Knowledge Harvesting: Achievements and Challenges

Gerhard Weikum, Johannes Hoffart, Fabian Suchanek

Knowledge Harvesting: Achievements and Challenges

Gerhard Weikum[1], Johannes Hoffart[2], and Fabian Suchanek[3]

[1] Max Planck Institute for Informatics, weikum@mpi-inf.mpg.de
[2] Ambiverse GmbH, johannes@ambiverse.com
[3] Telecom ParisTech University, fabian@suchanek.name

Abstract. This article gives an overview on knowledge harvesting: automatically constructing large high-quality knowledge bases from Internet sources. The first part reviews key principles and best-practice methods. The second part points out open challenges for future research.

1 Introduction

Enhancing computers with "machine knowledge" that can power intelligent applications is a long-standing goal of computer science [34]. Major advances on *knowledge harvesting* – methods for turning noisy Internet content into crisp knowledge structures on entities and relations – have made this formerly elusive vision practically viable today.

A prominent use case where *knowledge bases (KB's)* have become a key asset is search engines. When we send a query like "jobs biography" to Bing or Google, we obtain information on the life of Steve Jobs. So the search engine automatically detects that we are interested in facts about an individual entity. On the other hand, for a query like "jobs in bay area", the search engine locates the spatial entity Bay Area and properly interprets the query as a request for local job ads. Finally, for the query "jobs at apple", the system returns a mix of two different interpretations. All this is feasible because the search engine has a huge knowledge base on its back-end servers, aiding in the discovery of entities in user requests (and their contexts) and in finding concise answers.

The KB's in this setting are centered on individual entities, containing:

- entities like people, places, organizations, products, events (e.g., `SteveJobs`, the `GoldenGateBridge`, the `Pixar` company, the `iPhone7`, the `WoodstockConcert`),
- the semantic classes to which entities belong (e.g., `SteveJobs type entrepreneur`, `SteveJobs type computerPioneer`, `SteveJobs type ZenBuddhist`),
- relationships between entities (e.g., `SteveJobs founded AppleInc`, `SteveJobs invented iPhone`, `SteveJobs diedOf PancreaticCancer`), as well as
- their validity times (e.g., `SteveJobs wasCEOof Pixar [1986,2006]`).

This concept of a comprehensive KB goes back to pioneering work in Artificial Intelligence on universal knowledge bases in the 1980s and 1990s, most notably, the *Cyc* project at MCC in Austin [35] and the *WordNet* project at

Princeton [19]. However, these knowledge collections have been hand-crafted and manually curated. Thus, knowledge acquisition was inherently limited in scope and scale. Starting this millenium with the Semantic Web vision, domain-specific *ontologies* [59] have been developed, but these are also manually created. In the last decade, *automatic knowledge harvesting* from Web and text sources has become a major research avenue, and has made substantial practical impact. Knowledge harvesting is the core methodology for the automatic construction of large knowledge bases, going beyond manually compiled knowledge collections like Cyc or WordNet.

These achievements are rooted in academic research and community projects. Salient projects that started ten to fifteen years ago are DBpedia [2], Freebase [5], KnowItAll [18], WebOfConcepts [11], WikiTaxonomy [50] and Yago [60]. More recent projects with publicly available data include BabelNet [44], ConceptNet [58], DeepDive [57], EntityCube (aka. Renlifang) [46], KnowledgeVault [15], NELL [7] Probase [73], Wikidata [68], XLore [70].

The largest general-purpose KB's with publicly accessible contents are BabelNet (`babelnet.org`), DBpedia (`dbpedia.org`), Wikidata (`http://wikidata.org`) and Yago (`yago-knowledge.org`). They contain millions of entities, organized in hundreds to hundred thousands of semantic classes, and hundred millions to billions of relational facts on entities. These and other knowledge resources are interlinked at the entity level, forming the Web of Linked Open Data [24].

Our own endeavor on knowledge harvesting was motivated by research on semantic search. Later it became the *Yago-Naga project*, with the first release of the Yago KB (`yago-knowledge.org`) in February 2007. The strength of Yago is its rich type system with hundred thousands of classes. When IBM Watson won the Jeopardy quiz show, it leveraged Yago's knowledge of fine-grained entity types for semantic type checking [28]. More recent Yago releases incorporated temporal and spatial knowledge [25] and multilingual properties [53]. Yago is maintained as a joint project of the Max Planck Institute for Informatics and the Télécom ParisTech University.

Over the last five years, knowledge harvesting has been adopted at big industrial stakeholders, and large KB's have become a key asset in a variety of commercial applications, including semantic search (see, e.g., [4]), analytics (e.g., aggregating by entities), recommendations and data integration (i.e., to combine heterogeneous datasets in and across enterprises). Examples are the Google Knowledge Graph (with Freebase as a catalyst), the use of KB's in IBM Watson, Amazon's Evi, the Baidu Knowledge Graph, Facebook's Graph Search, Microsoft Satori, Wolfram Alpha as well as domain-specific knowledge bases in business, finance, life sciences, and more (e.g., at Bloomberg, Mayo Clinic, Siemens, Walmart, etc.). In addition, KB's have found wide use as a distant supervision source for a variety of tasks in natural language processing, such as entity linking.

This article gives an overview of knowledge harvesting. Section 2 reviews research achievements and the state of the art, identifying key principles and best-practice methods. Section 3 presents open challenges that provide opportunities for future research.

2 Achievements

2.1 Knowledge Base Model

Knowledge representation has received great attention in AI research, leading to sophisticated forms of epistemic logics and, with the advent of the Semantic Web, description logics for ontologies [59]. However, none of these powerful models ever led to sizable collections of knowledge. The main reason then was the lack of data to populate knowledge models with interesting instances.

Entities, Classes and Relations: The former limitation of sparse data on instances was eventually overcome by the advent of knowledge-sharing communities like Wikipedia and the increasing availability of public datasets. This enabled the extraction of entities and their properties from high-quality Internet sources to populate a KB. Most of today's KB's use a simple relational representation where all knowledge is cast into grounded formulas of n-ary predicates (i.e., relation symbols). Constants (i.e., 0-ary predicates) denote *entities* or *literal values* (e.g., coordinates, dates and other numbers), unary predicates correspond to *classes* of same-type entities, and higher-arity predicates are used to capture entity *attributes* or *relationships* with other entities.

SPO Triples: The widely used RDF data model even restricts n-ary relations to be binary and casts the unary predicates for class membership into the binary form ⟨*entity*⟩ `type` ⟨*class*⟩ – in infix notation, with `type` being the predicate. The elemtary formulas in an RDF-compliant KB are also called *subject-predicate-object* triples, or *SPO triples* for short. S is required to be an entity, whereas O can be either an entity (e.g., someone's spouse) or a literal (e.g., someone's birthdate). The following shows examples for such triples (in conceptual form, disregarding the specifics of the RDF syntax):

`SteveJobs type computerPioneer`	`SteveJobs type entrepreneur`
`entrepreneur subclassOf businessperson`	`businessperson subclassOf person`
`SteveJobs hasDaughter LisaBrennan`	`AppleLisa namedAfter LisaBrennan`
`SteveJobs diedOf PancreaticCancer`	`SteveJobs diedOn 5-Oct-2011`

The two triples in the first line above are about class membership. In standard logics these would be written as unary-predicate formulas: *computerPioneer (SteveJobs)* and *entrepreneur (SteveJobs)*. This form of knowledge about *instances of classes* forms the backbone of a KB. The two triples in the second line above organize classes in a subsumption hierarchy of subtypes and supertypes. To avoid that this `subclassOf` predicate takes second-order form, the class predicates are cast into binary form with the `type` predicate. All triples for the `type` and `subclassOf` predicate constitute the *taxonomic knowledge* of a KB. Prior work on ontologies mostly focused on intensional knowledge centered on the `subclassOf` predicate, and hardly captured any instances for the `type` predicate.

Many KB's require that the subject arguments of `type` be *individual entities* (aka. named entities) that can be uniquely identified in the real world, this way

disregarding abstract entities (aka. general concepts) which are prone to subjective interpretation. For example, concepts such as universe, love or quantum physics are intentionally excluded to stay clear of potential pitfalls. This way, it has become feasible to construct huge KB's with millions of entities for many thousands of classes with negligible error rate and very high agreement on what is correct knowledge.

Taxonomic knowledge is already a huge asset for applications like search and analytics. For example, a query such as "Which singers have won the Nobel prize?" can be easily answered from the following triples (by intersecting instances of different classes):

`BobDylan type singer`	`BobDylan type NobelPrizeLaureates`

Likewise, an analytic task that compares frequencies of queries, clicks or references for musicians vs. politicians, is made easy by a KB that provides the class memberships of entities along with subclass knowledge such as:

`singer subclassOf musician`	`stateSecretary subclassOf politician`

SPO triples with predicates other than `type` or `subclassOf` are referred to as *facts*. The largest KB's contain billions of facts for several thousand different predicates. The predicates of interest (e.g., `bornIn, hasDaughter, namedAfter` etc.) are often pre-specified. We will reconsider this point in Section 2.2.5.

Beyond SPO Triples: KB's with this focus on binary relational facts are often called *knowledge graphs (KG's)*, as they correspond to labeled graphs with nodes denoting entities and predicate-labeled edges for the facts. However, the restriction to binary predicates is an oversimplification. There are many cases where ternary and higher-arity relations are needed to represent real-world situations. One important case is *temporal knowledge*: extending facts with their validity times, a timepoint or a timespan. For example, capturing when Steve Jobs (co-) founded Apple or from when until when he was CEO of his various companies calls for ternary predicates with an additional time argument:

`SteveJobs founded AppleInc 1-April-1976`
`SteveJobs wasCEOof AppleInc [9-July-1997, 24-Aug-2011]`

Generally, events often require three or more arguments to fully capture them: several entities involved in various roles, time, place, etc. As an example, consider the football match where Germany won 7:1 against Brazil on 8-July-2014 in Belo Horizonte. Such a composite fact cannot be broken down into binary facts without losing information. The RDF data model then usually escapes to so-called reification: assigning an entity id to the entire event, and adding multiple triples for the event's different aspects with the entity id as their common subject argument. However, this technique makes querying and reasoning with knowledge more tedious. Higher-arity predicates, on the other hand, are a natural representation.

Canonicalization: An important point in capturing crisp knowledge is the uniqueness of representing entities and facts. Many (but not all) KB's strive to

canonicalize all arguments in their SPO triples and other facts. For classes, this implies that differently named classes have (potentially) different instances (e.g., singer vs. songwriter vs. musician); conversely, differently named classes that would always agree on their instances should be unified (e.g., humans vs. people). For entities, the same principle applies: an entity can have different names, but should be captured only once with a unique id. For example, knowledge about the company Apple should not be registered twice or spread across different names like "Apple", "Apple Inc.", "Apple company". Instead, we need to map all relevant facts to the same entity, canoncially denoted, for example, as `AppleInc`, and keep its diverse surface names as *labels* (aka. alias names). This issue calls for *named entity disambiguation* (aka. entity linking) (see, e.g., [38, 56]). For populating a KB, this step can be integrated into a knowledge harvesting method or dealt with by specialized methods and tools.

The canonicalization principle carries over to entire facts. For example, a high-quality KB should represent the gist of the sentences *"Jobs was born in San Francisco."* and *"Apple's charismatic Steve is from the City by the Bay"* in a single fact `SteveJobs bornIn SanFrancisco`. Without this unique representation, querying a KB and using it for inference would be awkward.

Lessons Learned: A key point in enabling knowledge harvesting at large scale has been to use a simple SPO-style representation as a backbone. This is sufficient for a *core KB* to capture taxonomic classes and basic facts about millions of individual entities. We emphasize, however, that this does not rule out additional knowledge using higher-arity predicates and more sophisticated representations. In particular, a rich KB should also comprise intensional assertions about the world: constraints that couple different predicates or rules for deducing additional facts. For example, specifying that a person can have only one birthplace and that the birthplace must be an entity of type location, is a vital piece of knowledge. Obviously, this calls for more expressive predicate logics. We will come back to this in Section 2.2.4.

2.2 Knowledge Gathering and Cleaning

For a KB model as outlined above, the core task of knowledge harvesting is to i) identify appropriate data sources as input, ii) extract entities, classes and relational facts, and iii) organize them into a clean KB. This builds on techniques from the area of *information extraction (IE)*, based on patterns in Web pages and text documents (see [9, 55] for surveys). However, IE copes with one source as fixed input (e.g., one Web site), whereas knowledge harvesting has freedom in choosing its sources and can exploit redundancy and statistics across many sources. Important knowledge is often expressed in many sources in complementary ways. We can pick low-hanging fruit by choosing the best suitable sources and treat sources with different degrees of noise in very different ways.

This principle suggests a layered approach where we first tap sources with limited noise in content and structure, using robust extraction methods for high-quality output. This is why many KB's intensively build on harvesting semi-

structured elements of Wikipedia: category names, infoboxes, lists, headings, etc. For specific domains (e.g., health), there are often specific sources that should be prioritized (e.g., repositories such as Medscape, DrugBank, FAERS, etc.).

As a second stage, additional knowledge can then be harvested from riskier sources. We will see below that the previously compiled high-quality knowledge is beneficial in filtering noise and cleaning candidate facts. In other words, once we have a strong core KB, it is an asset in acquiring more knowledge, deeper knowledge and better knowledge – without degrading in quality.

2.2.1 Harvesting Entities and Classes

The first goal of knowledge harvesting is to compile a comprehensive set of semantic classes (e.g., guitarists, electric guitarists, left-handed guitarists, etc.) and their instances (i.e., individual entities such as Bob Dylan, Jimmy Page, Jimi Hendrix, etc.). In addition, we want to capture subsumptions among classes. We could address this task *ab initio*, starting with raw Internet contents, but it would be unwise to ignore pre-existing high-quality resources. First, classic work on linguistics and cognition led to *WordNet* [19], a large repository of words and word senses with lexical relations like synonymy, antonymy, hypernymy (i.e., subsumption), etc. This can be seen as a source of more than 100,000 semantically classes, carefully organized into a subclass/superclass taxonomy. Second, knowledge-centric communities like *Wikipedia* organize articles in category systems. While these were noisy in the early years of Wikipedia (with improper or misleading categories), the editorial guidelines and manual curation of Wikipedia have eventually led to a very rich and reasonably accurate system of about half a million categories. Given these prior assets, one line of successful research has derived taxonomic knowledge for high-quality KB's from WordNet and Wikipedia [44, 50, 60]. The Yago KB, for example, has carefully unified WordNet classes and Wikipedia categories and constructed a high-quality taxonomy with more than 300,000 fine-grained classes.

Once we settle on a class taxonomy, the next step is to populate the classes with individual entities (where one entity can belong to several classes). Here, Wikipedia is by far the largest and best asset to tap into, as it already organizes articles about entities within its category system. Most of the large KB's have seized this opportunity. It is straightforward when the taxonomy is based on Wikipedia categories. For connecting to WordNet classes, though, clever alignment and pruning techniques are needed [60]. The philosophy of "picking low-hanging fruit first" also carries over beyond Wikipedia. For example, Yago has integrated GeoNames (`geonames.org`) for spatial entities [25]. For health entities like diseases, symptoms, drugs, etc., manually curated sources like UMLS, MeSH, DrugBank, etc. can be used very effectively (see, e.g., [17]). Generally, entities of specific types (e.g., books, songs, medical drugs, etc.) can often be harvested from dedicated sources or specific identifier systems [62]. As a caveat, we note that this does not completely cover the world of *emerging entities* like

new events or people who suddenly become notable (e.g., a new singer). This aspect of knowledge dynamics will be discussed in Section 2.3.

A major alternative to relying on Wikipedia-style sources is to tap all kinds of Internet sources at large scale. A classic approach for this line of *taxonomy induction* research is to use so-called Hearst patterns like *X such as Y* or *X, Y and other Z* and match them against Web and text contents to extract class-entity or subclass-superclass pairs [23]. Using advanced data-mining and machine-learning techniques, this idea can be greatly generalized, to tap into additional patterns, http links and HTML tables (where a column name and cell value may indicate a class-entity pair), or into query-and-click logs of big search engines (see, e.g., [10, 31, 47, 67, 69, 73]). These are powerful methods; however, they require large-scale machinery and access to big data like complete Web crawls or search engine logs.

2.2.2 Gathering Facts from Wikipedia The most straightforward way of gathering binary facts is to harvest infoboxes in Wikipedia. These provide attributes and relationships of the entity featured in an article, in semi-structured form. Here is an example in the wiki markup language:

```
{{ Infobox person
    | name        = Steve Jobs
    | birth_date  = Birth date|1955|2|24|mf=y
    | birth_place = [[San Francisco]], California, U.S.
    | death_cause = [[Pancreatic cancer]] and [[respiratory arrest]]  }}
```

Naturally, these P and O components for SPO triples can be extracted by regular expressions (i.e., finite state automata). These expressions may even be automatically learned from samples with manual markup. Unfortunately, even Wikipedia infoboxes exhibit noise and terminological diversity. For example, birthplaces could be stated differently in different articles: `birth_place = ...`, `born_in = ...`, `born_in_city = ...`, etc.; and the values may be encoded in different ways (e.g., with or without the state in a country). To cope with this heterogeneity, *type-checking* the outputs of a regex matcher is a boost in quality [25, 60]. Here, having rich taxonomic knowledge – entities and their fine-grained types – is a huge benefit. The result is clean facts in canonicalized form.

There are other opportunities to extract facts from semi-structured elements (i.e., headings, tables, etc.) in Wikipedia or similar high-quality sources. As the diversity of how facts are expressed increases, this calls for stochastic variants of automata, like Conditional Random Fields (CRF's). Facts from infoboxes can be used to train CRF-based extractors (e.g., [72]).

2.2.3 Gathering Fact Candidates from Text

For high recall (i.e., gathering as many facts as possible), we eventually need to tap into natural language text as input, facing even higher degrees of noise and variability. In this setting, *pattern-based harvesting* has been the method of choice. For example, birthplaces of people are often expressed by phrases such

as "his birthplace is", "her hometown", "is from", etc. Of course, it would be daunting to manually specify such patterns for hundreds or thousands of predicates. Instead, a distantly supervised approach has become prevalent, centered on the principle of *pattern-fact duality* [1, 6, 18, 40]. For each relation of interest, a small set of *seed facts* is needed, for example, the correct birthplaces of a few prominent people (which could be obtained from semi-structured high-quality sources). These facts can be matched against a corpus (or the entire Internet) by searching for sentences that contain the subject entity and the object entity. The key idea then is that facts frequently co-occur with connecting phrases (e.g., verbal phrases) that can be distilled into patterns. The patterns in turn co-occur with other, newly seen facts. This procedure – alternating between facts and patterns – can be iterated, and eventually yields a large number of new fact candidates. Patterns can be surface phrases, but can also cover generalizations such as lifting the words "his" and "her" into personal pronouns, capture non-adjacent words, or use dependency parsing to consider the syntactic structure of sentences. Also, HTML tables in Web pages can be tapped for fact harvesting in a similar vein; this has been successfully pursued, for example, by the NELL and Knowledge Vault projects [15, 41].

Of course, this gathering process needs to be comlemented by computing statistical measures of confidence and support. Otherwise, spurious patterns may easily lead to drifting targets. For example, starting with Steve Job's birthplace San Francisco among the seed facts, the method could pick up the pattern "his favorite place" after a few iterations. By judicious thresholding, one can obtain a good set of assertions for facts. This approach typically results in an accuracy around 80% – that is, there are still 20% of the assertions incorrect, due to noise in the patterns.

2.2.4 Cleaning Fact Candidates

The high recall of the outlined gathering methods comes at the expense of potentially degrading in precision: introducing many false candidates for facts. For example, we may obtain the following assertions for the `birthplace` relation:

`SteveJobs birthplace SanFrancisco`	`SteveJobs birthplace USA`
`SteveJobs birthplace Kyoto`	`SteveJobs birthplace AppleInc`

These fact candidates may be derived from diverse patterns for the `birthplace` predicate, such as "his hometown", "citizen of", "his favorite place" or "mostly at". The resulting triples have to be cleaned: removing spurious assertions, and also mapping patterns of good triples – "his hometown" in the above case – onto canonicalized predicates. Next, we discuss methods for this purpose.

Consistency Constraints: A key idea is to mimic human skepticism: use plausibility considerations to rule out dubious assertions. In technical terms, we can impose *consistency constraints* on the candidate space, to discard assertions that violate certain conditions. The simplest idea is to enforce *type constraints*, and

this actually provides enormous mileage towards building a high-quality KB. For the above example, the relation about where people are born should be specified with a type signature `birthplace`: *person* × *location*, or more specifically `birthplace`: *person* × *city*. This constitutes a logical constraint:

$$\forall x, y\,(birthplace(x, y) \Rightarrow (type(x, person) \wedge type(y, city)))$$

The constraint is violated by 2 of the 4 candidate facts above, leaving only San Francisco and Kyoto as possible cities where Steve Jobs was born.

This logics-based approach is far more general than mere type checking. There are other kinds of constraints to be harnessed, most notably, *functional constraints* – many relations are actually functions – and *inclusion constraints* between different relations. For example, each person can have only one birthplace, and birthplaces are usually among the cities where a person has lived:

$$\forall x, y, z\,((birthplace(x, y) \wedge birthplace(x, z)) \Rightarrow y = z)$$
$$\forall x, y\,(birthplace(x, y) \Rightarrow livedIn(x, y))$$

These constraints have to be manually specified by a knowledge engineer. However, this is a fairly easy modeling task, and not a bottleneck at all. There are also techniques for automatically learning constraints from data (see, e.g., [7, 41]), but this comes at a higher risk. In general, constraints can be hard or soft: absolutely excluding any violation or tolerating a certain degree of exceptions. For example, the above formula that couples `birthplace` and `livedIn` is soft. In such cases, the constraints may be weighted. Fact candidates are weighted as well, typically by some notion of confidence based on statistics from the gathering stage. This opens the way to deciding between San Francisco and Kyoto for the birthplace of Steve Jobs.

Consistency Reasoning: Fact candidates should not solely be checked against each constraint in isolation. Instead, it is beneficial to perform *joint inference* over a set of assertions and a set of constraints. Consider, for example, the following noisy candidates and soft constraints, with the last two constraints stating that someone can either be a scientist or a musician, but never both:

`BobDylan hasWon Grammy`
`BobDylan hasWon LiteratureNobelPrize`
`BobDylan hasWon PhysicsNobelprize`

$\forall x\,(hasWon(x, Grammy) \Rightarrow type(x, musician))$
$\forall x\,(hasWon(x, PhysicsNobelPrize) \Rightarrow type(x, scientist))$
$\forall x\,(type(x, scientist) \Rightarrow \neg type(x, musician))$
$\forall x\,(type(x, musician) \Rightarrow \neg type(x, scientist))$

There is no way to keep all three fact candidates while enforcing all constraints. There are two different combinations of prizes, though, that are consistent. By viewing the data as a set of logical formulas, this becomes a test for *satisfiability*. By grounding the constraints with the constants from the candidate fact pool, the problem is reduced to an instance of the *MaxSat problem*.

Moreover, since each individual formula has a weight (see above), the task is to compute a *Weighted MaxSat* solution. Although this is a classical NP-hard problem, there are good approximation techniques and there are ways of customizing them to this specific task of knowledge cleaning [43, 61]. In the example, assume that the weight for the Grammy fact is much higher than the weight for the Physics Nobel Prize and the weights for the three constraints are identical. Then, the best solution is to accept the facts about the Grammy and Literature Nobel Prize while dropping the assertion about the Physics Nobel Prize.

Probabilistic Graphical Models: The above line of thought has been very fruitful for knowledge cleaning and comes in a variety of ways: MaxSat reasoning is one approach, integer linear programming another one, and there are also powerful probabilistic inference methods along these lines. The latter include especially probabilistic graphical models, where random variables for accepting or refuting fact candidates are coupled through logical constraints [14, 30]. In this setting, the MAP inference (MAP = maximum a posteriori) is equivalent to solving a weighted MaxSat problem. Approximation algorithms include SAT solvers, Monte Carlo sampling, variational calculus and more. Applications to knowledge harvesting have been developed, among others, by [7, 52, 57, 61, 74, 75]. Constraints have also been leveraged for estimating the confidence in specific extractions and for training fact extraction methods (see, e.g., [41]).

Beyond Triples: Methods of this kind can be further extended to go beyond binary relations. An important use case is *temporal knowledge* where facts need to be annotated with timepoints or timespans when they are valid [37, 63, 71]. For example, properly interpreting a fact such as `BobDylan spouseOf SaraDylan` requires the corresponding time scope `[Nov-1965, June-1977]`. So strictly speaking, we are looking at a ternary relation here: `spouseOf`: *person* × *person* × *time*. This is a special case of *higher-arity relations*, often but not only in combination with entities of type *event*. Knowledge harvesting methods, as outlined above, can be further extended to this end (see, e.g., [32]).

2.2.5 Other Approaches

A potential concern about the above methods is their limitation to prespecified predicates. For example, we can harvest composers of songs or artists who covered songs only after a human curator provides the relevant predicates with type signatures: `composedMusic`: *musician* × *song* and `coveredMusic`: *musician* × *song*. On the other hand, facts on song lyrics being about specific people cannot be harvested unless we explicitly model such a predicate. Therefore, we refer to the presented methods as *model-driven knowledge harvesting*.

Open IE: The paradigm of *Open Information Extraction (Open IE)* [3, 12, 39] offers unsupervised harvesting of fact triples in an open-ended manner, without any modeling effort. Using linguistic patterns, Open IE collects all kinds of triples where S, P and O are meaningful phrases, typically noun phrases for S and O and verbal phrases for P. These are not canonicalized; so outputs could be:

"Dylan's Hurricane"	"covers the story of"	"the black boxer Carter"
"Hurricane"	"is a protest song about"	"racist victim Carter"
"Sara"	"is a love song about"	"Dylan's wife"
"the love song Sara"	"is about"	"his ex-wife Sara Dylan"

Such output requires disambiguating the S and O arguments of the phrase-level triples, which is typically an entity-linking task [56] against an existing KB of entities. Canonicalizing the P components, on the other hand, is an open challenge as the space of possible predicates is unknown in this open-ended setting. Research along these lines, based on clustering techniques, includes [20].

Deep Learning: With recent breakthroughs in deep learning [33], an intriguing thought could be to bypass the explicit construction of a KB, and rather use end-to-end learning on a per-task basis (e.g., question answering or describing videos). However, this raises caveats. First, it would require huge amounts of labeled training data which are often unavailable. Second, this expensive training would have to be repeated for every new task. Third, machine learning outputs (i.e., predictions, recommendations, answers or even decisions) are not easily explainable to human users. Explicit KB's, on the other hand, are a reusable asset for many tasks, they can inform and constrain the learning of models, and they support user-comprehensible explanations.

2.3 Knowledge Evolution and Quality

Change is the only constant in knowledge. Attribute values of entities (e.g., city populations) and relationships between entities (e.g., the CEO of a company or a person's spouse) change over time. Moreover, new entities of interest are created all the time and need to be added to the KB (e.g., new songs, sports matches, babies of celebrities). Also, existing entities may be irrelevant for a KB at some point, but become prominent at a later point. Typical examples are when a "garage band" or "garage company" becomes succesful. If Wikipedia had already existed in 1976, it would probably have dismissed Apple for insufficient notability.

Temporal Knowledge: So KB's must be continuously updated. This requires keeping *versions of facts*, along with their *temporal validity* scopes. We already discussed methods for harvesting temporal knowledge in Section 2.2.4. Some of the major KB's have rigorously followed this principle (e.g., [25]).

Active Knowledge: For some kinds of highly dynamic and specialized knowledge, explicitly capturing fact versions is not practically feasible. For example, the chart positions of a song and the box office counts of a movie change so rapidly that it is hardly meaningful to materialize such facts in the KB. Instead, a preferable way is to keep links to specialized databases and to Web services that return up-to-date values on demand [51].

Emerging Entities: A specific aspect of dynamic knowledge is to cope with newly emerging entities. When discovering entity names in input sources (text, Web tables, etc.), we first aim to disambiguate them onto the already known

entities in the KB. However, even if there is a good match in the KB, it is not necessarily the proper interpretation. For example, when the documentary movie "Amy" was first mentioned a few years ago, it would have been tempting to link the name to the soul singer Amy Winehouse. However, although the movie is about the singer's life, the two entities must not be confused. To rectify this situation, each observed name should always be potentially associated with an additional virtual candidate: *none* of the known entities [26]. When the evidence for the name denoting an *out-of-KB entity* is stronger than for a known entity, we capture the name as a new entity, along with its context. After a while, we will thus obtain a repository of recently emerging entity names. Then, clustering techniques can be used to group the names, and knowledge curators can be asked to confirm these canoncialized groups. Finally, the confirmed entities can be registered in the KB. To keep the effort for the curation step as low as possible, new entity candidates should be presented with informative context [27].

On-the-fly Knowledge Bases: A specific case for out-of-KB entities is constructing ad-hoc KB's on the fly. Suppose a new corpus of documents becomes available, for example, the Panama Papers or a batch of articles on a hot political or health topic, such as the UK Brexit or Zika infections. Then we should be able to automatically build a domain- and corpus-specific KB overnight, to support journalists and analysts in exploring the topic and analyzing particularly interesting issues.

Knowledge Curation: As a KB is continuously updated and keeps growing, it is virtually inevitable that errors sneak in and degrade the KB quality. Versioning of facts helps to control the maintenance process, but may also lead to conflicting versions. For example, when a new name for the CEO for a company is detected by harvesting fresh online sources, should this be added as a new fact or is it, perhaps, evidence that the previous fact in the KB was incorrect? One case leads to a new version, the other should result in overwriting the prior version. Even worse, an error may be detected only post-hoc, days or weeks after a new version was added and became interlinked with other facts.

Thus, *quality assurance* requires a fundamental solution. Fact cleaning, as discussed in Section 2.2.4 is an important element, and so is versioning. However, more is needed for a comprehensive approach. Today, very large KB's resort to *manual curation*, by having human volunteers (e.g., in an online community like Wikidata) or paid workers (in a commercial KB) checking newly added or altered facts. Such a *crowdsourcing* solution can be orchestrated in various ways, with the goal of optimizing the benefit/cost ratio (e.g., [29]).

Fact Checking: An alternative or complement to human curators is to harness the sources of evidence for doubtful facts in a more principled manner. This may entail actively searching for evidence or counter-evidence about facts, as a continuous background process. A typical approach is to consider a small number of alternative O values for given S and P of a fact, such as birthplaces for Steve Jobs: San Francisco vs. Cupertino vs. Kyoto. Then statistics derived from Internet sources (i.e., databases, Web sites, news, etc.) can guide the analysis and assessment of candidates towards finding the truth (see [36] for a survey).

This form of *knowledge corroboration* or *knowledge fusion* aggregates the observation confidence from different sources, where sources are weighted by their *trustworthiness*. Therefore, reasoning on truth (of statements) and trust (of sources) are often intertwined. Not surprisingly, joint inference methods, based on probabilistic models, have been pursued to this end (e.g., [16, 42, 48, 49]).

3 Challenges

Notwithstanding the great advances of knowledge harvesting over the last decade, there are major challenges left open – raising the bar for what computers should know. In the following, we discuss a few strategic challenges, pointing out opportunities for future research.

3.1 Knowledge Base Coverage

No matter how large a KB can grow, it will never be fully complete. The gaps, relative to an ideal KB, take different forms as discussed next.

Locally Incomplete Knowledge: This form of incompleteness can be formally characterized by referring to SPO triples that are in the KB and triples that should ideally be included but are missing. The most obvious case is when some O values are absent for a given S and P value – for example, when the KB contains some movies of a director but not all of them. Another case is when for a given P value, we have O values for some S but not all of them – for example, knowing spouses of some people but not knowing any spouses for other married people. The challenge here is not just to fill these gaps, but to realize when and where gaps exist. Reasoning over locally closed worlds is one recent approach [22]. More research is needed to equip KB's with self-reflection abilities to automatically detect their own gaps.

Long-tail Entities and Classes: There are many lesser known musicians, regional politicians and good but not exactly famous scientists. How can we identify these long-tail entities in the Internet, and harvest facts about them? Long-tail classes pose a similar problem: even with hundred thousands of classes in some KB's, one could always add more interesting ones. For example, what if we want to capture classes like `YogaPractitioners` or `BobDylanFans` (both of which would have Steve Jobs as a member)? Where in the class taxonomy should these classes be placed, and how can we find their instances?

Missing Facts: KB's have largely been constructed in an opportunistic manner, tapping into Wikipedia and its semi-structured data elements. If something is not explicitly said in Wikipedia or stated only in sophisticated form in the article's text, most KB's will miss it. This has resulted in high coverage of elementary facts like birthdates, marriages, albums of musicians, etc., but has neglected a diversity of salient facts that stand out to a human but are not easily captured by a machine. For example, what is notable about Bob Dylan's album Blonde on Blonde? KB's offer the release date, the list of songs, etc. To a human, however,

salient properties are, for example, that this was one of the first albums on which Dylan "went electric" (using electric instruments – irritating many of his folk music fans), and that its song "Sad Eyed Lady of the Lowlands" is about Dylan's wife Sara. The song was later covered by Joan Baez, who had a romantic relationship with Bob Dylan in the early 1960s and with Steve Jobs in the early 1980s. Her song "Diamonds and Rust" is about Bob Dylan. None of this is captured by any KB; most do not even have the proper predicates like `romanceWith` or `songIsAbout`.

3.2 Commonsense, Rules and Socio-Cultural Knowledge

Automatically constructed KB's have mostly focused on harvesting encyclopedic fact knowledge. However, for semantic search and other intelligent applications (e.g., conversational bots in social media), machines need a broader understanding of the world: properties of everyday objects, human activities, plausibility invariants and more. This overriding goal calls for various research directions.

Commonsense: One objective is to distill commonsense from Internet sources. This is about properties of objects like size, color, shape, parts or substance of which an object is made of, etc., and knowledge on which objects are used for which activities as well as when and where certain activities typically happen. For example, a rock concert involves musicians, instruments – almost always including drums and guitars, speakers, a microphone for the singer; the typical location is a stage, and so on. This background knowledge is beneficial for the interpretation of user questions, and also for retrieving images and videos when queries refer to abstractions or emotions that cannot be directly matched by captions, tags or other text. Today's search engines perform poorly on queries such as "exhausted band at hippie concert" (where users may want to find footage of concerts by the Grateful Dead or the Doors). Recent work on acquiring commonsense includes ConceptNet [58] and WebChild [64]; research with the specific focus of organizing knowledge on human activities includes [65]. There is, however, still a long way to go for computers to learn what every child knows.

Visual Knowledge: Commonsense knowledge is often more expressed in visual form than in textual sources; for example, think of colors, shapes and sizes of objects. This observation entails the dual goals of i) tapping visual contents like images and videos for acquiring knowledge and ii) constructing a KB *about* visual properties. Along the latter lines, ImageNet [13] is the most notable endeavor, which has populated a large fraction of WordNet classes with images. The NEIL project [8] has a gone a step further by extracting visual relationships between objects. WebChild has acquired different kinds of part-whole knowledge, at large scale, from combining textual and visual cues [66]. This is an instantiation of the general challenge of jointly distilling knowledge from vision and language (e.g., from movie scenes and their narratives [54]).

Rules: Another key element for advancing the intelligent behavior of machines is to capture invariants over certain kinds of facts, in the form of logical rules. For example, a rule about scientists and their advisors could state that the

advisor be on the faculty of the scientist's alma mater – as of the time when the scientist graduated. Similarly to the role of constraints in fact cleaning (see Section 2.2.4), rules can have exceptions. But regardless of their soft nature, rules can be a great asset to answer more queries and to infer additional facts for *KB completion*. For example, a person who has or had a position on the government of a certain country, is most likely a citizen of that country. To acquire this kind of intensional knowledge, rule mining methods have been applied to large KB's [21]. However, the state of the art has major limitations: rules are restricted in their logical form to Horn clauses or at least clauses. This disallows rules with existential quantifiers or with disjunctions in the rule head; for example, expressing that every human person has a mother and that every human is male or female, would be beyond the current scope. An alternative approach to KB completion, which bypasses logical representations, is to start with a tensor of SPO triples and use matrix or tensor factorization methods to predict additional triples (similarly to recommender systems) [45]. However, the derivation of new facts is not easily explainable with such methods.

Commonsense rules were already in the focus of the seminal Cyc project [35], but Cyc relied on human experts to manually specify logical axioms. A major challenge with automatic rule mining arises from the *open world assumption* that underlies KB's and the *bias in observations* from Internet sources. For example, if a KB does not contain any person who has won two different Nobel prizes, this should not imply a functionality constraint or cardinality constraint. In fact, Marie Curie is a counterexample anyway, but some KB's may have only a partial list of her awards. Likewise, a KB may have a restricted view on the wealth of entrepreneurs: for example, all founders of IT companies have become billionaires. This can be caused by the bias in the KB construction (e.g., by harvesting only successful entrepreneurs from Wikipedia), and should not entail a rule in the open world.

Socio-Cultural Knowledge: Another dimension where today's KB's have a huge gap is the socio-cultural context of facts or rules. Consider statements on people making discoveries and inventions. On first glance, one would expect that these are objective and universally agreed upon. On second thought, however, it becomes clear that it depends on the background and viewpoint of users. For example, most people in the US would say that the computer was invented by Eckert and Mauchley, whereas a German would give the credit to Konrad Zuse and a British may point out Alan Turing (or perhaps Charles Babbage). This depends not just on geographical context; for example, teenagers may widely think of Steve Jobs as the (re-) inventor of the (mobile) computer. For commonsense knowledge, it is even more critical to capture socio-cultural contexts. For example, shaking hands when people meet is the usual way of welcoming someone only in parts of the world. In other regions, people often hug or kiss each other (e.g., in France), or make gestures with both hands (e.g., in Thailand).

4 Conclusion

Knowledge harvesting has made great impact in enabling the automatic construction of large knowledge bases, sometimes called knowledge graphs. These have become essential assets in search and analytics over Internet contents and enterprise data. In addition to reviewing the underlying methodological achievements, this article has pointed out open challenges towards the next level of machine knowledge.

The success of deep learning, to enable smart computer behavior by training on raw data, may open up new perspectives on knowledge harvesting as well. Do computers need this kind of explicit knowledge representation at all, or is task-specific end-to-end learning in a sub-symbolic manner sufficient? We believe that machine knowledge and machine learning are complementary assets: the more you know the better you learn, and better learning enables acquiring more and deeper knowledge. A final challenge and research opportunity is to explore this potential synergy.

References

1. E. Agichtein, L. Gravano: Snowball: Extracting Relations from Large Plain-Text Collections. ACM DL 2000
2. S. Auer et al.: DBpedia: A Nucleus for a Web of Open Data. ISWC 2007
3. M. Banko et al.: Open Information Extraction from the Web. IJCAI 2007
4. H. Bast, B. Buchhold, E. Haussmann: Semantic Search on Text and Knowledge Bases. Foundations & Trends in Information Retrieval 10(2-3), 2016
5. K. Bollacker et al.: Freebase: a Collaboratively Created Graph Database for Structuring Human Knowledge. SIGMOD 2008
6. S. Brin: Extracting Patterns and Relations from the World Wide Web. WebDB 1998
7. A.J. Carlson et al.: Toward an Architecture for Never-Ending Language Learning. AAAI 2010
8. X. Chen, A. Shrivastava, A. Gupta: NEIL: Extracting Visual Knowledge from Web Data. ICCV 2013
9. L. Chiticariu, Y. Li, F. Reiss: Transparent Machine Learning for Information Extraction: State-of-the-Art and the Future. Tutorial at EMNLP 2015
10. M. Craven et al.: Learning to Construct Knowledge Bases from the World Wide Web. Art. Intell. 118(1), 2000
11. N. Dalvi et al.: A Web of Concepts. PODS 2009
12. L. Del Corro, R. Gemulla: ClausIE: Clause-based Open Information Extraction. WWW 2013
13. J. Deng et al.: ImageNet: A Large-Scale Hierarchical Image Database. CVPR 2009
14. P. Domingos, D. Lowd: Markov Logic: An Interface Layer for Artificial Intelligence. Morgan & Claypool 2009
15. X.L. Dong et al.: Knowledge Vault: A Web-Scale Approach to Probabilistic Knowledge Fusion. KDD 2014
16. X.L. Dong et al.: Knowledge-Based Trust: Estimating the Trustworthiness of Web Sources. PVLDB 8(9), 2015

17. P. Ernst, A. Siu, G. Weikum: KnowLife: a Versatile Approach for Constructing a Large Knowledge Graph for Biomedical Sciences. BMC Bioinformatics 16(157), 2015
18. O. Etzioni et al.: Unsupervised Named-Entity Extraction from the Web: an Experimental Study. Art. Intell. 165(1), 2005
19. C. Fellbaum, G. Miller: WordNet: An Electronic Lexical Database. MIT Press 1998
20. L. Galarraga et al.: Canonicalizing Open Knowledge Bases. CIKM 2014
21. L. Galarraga et al.: Fast Rule Mining in Ontological Knowledge Bases with AMIE+. VLDB J. 24(6), 2015
22. L. Galarraga et al.: Predicting Completeness in Knowledge Bases. WSDM 2017
23. M. Hearst: Automatic Acquisition of Hyponyms from Large Text Corpora. COLING 1992
24. T. Heath, C. Bizer: Linked Data: Evolving the Web into a Global Data Space. Morgan & Claypool 2011
25. J. Hoffart et al.: YAGO2: A Spatially and Temporally Enhanced Knowledge Base from Wikipedia. Art. Intell. 194, 2013
26. J. Hoffart, Y. Altun, G. Weikum: Discovering Emerging Entities with Ambiguous Names. WWW 2014
27. J. Hoffart et al.: The Knowledge Awakens: Keeping Knowledge Bases Fresh with Emerging Entities. WWW 2016
28. IBM Journal of Research and Development 56(3/4): Special Issue on "This is Watson". 2012
29. A. Kobren et al.: Getting More for Less: Optimized Crowdsourcing with Dynamic Tasks and Goals. WWW 2015
30. D. Koller, N. Friedman: Probabilistic Graphical Models. MIT Press 2009
31. Z. Kozareva, E.H. Hovy: Learning Arguments and Supertypes of Semantic Relations Using Recursive Patterns. ACL 2010
32. S. Krause et al.: Large-Scale Learning of Relation-Extraction Rules with Distant Supervision from the Web. ISWC 2012
33. Y. LeCun, Y. Bengio, G. Hinton: Deep Learning. Nature 521, 2015
34. D. Lenat, E. Feigenbaum: On the Thresholds of Knowledge. Art. Intell. 47(1), 1991
35. D. Lenat: CYC: A Large-Scale Investment in Knowledge Infrastructure. Comm. ACM 38(11), 1995
36. Y. Li et al.: A Survey on Truth Discovery. SIGKDD Explorations 17(2), 2015
37. X. Ling, D.S. Weld: Temporal Information Extraction. AAAI 2010
38. X. Ling, S. Singh, D.S. Weld: Design Challenges for Entity Linking. TACL 3, 2015
39. Mausam et al.: Open Language Learning for Information Extraction. EMNLP-CoNLL 2012
40. M. Mintz et al.: Distant Supervision for Relation Extraction without Labeled Data. ACL/IJCNLP 2009
41. T. Mitchell et al.: Never-Ending Learning. AAAI 2015
42. S. Mukherjee, G. Weikum, C. Danescu-Niculescu-Mizil: People on Drugs: Credibility of User Statements in Health Communities. KDD 2014
43. N. Nakashole, M. Theobald, G. Weikum: Scalable Knowledge Harvesting with High Precision and High Recall. WSDM 2011
44. R. Navigli, S. Ponzetto: BabelNet: The Automatic Construction, Evaluation and Application of a Multilingual Semantic Network. Art. Intell. 193, 2012
45. M. Nickel et al.: A Review of Relational Machine Learning for Knowledge Graphs. Proc. IEEE 104(1), 2016
46. Z. Nie, J.-R. Wen, W.-Y. Ma: Statistical Entity Extraction From the Web. Proc. IEEE 100(9), 2012

47. M. Pasca: Open-Domain Fine-Grained Class Extraction from Web Search Queries. EMNLP 2013
48. J. Pasternack, D. Roth: Latent Credibility Analysis. WWW 2013
49. K. Popat et al.: Where the Truth Lies: Explaining the Credibility of Emerging Claims on the Web and Social Media. WWW 2017
50. S. Ponzetto, M. Strube: Deriving a Large-Scale Taxonomy from Wikipedia. AAAI 2007
51. N. Preda et al.: Active Knowledge: Dynamically Enriching RDF Knowledge Bases by Web Services. SIGMOD 2010
52. J. Pujara et al.: Knowledge Graph Identification. ISWC 2013
53. T. Rebele et al.: YAGO: a Multilingual Knowledge Base from Wikipedia, Wordnet, and Geonames. ISWC 2016
54. A. Rohrbach et al.: A Dataset for Movie Description. CVPR 2015
55. S.Sarawagi: Information Extraction. Foundations & Trends in Databases 1(3), 2008
56. W. Shen, J. Wang, J. Han: Entity Linking with a Knowledge Base: Issues, Techniques, and Solutions. IEEE Trans. Knowl. Data Eng. 27(2), 2015
57. J. Shin et al.: Incremental Knowledge Base Construction Using DeepDive. PVLDB 8(11), 2015
58. R.Speer, C.Havasi: Representing General Relational Knowledge in ConceptNet 5. LREC 2012
59. S. Staab, R. Studer: Handbook on Ontologies. Springer 2009
60. F. Suchanek, G. Kasneci, G. Weikum: YAGO: a Core of Semantic Knowledge. WWW 2007
61. F. Suchanek, M. Sozio, G. Weikum: SOFIE: a Self-Organizing Framework for Information Extraction. WWW 2009
62. A. Talaika et al.: IBEX: Harvesting Entities from the Web Using Unique Identifiers. WebDB 2015
63. P. Talukdar, D. Wijaya, T. Mitchell: Coupled Temporal Scoping of Relational Facts. WSDM 2012
64. N. Tandon et al.: WebChild: Harvesting and Organizing Commonsense Knowledge from the Web. WSDM 2014
65. N. Tandon et al.: Knowlywood: Mining Activity Knowledge From Hollywood Narratives. CIKM 2015
66. N. Tandon et al.: Commonsense in Parts: Mining Part-Whole Relations from the Web and Image Tags. AAAI 2016
67. P. Venetis et al.: Recovering Semantics of Tables on the Web. PVLDB 4(9), 2011
68. D. Vrandecic, M. Krötzsch: Wikidata: a free collaborative knowledgebase. Commun. ACM 57(10), 2014
69. R.C. Wang, W.W. Cohen: Iterative Set Expansion of Named Entities Using the Web. ICDM 2008
70. Z. Wang et al.: Xlore: a Large-Scale English-Chinese Bilingual Knowledge Graph. ISWC 2013
71. Y. Wang et al.: Coupling Label Propagation and Constraints for Temporal Fact Extraction. ACL 2012
72. F. Wu, R. Hoffmann, D. Weld: Information Extraction from Wikipedia: Moving Down the Long Tail. KDD 2008
73. W. Wu et al.: Probase: a Probabilistic Taxonomy for Text Understanding. SIGMOD 2012
74. L.Yao et al.: Structured Relation Discovery using Generative Models. EMNLP 2011
75. J. Zhu et al.: StatSnowball: a Statistical Approach to Extracting Entity Relationships. WWW 2009