Frame Instance Extraction and Clustering for Default Knowledge Building

Avijit Shah[1], Valerio Basile[2], Elena Cabrio[2], and Sowmya Kamath S.[1]

[1] Department of Information Technology,
National Institute of Technology Karnataka (NITK), Surathkal, India
avijit.shah09@gmail.com, sowmyakamath@nitk.edu.in
[2] Université Côte d'Azur, Inria, CNRS, I3S, France
valerio.basile@inria.fr, elena.cabrio@unice.fr

Abstract. Obtaining and representing common-sense knowledge, useful in a robotics scenario for planning and making inference about the robots' surroundings, is a challenging problem, because such knowledge is typically found in unstructured repositories such as text corpora or small handmade resources. The work described in this paper presents a methodology for automatically creating a default knowledge base about real-world objects for the robotics domain. The proposed method relies on clustering frame instances extracted from natural language text as a way of distilling default knowledge. We collect and parse a natural language corpus using the Web as a source, then perform an agglomerative clustering of frame instances according to an appropriately defined similarity measure, and finally extract prototypical frame instances from each cluster and publish them in LOD-complaint format to promote reuse and interoperability.

1 Introduction

Smart machines like robots are becoming ubiquitous in industrial and urban scenarios, assisting humans in their day to day activities. A critical requirement for such intelligent machines is the ability to learn from their environment as humans do, specifically in handling certain common sense tasks that can significantly improve productivity. For instance, when a household robot is instructed to fetch a knife, the fundamental requirement would be an understanding of what a knife is, what it is used for, its location in the house, and so on. Considering the task of instructing the robot to bring something to eat, the robot must first discern that available items like apple, rice, bread and egg are food, then, associate them with the act of eating, and finally infer other relevant information (such as other objects involved in the situation, e.g., cutlery and bowls). We call this type of information as *default* knowledge, a kind of task-oriented background knowledge that allows the robot to perform its tasks when more local/specific knowledge is not available. We call such kind of knowledge "default knowledge" rather than, e.g., background knowledge or simply common sense, to put the emphasis on the robot actions rather than reasoning and inference.

Since the process of manual creation of such common-sense knowledge repositories is highly labor- and cost-intensive, alternative methods for capturing it automatically are critical. In this respect, the Web can be a good candidate as a source of default knowledge as it enables access to a large volume of information about any topic. However, most large-scale resources of structured knowledge on the Web are concerned with named entities like persons and places, while objects and other generic concepts are less represented. A great amount of information about generic concepts is found on the Web in the form of natural language meant for humans to consume. The challenge then is to deal with verbose and ambiguous natural language content in order to gather default knowledge and in designing systems that extract facts and represent them in a concise and machine understandable format.

Our ongoing work is an effort to address some of the shortcomings of the existing general knowledge resources (see Section 2) with the aim of enabling robots with the ability for autonomous default knowledge learning about previously unseen objects. The work described in this paper presents one of the methodologies we are currently developing to create a default knowledge base about real-world objects for the robotics domain. In particular, this methodology relies on clustering frame instances extracted from natural language text as a way of distilling default knowledge and encode it in LOD-complaint format to promote reuse and interoperability.

The paper is organized as follows: in Section 3 we discuss the detail of the proposed methodology for knowledge extraction from natural language text; in Section 4, we report on our findings after examining the collected default knowledge; and in Section 5, we draw conclusions and lay out a path for future directions of research.

2 Related Work

Several researchers have tried to address this challenge in contexts related to the Semantic Web, knowledge management and machine learning. In the linked data ecosystem, DBpedia[3] is perhaps the most well-known resource and one of the most connected to other resources. This large-scale dataset is automatically extracted from Wikipedia and often acts as a hub between different LOD (Linked Open Data) resources. YAGO [15] provides a mapping between DBpedia the lexical resource WordNet [11], which provides additional semantic information, notably a hierarchical structure of concepts based on the hypernym relation. ConceptNet [14] contains approximately 28 million tuples automatically derived from Wiktionary[4] and a number of other resources. It is structured as a multigraph (i.e., a graph with multiple edges connecting the same pairs of nodes), thus it cannot be represented using RDF. The overlap of ConceptNet with DBpedia is low, especially with respect to general objects, preventing its use as a linked data resource for general knowledge. Another noteworthy project in

[3] `http://dbpedia.org`
[4] `http://www.wiktionary.org/`

this context is NELL (Never-Ending Language Learning), an ongoing effort to "read the Web" [5], using a continuously-refined process of knowledge extraction from text. NELL contains more than 50 million "candidate beliefs", i.e., facts with varying degrees of confidence, more than 3 millions of which held with high confidence. However, from the LOD perspective, approaches like NELL and ConceptNet have some limitations in terms of linking towards Web resources, that is, in both cases, terms are generic and potentially ambiguous strings rather than URIs. Moreover, some predicates found in NELL are difficult to use for the purpose of a general knowledge base. This is the case for predicates such as "found in X" where 'X' is a location, that could be better expressed as a relation between a concept and its location. We also found that most predicates are not defined on general entities (classes, ontologically speaking).

An unsupervised approach for natural language understanding called *machine reading* was developed by Etzioni et al [7]. It subsumes techniques like Information Extraction and Question Answering and similar multiple Textual Entailment steps that form a set of beliefs based on the text, resulting in a tool called KnowItAll [8]. TextRunner [18] works on the concepts of machine reading for collecting all extracted triples into an extraction graph. In [2] a technique for building a knowledge base of object-location pairs was proposed. Such knowledge is stored in the form of triples containing the object type, its typical location and common semantic frames associated with it.

Ontologies have also played a major role in collecting and organizing commonsense knowledge, whether open domain or domain specific. Among these, DOLCE [4] and OpenCyC [10] are two popular upper level ontologies that define taxonomic relations between concepts and relations, rules and constraints. Both DOLCE and OpenCyC are linked data. In the AI/robotics domain, KnowRob [16] is a gold-standard reference ontology and reasoning framework for knowledge about domestic environments, including sensors, actuators and other specific aspects of robotic applications. Ontologies can be relatively small compared to large-scale efforts where the modus operandi is to extract knowledge automatically, but the quality of ontological data is often severely affected due to their handcrafted design and building process.

3 Default Knowledge Extraction from Natural Language

The methodology proposed for building the repository of default knowledge is based on the analysis of natural language and the use of statistical methods to extract relevant knowledge while filtering out noise and less informative items. The process comprises a number of sub-phases, as listed below:

1. collecting a corpus of natural language text;
2. parsing the text and extracting frame instances expressed in the natural language;
3. clustering frame instances using an appropriately defined similarity measure;
4. extracting prototypical frame instances from the clusters.

Here, a frame instance is a structure composed of a frame type, i.e., the type of the situation described in the text, and a set of frame elements, i.e., the entities involved in the frame instance. After the last step, the resulting frame instances can be either prototypical members of their original cluster or totally new instances, which represent the default knowledge extracted from the natural language corpus. We describe each of these steps in detail in the following section.

3.1 Source Corpus

To collect information relevant to the task of default knowledge building, it is important that the source text contains a good amount of default knowledge in the first place. Despite the availability of large-scale corpora of written English, this prerequisite is not trivial, given that most existing resources are centered around encyclopedic text (e.g., Wikipedia) or newswire material (e.g., Wall Street Journal corpus [6]). In other words, most of the text available on the Web or otherwise is about entities like people, events, and places, rather than common objects. Following this observation, we created a new corpus by crawling websites containing basic content, specifically meant for learners of the English language, for this study. The intuition behind this was that, such text is more likely to contain explicit mentions to common objects and their properties.

For this purpose, we manually analyzed various websites that carry different stories. We identified ESL YES[5] and the University of Victoria's UVCS[6] as the best fit to our requirement. To crawl the stories from these websites, we implemented a web crawler using the Goose Python package[7] for scraping the Web content and the Lxml library[8] for extracting the text from the pages. The text of these stories is stored in JSON structures along with its associated metadata, namely, story title, author, domain, paragraph count, line count, and word count. In Table 1, we summarize some statistics of the corpus generated after crawling these two sources on the Web.

Table 1. Statistics of the corpus.

Total # stories	1,653
Total # sentences	34,384
Total # tokens	282,664
Average sentence length (# words)	8

[5] ESL Yes 1,600 Free ESL Short Stories, Exercises, Audio, `http://www.eslyes.com/`
[6] English Language Centre Study Zone, `http://web2.uvcs.uvic.ca/elc/studyzone/`
[7] HTML Content/Article Extractor, `https://github.com/grangier/python-goose`
[8] XML Processing Lib in Python, `http://lxml.de/`

3.2 Frame Instance Extraction

The objective of this phase is to parse the corpus and extract a set of frame instances, which are in RDF format. Towards this aim, we employed and adapted KNEWS [3], a tool that relies on logical and lexical semantics to extract knowledge from natural language. KNEWS is an NLP pipeline comprising of modules for semantic parsing, word sense disambiguation and entity linking that run separately on target text. The output of the modules is aligned and mapped to several resources to generate RDF triples describing the instances of frames, according to the framework of *frame semantics* [9], and specifically its implementation in FrameNet [1]. The frame instances are represented in RDF using the FrameBase schema [13].

An example of output generated by KNEWS from the sentence "The robot is driving a car." is shown in Figure 1. The frame evoked by the natural language is recognized to be of type `Operate_vehicle`, the `Driver` and the `Vehicle` roles are filled by the concepts identified respectively by the Wordnet synsets `02764397-n` (automaton, golem, robot, "a mechanism that can move automatically") and `02961779-n` (auto, automobile, car, machine, motorcar ,"a motor vehicle with four wheels; usually propelled by an internal combustion engine").

```
<http://framebase.org/ns/fi-Operate_vehicle_031fa5ad>
  <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
  <http://framebase.org/ns/frame-Operate_vehicle-drive.v> .
<http://framebase.org/ns/fi-Operate_vehicle_031fa5ad>
  <http://framebase.org/ns/fe-Driver>
  <http://wordnet-rdf.princeton.edu/wn31/02764397-n> .
<http://framebase.org/ns/fi-Operate_vehicle_031fa5ad>
  <http://framebase.org/ns/fe-Vehicle>
  <http://wordnet-rdf.princeton.edu/wn31/02961779-n> .
```

Fig. 1. Output of KNEWS

We parsed the corpus (as described in Section 3.1) and extracted 114,536 frame instances comprising of 154,422 frame elements. We counted 686 distinct frame types, 222 roles filled by 3,398 distinct types of concepts. Table 2 shows the most frequently extracted frame types and roles and their relative frequency. KNEWS was unable to select a WordNet-FrameNet mapping for 39,622 frame instances (29.9% of the total number), thus we marked such frame types as `Unmapped` and retained only the original WordNet synsets associated with the event that triggered the frame extraction (typically the main verb of a sentence). Similarly, *vn-* (from VerbNet) indicates that KNEWS could not map a thematic role to the corresponding FrameNet role.

3.3 Measuring Frame Instance Similarity

Once a large collection of frame instances was generated, a study of how structured the set was, was undertaken, starting with an analysis of the relationships

Table 2. Most frequent frame types and roles in the frame instance collection.

Frame type	Frequency	Role	Frequency
Becoming	9,932 (8.6%)	vn-Theme	36,597 (23.6%)
Cause	6,878 (6.0%)	vn-Agent	28,316 (18.3%)
Causation	6,546 (5.7%)	vn-Patient	10,580 (6.8%)
Communication	5,628 (4.9%)	vn-Topic	6,513 (4.2%)
Statement	5,462 (4.7%)	vn-Cause	6,049 (3.9%)
Text	4,896 (4.2%)	Speaker	5,408 (3.5%)
Being	4,866 (4.2%)	vn-Experiencer	5,231 (3.3%)
Spelling	4,361 (3.8%)	Cognizer	3,671 (2.3%)
Have	4,261 (3.7%)	Theme	3,460 (2.2%)
Prevarication	4,205 (3.6%)	Agent	2,588 (1.6%)

between the frame instances. With this objective, we defined and implemented a method to measure the similarity between two frame instances. Note that methods exist in literature to compute similarity between frames, surveyed in [12]. However, here we need to measure the similarity between frame *instances*, that is, frames equipped with actual fillers for their roles.

A frame instance fi_i, as extracted by KNEWS, has two components: a frame type ft_i and a list of frame elements $fe_i = \{fe_i^1, ..., fe_i^k\}$. Accordingly, the similarity between two frame instances fi_1 and fi_2 is a linear combination of the similarity between the two frame types and the distance between the frame elements contained in the frame instance:

$$sim(fi_1, fi_2) = \alpha sim_{ft}(fi_1, fi_2) + (1 - \alpha) sim_{fe}(fi_1, fi_2) \quad (1)$$

The similarity $sim(fi_1, fi_2)$ is defined to be a number in the range $[0, 1]$, while the α parameter controls the extent to which the similarity is weighted towards the frame types or the frame elements.

The frame type and frame elements given by KNEWS are generated after the word sense disambiguation step, thus they are always linked to WordNet synsets, although, they are mapped to FrameNet during later processing. This means that, we can directly apply a WordNet-based similarity metric such as Wu-Palmer [17] to asses the similarity between the frame types. The Wu-Palmer similarity between synsets is defined as the length of the shortest path between the two synsets in the WordNet taxonomy, weighted by the depth of the synsets in the tree. According to such a measure when applied to frame instances, for instance, the frame type `Eating` is more similar to `Drinking` than to `Driving`, because of the similarity between their underlying WordNet synsets. Thus, to calculate this first half of the equation, we take the similarity between two frame types as the Wu-Palmer similarity between their corresponding synsets:

$$sim_{ft}(fi_1, fi_2) = wup(ft_i, ft_j) \quad (2)$$

The second half of equation 1 deals with the similarity computation between two sets of frame elements. We employ again the WordNet-based synset similarity measure, but this time an extra step of aggregation is needed. For each synset corresponding to the frame elements $fe_i \in fi_1$, we compute all the similarity scores of synsets corresponding to the frame elements $fe_j \in fi_2$, and select

the best match. The aggregation by maximum is an approximation of the best match algorithm on bipartite graphs. The resulting similarities are averaged over all the frame elements. Since this process is asymmetrical, we compute it in both directions and take the average of the results:

$$sim_{fe}(fi_1, fi_2) = \frac{1}{2}\Big(\frac{1}{|fi_1|}\sum_{fe_i \in fi_1} \max_{fe_j \in fi_2} wup(fe_i, fe_j) + \\ + \frac{1}{|fi_2|}\sum_{fe_i \in fi_2} \max_{fe_j \in fi_1} wup(fe_i, fe_j)\Big) \tag{3}$$

Note that at this stage, the specific roles of the elements are ignored during the process of calculation of similarity between frame elements. Such additional factors could be taken into account by setting the similarity between two frame elements to 0 if their roles are different.

3.4 Clustering Frame Instances

With a collection of frame instances in place, and a way of measuring the pairwise similarity between frame instances, we can now proceed to the next step in our process of extracting default knowledge from natural language, namely, clustering the frame instances. For this, we perform hierarchical clustering to group the generated frame instances into hard clusters, that is, each frame instance is initially assigned to exactly one cluster. The underlying hypothesis is that clustering frame instances will allow us to extract a number of instances that can be considered default knowledge. For instance, if we observe several instances of a `Drinking` frame involving `Water` but only one case of `Drinking` linked to `Gasoline`, we can confidently say that water is something that can be drunk while gasoline is unlikely to be. We expect these phenomena to surface when clustering all the collected frame instances, for instance, finding that a `Gasoline`-involving frame instance is further away from the centroid of a `Drinking` cluster than a `Water`-involving frame instance.

We produced a hierarchical hard clustering of the frame instances using the complete-linkage agglomerative method implemented in the SciPy[9] Python library. The input to the clustering algorithm is a distance matrix where $distance = (1 - similarity)$ and similarity is calculated as described in Section 3.3. The output of the clustering is a dendrogram, a tree-like structure where each cluster is a node and links between nodes are based on the similarity between clusters.

We experimented with different clustering configurations, and empirically determined thresholds to cut the dendrograms and produce clusters, favoring values that induced a small number of clusters. With respect to the α parameter, we decided to study the behavior of the clustering process for the two sides of equation 1 separately. That is, we performed the clustering with $\alpha = 0$ and $\alpha = 1$, leaving the evaluation of intermediate values for future work. Finally, for this study, we choose to ignore the semantic roles, effectively considering the

[9] Available online: `https://www.scipy.org/`

frame elements as bags of concepts. Table 3 shows two examples of clusters of frame instances, one for each value of α under consideration.

Table 3. Two examples of frame instance clusters. The frequency of frame types and frame elements are reported in parenthesis. Frame types and elements with low frequency (1) have been removed for readability.

Similarity metric	Frame types	Frame elements
Based on frame types ($\alpha = 1$)	Commerce_buy (75)	*Goods* thing-n#8-n (11)
		Goods star+sign-n#1-n (4)
		Goods ticket-n#1-n (3)
		Goods book-n#1-n (2)
		Goods clothes-n#1-n (2)
		Goods placard-n#1-n (2)
		Goods cycle-n#6-n (2)
		Buyer thing-n#8-n (2)
		Goods machine-n#6-n (2)
		Goods shirt-n#1-n (2)
		Goods filter-n#2-n (2)
		Goods pellet-n#2-n (2)
		Buyer male-n#2-n (2)
Based on frame elements ($\alpha = 0$)	Stimulus_focus (8)	*vn-Theme* book-n#1-n (24)
	Categorization (4)	*Item* book-n#1-n (4)
	Hear (4)	*vn-Patient* book-n#1-n (2)
	Reading (4)	
	Reading_aloud (4)	

As a final step, we perform a straightforward aggregation to extract RDF triples from the clusters. From each cluster, we select the most frequent frame type and the most frequent frame element along with its role. Such information is represented by an RDF triple (not reified) of the form `frame_type, role, frame_element`, such as, for instance, the following:

```
<http://framebase.org/ns/frame-Ride_vehicle>
  <http://framebase.org/ns/fe-Vehicle>
  <http://wordnet-rdf.princeton.edu/wn31/02837983-n>
```

We perform this step for the two clustering methods determined by the α parameter, obtaining about 300 triples each. The datasets are published on the Web at the page `http://project.inria.fr/aloof/data/`.

4 Discussion

From a qualitative viewpoint, we observed that the clustering based on similarity between frame types produces clusters with one or few similar frame types with many different frame elements. This is in line with the intuition behind the similarity measure defined in Section 3.3. In other words, such clusters answer the question "what kind of entities typically occur in similar situations?". For instance, one of the clusters we produced contains frame instances of type `Bringing` and `Operate_vehicle` and frame elements like `machine-n#6-n` (Vehicle), `tractor-n#1-n` (Vehicle), `individual-n#1-n` (Driver) and `location-n#1-n`

(Goal). However, the limited size of the corpus results in a high degree of sparseness with respect to the topics, therefore it becomes challenging to separate informative items from what we could consider noise – in the example above, the cluster also contains elements such as `thing-n#8-n` (for Vehicle). This shows that our strategy of filtering out any low frequency elements is not powerful enough, and needs to be further improved.

Conversely, from a clustering based on the similarity between frame elements, relations emerge between frame types that typically involve the same type of entities. These kind of clusters can help answering questions of the type "in what situations (and with what role) are certain entities usually found?". Following up from the previous example, one of the clusters extracted with this method contains entities such as `machine-n#6-n` and `bike-n#1-n`, and the frame types include `Bringing`, `Operate_vehicle`, `Commerce_buy`, `Setting_out`, `Ride_vehicle`, and `Carry_goods`. Unfortunately, the same caveat about noise applies here too, that is, a certain level of noise is always present (in this example, the cluster also contains frame types `Cause_change`, `Body_movement`, `Causation`, and `Becoming_aware`).

Filtering out frame types and elements from the clusters based on their frequency, as we do to produce the final collection of default knowledge in RDF triples, helps to reduce the noise and cut down uninformative items. However, frequency-based filtering may be too blunt an edge, potentially resulting in discarding important items, unless the amount of source material to parse is enough to ensure that relevant knowledge stands out.

5 Conclusion and Future Work

In this paper, we presented a method to automatically extract default knowledge from natural language text in the form of prototypical frame instances and represent it using RDF triples. Such knowledge can be used by a robot for planning and making inferences about its surroundings. Our method is based on parsing natural language with a knowledge extraction software to extract a large set of frame instances. Then, these instances are clustered together according to the type of their frames and the type of their frame elements. Finally, we extract new default knowledge from the clusters and publish the resulting RDF dataset.

From here, this study can follow a number of possible directions. Firstly, the corpus we collected from ESL material is somewhat limited both in size and in terms of the covered topics. This is reflected in the entities found as frame elements in the clusters, and could be alleviated by building a larger corpus, perhaps focusing on specific types of entities, e.g., domestic objects. Secondly, while we experimented with some of the parameters, the literature on clustering is very vast and we just scratched the surface of the many methods available. In future work, we intend to try alternative approaches for clustering frame instances and extract new triples from the clusters, and also to evaluate the output extensively using standard cluster-purity metrics.

References

1. Baker, C.F., Fillmore, C.J., Lowe, J.B.: The berkeley framenet project. In: 17th International Conference on Computational Linguistics. pp. 86–90. ACL '98 (1998)
2. Basile, V., Cabrio, E., Gandon, F.: Building a General Knowledge Base of Physical Objects for Robots. In: 13th International Conference, ESWC 2016 (May 2016)
3. Basile, V., Cabrio, E., Schon, C.: KNEWS: Using Logical and Lexical Semantics to Extract Knowledge from Natural Language. In: European Conference on Artificial Intelligence (ECAI) 2016 conference (2016)
4. Bechhofer, S., Horrocks, I., et al.: Wonderweb: Ontology infrastructure for the semantic web (2004)
5. Carlson, A., Betteridge, J., Kisiel, B., Settles, B., Hruschka Jr, E.R., Mitchell, T.M.: Toward an architecture for never-ending language learning. In: AAAI. vol. 5, p. 3 (2010)
6. Charniak, E.: BLLIP 1987-89 WSJ Corpus Release 1 (2000)
7. Etzioni, O., Banko, M., Cafarella, M.J.: Machine reading. In: 21st National Conference on Artificial Intelligence. pp. 1517–1519. AAAI'06 (2006)
8. Etzioni, O., Cafarella, M., Downey, D., Kok, S., Popescu, A.M., Shaked, T., Soderland, S., Weld, D.S., Yates, A.: Web-scale information extraction in knowitall: (preliminary results). In: 13th International Conference on World Wide Web. pp. 100–110 (2004)
9. Fillmore, C.J.: Frame semantics, pp. 111–137. Hanshin Publishing Co., Seoul, South Korea (1982)
10. Lenat, D.B.: Cyc: A large-scale investment in knowledge infrastructure. Commun. ACM 38(11), 33–38 (1995)
11. Miller, G.A.: Wordnet: A lexical database for english. Commun. ACM 38(11), 39–41 (1995)
12. Pennacchiotti, M., Wirth, M.: Measuring frame relatedness. In: Lascarides, A., Gardent, C., Nivre, J. (eds.) EACL. pp. 657–665. The Association for Computer Linguistics (2009)
13. Rouces, J., de Melo, G., Hose, K.: FrameBase: Representing n-ary relations using semantic frames. In: Proceedings of ESWC 2015. pp. 505–521 (2015)
14. Speer, R., Havasi, C.: Representing general relational knowledge in conceptnet 5. In: LREC. pp. 3679–3686 (2012)
15. Suchanek, F.M., Kasneci, G., Weikum, G.: Yago: A core of semantic knowledge. In: Proceedings of the 16th International Conference on World Wide Web. pp. 697–706. WWW '07, ACM, New York, NY, USA (2007), http://doi.acm.org/10.1145/1242572.1242667
16. Tenorth, M., Beetz, M.: Knowrobknowledge processing for autonomous personal robots. In: Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on. pp. 4261–4266. IEEE (2009)
17. Wu, Z., Palmer, M.: Verbs semantics and lexical selection. In: Proceedings of the 32Nd Annual Meeting on Association for Computational Linguistics. pp. 133–138. ACL '94, Association for Computational Linguistics, Stroudsburg, PA, USA (1994), http://dx.doi.org/10.3115/981732.981751
18. Yates, A., Cafarella, M., Banko, M., Etzioni, O., Broadhead, M., Soderland, S.: Textrunner: Open information extraction on the web. pp. 25–26. NAACL-Demonstrations '07 (2007)