contain the word *mountain* explicitly, but just semantically related expressions such as *hiking up a hill* etc. The Knowlywood search also

Table 10: Anecdotal examples for Scene Search results

Query	Knowlywood based scheme	Text based retrieval scheme
man climbs mountain	.. following Jack , and helps him **climb a mountain** and find a crystal that will transport Jack home both... from TV series: **Samurai Jack**	.. from deep in the **mountains** ... had entered the **Mountain** ...carried deeper into the **mountain** . .... **climbs** out of the river... looking for the **Mountain** of Skulls as well...turns his attention to Lysinka and the others down the **mountain** .. from Novel: **Conan**
man shoots video	.. While **shooting Dixons music video** , Silver gets a call from the fertility clinic informing her that the IVF procedure has been moved up to the next day .. from Beverly Hills, 90210	.. the woman **shoots Alex with a video game gun** as the woman traps her in the game that the.. from TV series: **Totally Spies**
kill a bird	..mark go hunting with sophie 's dad. jeremy go hunting with sophie 's dad. mark tries to **kill a bird** . the man injures it simply. the man tries to break its neck.. from Sitcom: Peep Show	.. Carlos and Susan are still painting over the graffiti on the wall as those people discuss **To Kill a Mocking Bird** , however , while talking , .. from TV series: Desperate Housewives

correctly identifies the true meaning of an activity even if it contains verbs with ambiguous meaning. For example, the query *shoot a video* is often interpreted wrongly by the text retrieval engine and therefore it returns irrelevant snippets referring to shooting with a gun, etc. Table 10 provides some anecdotal examples of queries and scene search results by the two competitors.

8. RELATED WORK

Large-scale knowledge graphs have become a major trend both in academia (DBpedia, NELL, Yago, etc.) and industry (Google, Microsoft, etc.). However, these are focused on facts about individual entities, rather than commonsense. Commonsense KBs like ConceptNet [17], VerbOcean [10], or WebChild [41], on the other hand, focus on simpler relationships between concepts such as *partOf*, *usedFor*, *hasColor*, *hasShape*, etc.

Formal upper-level ontologies such as Cyc [22] and SUMO [26] contain some activity knowledge like agents involved in concepts expressed by verbs. For example, SUMO knows that kissing involves two humans as agents and their lips. However, this is manually modeled and the amount of activity knowledge in these ontologies is tiny compared to what Knowlywood captures. Also, these ontologies focus on knowledge that is expressible in first-order logics, and lack commonsense knowledge offered by Knowlywood such as typical locations, times, prev./next chains, and participants.

Interest in human activities goes back to Schank and Abelson's early work on scripts [35], where procedural knowledge was gathered manually. More recently, such knowledge has been crowdsourced via Amazon Mechanical Turk [32], but this data only covers 22 stereotypical scenarios. Other research has developed ways to mine activity knowledge from the Web using text analysis [8] and deep neural networks [23]. These methods aim at solving small temporal ordering tasks, which is different from our goal of producing a large KB. [37] analyze a collection of event similarity data, but do not construct any new activities.

Regarding multimodal data, [9] attempt to mine a large-scale collection of simple conceptual knowledge from images, e.g. that wheels are parts of cars. However, this work does not recognize activities. [33] relate crowdsourced activity descriptions to videos. However, this is a very small collection of only 26 activity types. Activities play an important role in different domains. [42] present a KB of object affordances for robotics, but cover only 250 objects.

Semantic parsing has received much attention in computational linguistics recently; see [1] and references there. So far it has been applied only to specific use-cases like natural-language question answering [4, 13] or understanding temporal expressions [19]. We believe that our work is the first to apply semantic parsing to large amounts of narrative text for KB construction. Our methodology uses techniques like ILP and graphical models that have been previously used in natural language analysis. However, applying them to the huge input in our setting requires judicious design decisions that are not straightforward at all.

Semantic role labeling (SRL) [16, 28] is highly related to semantic parsing, the goal being to fill the slots of a pre-defined frame with arguments from a sentence. However, state-of-the-art methods are slow and do not work well for our task of activity knowledge acquisition. Moreover, SRL methods typically consider Propbank [27] as a backbone, and Propbank lacks the semantic organization of verbs that VerbNet provides.

Word sense disambiguation (WSD) [25] is another component in semantic parsing and semantic role labeling. We use the state-of-the-art tool IMS (It-Makes-Sense) [43] as a WSD prior for our joint disambiguation ILP. Note, though, that WSD alone focuses on the lexical semantics of individual words – this is still far from full-fledged semantic parsing for populating an activity KB.

Taxonomy induction has a rich body of prior work in NLP and AI. However, this is primarily for hypernymy (isa, subclasses) between general concepts (classes, noun senses) (see, e.g., [31, 29] and references given there). There is little research on taxonomy induction for verbal phrases [20, 10, 24]. But this line of work does not consider rich attributes for actions, and is about general verbs rather than focused on human activities.

9. CONCLUSION

We have presented Knowlywood, the first comprehensive KB of human activities. It provides a semantically refined hierarchy of activity types, participating agents, spatio-temporal information, information about activity sequences, as well as links to visual contents. Our algorithms ensure that the entries are fully disambiguated and that inconsistent attributes are removed. Our experiments show that Knowlywood compares favorably to several baselines, including in use cases such as tag recommendations. We believe that the resulting collection of around one million activity frames is an important asset for a variety of applications such as image and video understanding.

10. REFERENCES

[1] Yoav Artzi, Nicholas FitzGerald, and Luke S Zettlemoyer. Semantic parsing with combinatory categorial grammars. In *ACL*, 2013.

[2] S. Auer, C. Bizer, J. Lehmann, G. Kobilarov, R. Cyganiak, and Z. Ives. DBpedia: A nucleus for a web of open data. In *In ISWC/ASWC*, 2007.

[3] Satanjeev Banerjee and Ted Pedersen. Extended gloss overlaps as a measure of semantic relatedness. In *IJCAI*, 2003.

[4] Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic parsing on freebase from question-answer pairs. In *EMNLP*, 2013.

[5] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: A collaboratively created graph database for structuring human knowledge. In *SIGMOD*, 2008.

[6] Matthias Brocheler, Lilyana Mihalkova, and Lise Getoor. Probabilistic similarity logic. *arXiv 1203.3469*, 2012.

[7] Lawrence D. Brown, T. Tony Cai, and Anirban DasGupta. Interval estimation for a binomial proportion. *Statistical Science*, 16(2):101–133, 2001.

[8] Nathanael Chambers and Dan Jurafsky. Unsupervised learning of narrative schemas and their participants. In *ACL*, 2009.

[9] Xinlei Chen, Abhinav Shrivastava, and Abhinav Gupta. Neil: Extracting visual knowledge from web data. In *ICCV*, 2013.

[10] Timothy Chklovski and Patrick Pantel. VerbOcean: Mining the web for fine-grained semantic verb relations. In *EMNLP*, 2004.

[11] Luciano Del Corro and Rainer Gemulla. ClausIE: Clause-based open information extraction. 2013.

[12] Anthony Fader, Stephen Soderland, and Oren Etzioni. Identifying relations for open information extraction. In *Proc. EMNLP*, 2011.

[13] Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. Open question answering over curated and extracted knowledge bases. In *SIGKDD*, 2014.

[14] Manaal Faruqui and Sebastian Pado. Towards a model of formal and informal address in english. In *EACL*, April 2012.

[15] Christiane Fellbaum, editor. *WordNet: An Electronic Lexical Database*. The MIT Press, 1998.

[16] Daniel Gildea and Daniel Jurafsky. Automatic labeling of semantic roles. *Comput. Linguist.*, 28(3):245–288, 2002.

[17] Catherine Havasi, Robert Speer, and Jason Alonso. ConceptNet 3: a flexible, multilingual semantic network for common sense knowledge. In *RANLP*, 2007.

[18] Karen Kipper, Anna Korhonen, Neville Ryant, and Martha Palmer. Extending VerbNet with novel verb classes. In *LREC*, 2006.

[19] Kenton Lee, Yoav Artzi, Jesse Dodge, and Luke Zettlemoyer. Context-dependent semantic parsing for time expressions. In *ACL*, 2014.

[20] Dekang Lin and Patrick Pantel. Dirt@ sbt@ discovery of inference rules from text. In *KDD*, 2001.

[21] Christopher D Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to information retrieval*, volume 1. Cambridge University Press, 2008.

[22] C. Matuszek, M. Witbrock, R.C. Kahlert, J. Cabral, D. Schneider, P. Shah, and D. Lenat. Searching for common sense: Populating Cyc from the Web. In *AAAI*, 2005.

[23] Ashutosh Modi and Ivan Titov. Inducing neural models of script knowledge. In *CoNLL*, Baltimore, MD, USA, 2014.

[24] Ndapandula Nakashole, Gerhard Weikum, and Fabian Suchanek. PATTY: A taxonomy of relational patterns with semantic types. In *EMNLP/CONLL*, 2012.

[25] Roberto Navigli. Word Sense Disambiguation: a survey. *ACM Computing Surveys*, 41(2):1–69, 2009.

[26] Ian Niles and Adam Pease. Towards a standard upper ontology. In *FOIS*, 2001.

[27] Martha Palmer, Daniel Gildea, and Paul Kingsbury. The Proposition Bank: An annotated corpus of semantic roles. *Computational linguistics*, 31(1):71–106, 2005.

[28] Martha Palmer, Daniel Gildea, and Nianwen Xue. Semantic role labeling. *Synthesis Lectures on Human Language Technologies*, 3(1):1–103, 2010.

[29] Marius Pasca. Acquisition of open-domain classes via intersective semantics. In *Proc. WWW*, pages 551–562, 2014.

[30] Ted Pedersen, Siddharth Patwardhan, and Jason Michelizzi. WordNet::Similarity: Measuring the relatedness of concepts. In *NAACL*, 2004.

[31] Simone Paolo Ponzetto and Michael Strube. Taxonomy induction based on a collaboratively built knowledge repository. *Artificial Intelligence*, 175(9-10), 2011.

[32] Michaela Regneri, Alexander Koller, and Manfred Pinkal. Learning script knowledge with web experiments. In *ACL*, 2010.

[33] Michaela Regneri, Marcus Rohrbach, Dominikus Wetzel, Stefan Thater, Bernt Schiele, and Manfred Pinkal. Grounding action descriptions in videos. *TACL*, 1:25–36, 2013.

[34] Marcus Rohrbach, Sikandar Amin, Mykhaylo Andriluka, and Bernt Schiele. A database for fine grained activity detection of cooking activities. In *CVPR*, 2012.

[35] R. Schank and R. Abelson. *Scripts, plans, goals and understanding: An inquiry into human knowledge structures*. Hillsdale, NJ., 1977.

[36] Karin Kipper Schuler, Anna Korhonen, and Susan Windisch Brown. VerbNet overview, extensions, mappings and applications. In *HLT-NAACL*, 2009.

[37] Tomohide Shibata and Sadao Kurohashi. Acquiring strongly-related events using predicate-argument co-occurring statistics and case frames. In *IJCNLP*, 2011.

[38] Ramakrishnan Srikant and Rakesh Agrawal. *Mining sequential patterns: Generalizations and performance improvements*. Springer, 1996.

[39] Trevor Strohman, Donald Metzler, Howard Turtle, and W Bruce Croft. Indri: A language model-based search engine for complex queries. In *International Conference on Intelligent Analysis*, 2005.

[40] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: A Core of Semantic Knowledge. In *WWW*, 2007.

[41] Niket Tandon, Gerard de Melo, Fabian Suchanek, and Gerhard Weikum. Webchild: Harvesting and organizing commonsense knowledge from the web. In *WSDM*, 2014.

[42] Karthik Mahesh Varadarajan and Markus Vincze. Afnet: The affordance network. In *ACCV (1)*. Springer, 2012.

[43] Zhi Zhong and Hwee Tou Ng. It makes sense: A wide-coverage word sense disambiguation system for free text. In *ACL*, 2010.